Análise sub-simbólica em SInCoPA
Discussão e implementações

Cristiano Figueiró
IIHAC/UFBA
figocris@gmail.com

Resumo: Nesse artigo irei abordar uma parte do desenvolvimento de meu sistema pessoal de composição, performance e análise (SInCoPA). O objetivo do sistema é auxiliar na criação de música interativa onde se busca uma polifonia densa com variações de textura nos diálogos do instrumentista com o computador. Ao longo do texto vou apresentar alguns módulos de análise realizada a partir de sinal de áudio. SInCoPA é construído com a linguagem Pure data (Pd). As abstrações apresentadas podem ser combinadas com outros programas e projetos de arte interativa.

Palavras-chave: música interativa; análise sub-simbólica; pd

Sub-symbolic Analysis in SinCoPA - Discussion and implementations

Abstract: In this article I will address one part of the development of my personal system of composition, performance and analysis (SInCoPA). The system goal is to assist the creation of interactive music where one seeks polyphony with a dense texture variations in dialogues between the computer and the performer. Throughout the text I will present the analysis modules made from the audio signal. SInCoPA is built with the language Pure Data (Pd). The abstractions presented can be combined with other software and projects for interactive arts.

Keywords: interactive music; sub-symbolic analysis; pd

SInCoPA

SInCoPA é um conjunto funções modulares que auxiliam a criação de música interativa. Cada função corresponde a uma abstração de Pure data (Pd). Pela característica modular de SInCoPA, as abstrações podem ser combinadas com outros programas e usadas em outros projetos como por exemplo instalações audiovisuais e programas de educação musical.

O conjunto de funções se divide em 3 grupos de abstrações de análise, cenário de interação (mapeamento) e geradores musicais. SInCoPA é disponibilizada como software livre através da licença GPL e seu código pode ser acessado gratuitamente[1]. Atualmente SInCoPA já foi testada em diferentes distribuições de Linux e OSX e algumas abstrações já funcionam em outros sistemas operaconais como Windows e Android.

Nesse artigo vou expor o atual estado de desenvolvimento de algumas funções de análise melódica e rítmica. O objetivo musical de SInCoPA é a criação de uma polifonia densa, altamente responsiva em relação ao estímulo musical realizado através da captação de áudio de um instrumento tradicional. Dentro dessa proposta, fez-se necessário o estabelecimento de métodos de análise sub-simbólica que dessem conta do mapeamento do sinal de áudio e de

1 Disponível para download em www.github.com/cristianofigo/sinc_abs

aspectos gerais do comportamento musical como por exemplo índices de estabilidade rítmica e permeabilidade melódica. Os resultados dessa análise alimentam parâmetros de geradores musicais que realizam um diálogo sonoro com o instrumentista em tempo-real.

ANÁLISE DE ÁUDIO COM [SINC-AUDIOANALISE]

Um dos fatores mais importantes para a prática de análise de áudio em tempo real é a flexibilidade de refinamento dos parâmetros de análise. Esses parâmetros devem estar ao alcance rápido e documentados e sinalizados na interface. O objetivo é reunir num só módulo, a detecção de ataques de notas (baseado no objeto [bonk~]) com a análise de frequências baseada no objeto [sigmund~], e ainda ligando o resultado das análises com objetos de conversão MIDI.

Nessa abstração procurou-se desenvolver uma interface que facilite a rápida prototipação e flexibilidade de parâmetros que podem se adaptar facilmente para diferentes fontes sonoras.

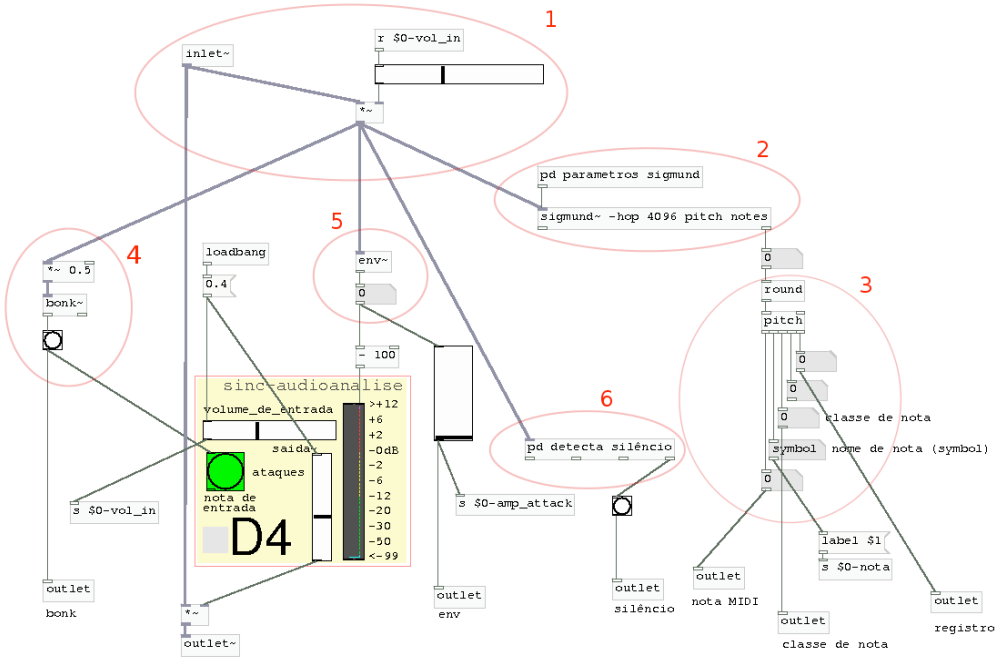

Fig. 1- Abstração [sinc-audioanalise].

Como podemos ver na área 1 da figura 1, temos o controle de um slider vertical(objeto [hslider]) controlando a amplitude geral do áudio de entrada. Esse controle é muito importante em situações em que se tem variações de amplificação entre ensaios e performance.

Na área 2 aparece um subpatch que pode ser visto na figura 2. Nesse subpatch podemos ver a organização dos parâmetros do objeto [sigmund~].

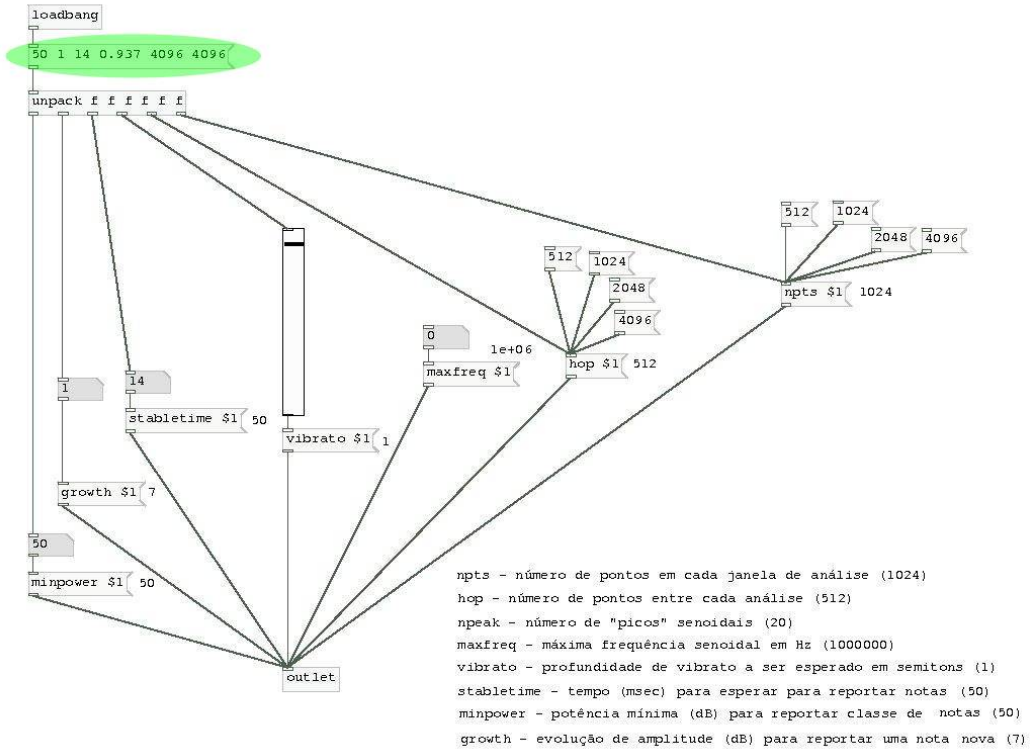

Fig. 2 – Sub-patch [pd parametros sigmund] dentro do patch visto na figura 1.

O objeto [sigmund~] faz análise de áudio no domínio da frequência e detecção de notas (PUCKETTE, 2009). Os parâmetros podem ser re-definidos em tempo-real através dos argumentos de criação do objeto ou através de mensagens como é o caso aqui. Nesse caso optou-se por pré-inicializar [sigmund~] com as opções : "*-hop 4096 pitch notes*" . Através de testes empíricos preferiu-se usar apenas a saída de "*notes*" por apresentar uma saída mais precisa para notas musicais. A mensagem em destaque na figura 2 representa a inicialização dos valores de todos parâmetros de [sigmund~].

CONTORNOS MELÓDICOS

As operações com contornos são uma ferramenta poderosa para composição musical. A manipulação de contornos pode ser aplicada a qualquer parâmetro musical. No Pd, qualquer dado pode facilmente escrito e lido em arrays gráficos. A análise de contornos melódicos pode dar pistas de como o discurso musical é estruturado e numa perspectiva de interação musical, influenciar geradores musicais na emergência de padrões melódicos. A primeira operação para análise de contornos é a redução para a forma normal. Segundo Silva:

> A forma normal de um contorno ocorre quando seus elementos aparecem em ordem temporal de ocorrência, e com valores definidos de 0 (menor valor) a n − 1 (maior valor),
>
> onde n representa o número de elementos do contorno .[...] Esta operação consiste em redefinir o elemento de menor valor para 0, o segundo elemento de menor valor para 1, o terceiro para 2, e assim por diante. Por exemplo, o contorno P(5 9 6 8) tem forma normal F(0 3 1 2), pois P0 , elemento de menor valor, é redefinido para 0, P2, segundo elemento de menor valor, para 1, P3 , terceiro de menor valor, para 2, e P1 para 3. (Silva, 2008)

Na figura 3, podemos ver um patch que faz a redução do contorno à forma normal. Nesse patch, o array “contorno_original”, tem oito elementos, variando de 0 a 127 podendo ser

alimentado pelo valor de pitch de um fluxo MIDI. A forma normal é calculada dentro do sub-patch [pd function contorno normal], e após o cálculo o resultado é escrito no array abaixo “contorno_normal”.

Fig. 3 – Redução de um contorno à forma normal.

Na figura 4 vemos a operação de redução à forma normal. Onde primeiro é feito uma leitura do array original com [until] e [tabread]. Após a leitura é calculado o valor mínimo e máximo com o objeto [list-minmax] e os dois valores são enviados com as variáveis globais [s min] e [s max]. O mapeamento dos valores é feito de forma linear. Nesse caso, o menor valor irá equivaler a zero e o maior valor equivaler a oito, que é o número de elementos do array. Ao final ainda os valores são convertidos para valores inteiros, antes de serem escritos no array de resultado.

A análise da similaridade entre contornos é uma ferramenta importante dentro de um sistema de composição interativa, pois pode revelar emergência e dissolução de padrões de execução e improvisação. Apresentamos aqui um primeiro experimento de análise de similaridade, aplicado em contornos de apenas três elementos. A idéia é alimentar um banco com três contornos estocados e a cada novo contorno que chega, o sistema devolve o índice do contorno estocado mais similar.

A interface do experimento pode ser vista na figura 5. A primeira etapa do experimento é a redução de todos contornos à forma normal.

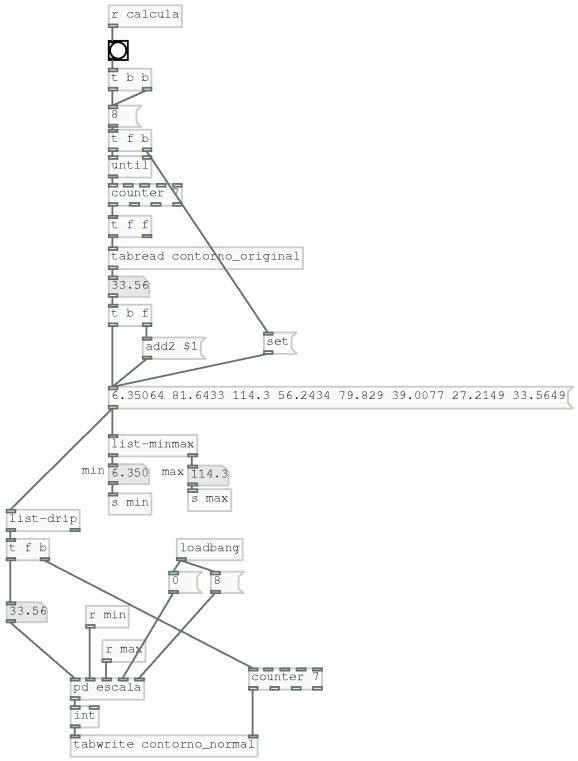

Fig. 4 – Operação de redução à forma normal.

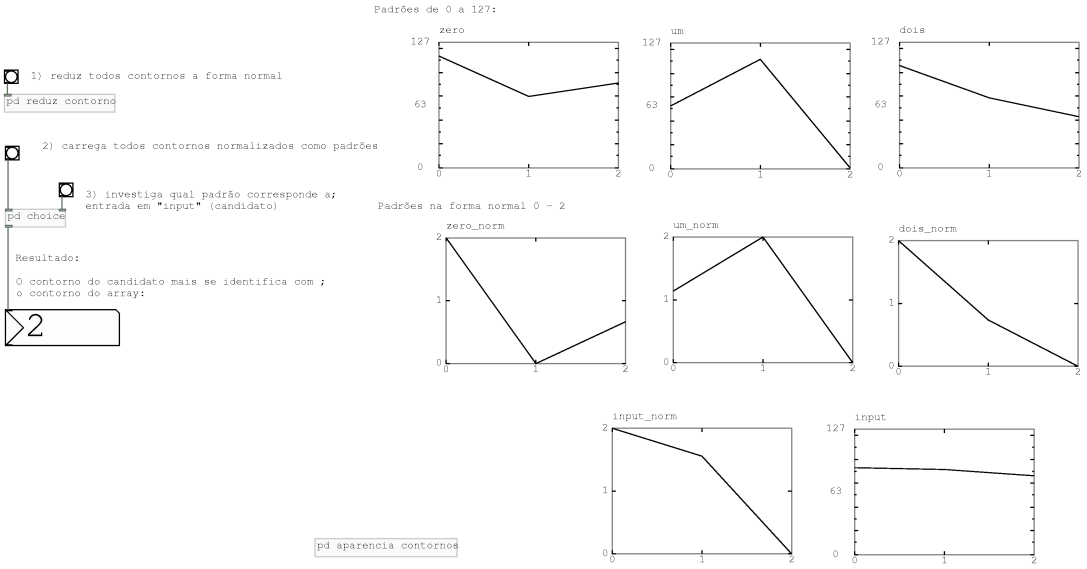

Fig. 5 – Similaridade entre contornos.

Na figura 6 vemos o interior do sub-patch [pd choice]. Onde os três contornos na forma normal são enviados ao objeto [choice], cada um precedido da mensagem “add”. A cada lista que chega em [choice], o objeto retorna o valor da lista estocada que mais se aproxima. O objeto [choice] estoca uma lista de vetores, cada um podendo ter até dez elementos. Quando mandamos uma nova lista de números, é retornado o índice do vetor conhecido que se combina mais proximamente com a nova lista. A qualidade da combinação é o produto interno dos dois vetores após outra normalização interna. O vetor que tem a direção mais próxima da lista/vetor de entrada ganha e seu índice é enviado para a saída. Esse objeto pode ser usado em diversas outras situações de análise em tempo-real e sua interface simples possibilita uma rápida prototipação.

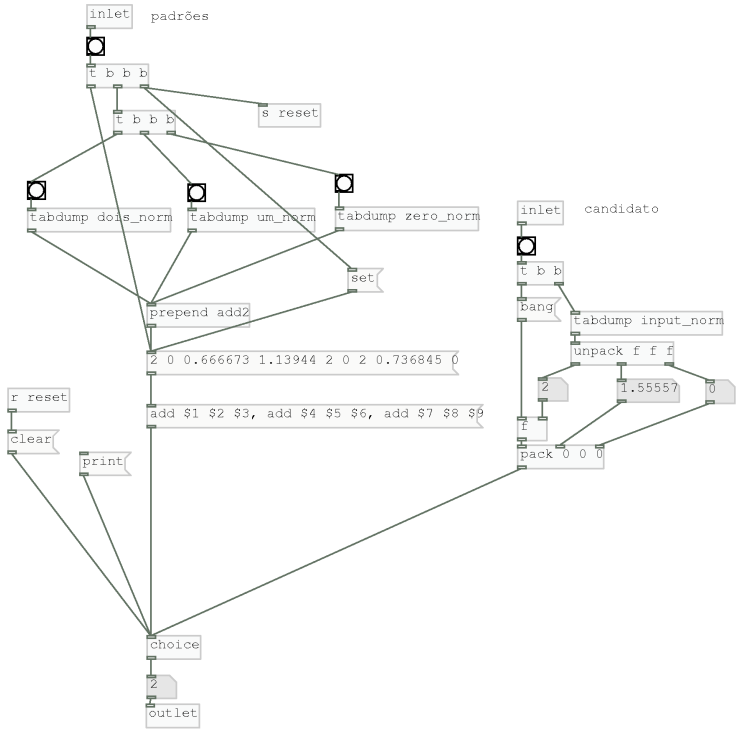

Fig. 6 – Operação de cálculo de similaridade entre contornos.

PERMEABILIDADE MELÓDICA

O conceito de permeabilidade melódica é aqui pensado como o índice de rigidez compositiva frente a um conjunto de alturas. Esse conceito foi introduzido por Ligeti :

> A perda da sensibilidade perante os intervalos conduz a um estado que poderíamos chamar de permeabilidade. Isso significa que estruturas de diferentes qualidades que transcorrem simultaneamente podem interpenetrar-se e mesmo dissolver-se completamente mudando apenas as relações de densidade horizontal e vertical, sendo indiferente, em princípio, quais intervalos se cruzam em detalhe [...] Embora a permeabilidade não tenha tido, até o momento, nenhuma influência decisiva sobre a forma, não era desconhecida nos estilos musicais antigos. Quem teve o grau mais baixo de permeabilidade até agora talvez tenha sido Palestrina, em cuja música vozes simultâneas, reguladas por leis expressas univocamente, enrolavam-se umas na outras. (Ligeti, 1958)

O objeto [sinc-permeabilidade] apresenta um primeiro estudo sobre o cálculo da permeabilidade melódica aplicada em composição interativa. A idéia é verificar a presença de um valor de classe de nota em uma lista de valores dinâmicos. Na figura 7 podemos ver um

[inlet classe de nota] que envia o mesmo valor para o array [$0-nota] e para a abstração [colecao_vs_pitch]. O array [$0-nota] coleciona as últimas seis classes de notas e envia uma lista com todos os valores através da variável [send lista_pitch].

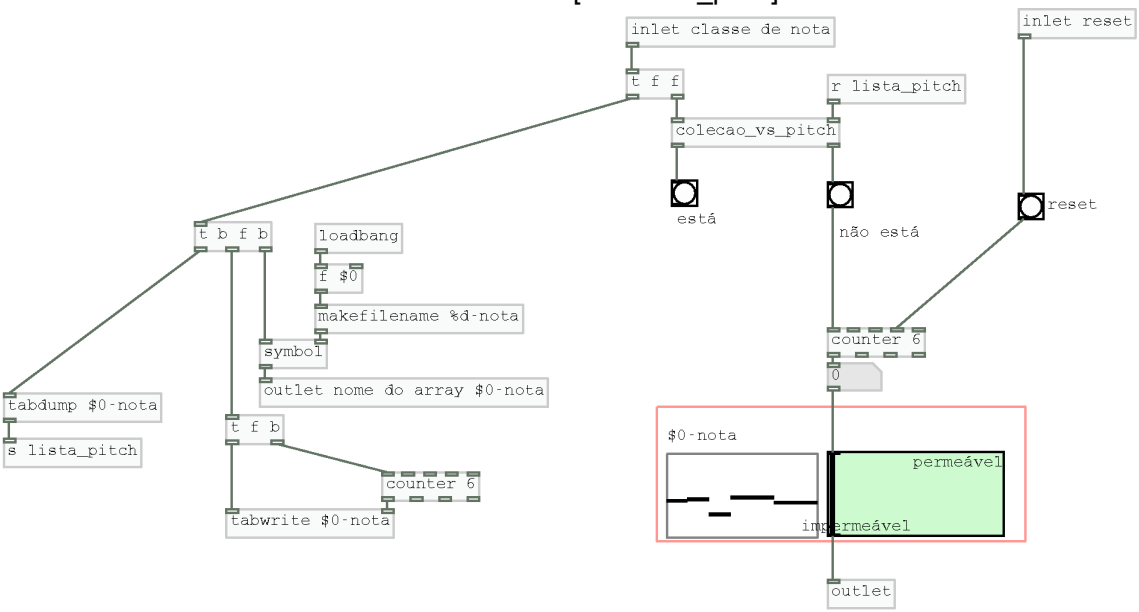

Fig. 7 - Patch que calcula índice de permeabilidade

Na figura 8 podemos ver a estrutura da abstração [colecao_vs_pitch] que verifica a ocorrência da atual classe de notas dentro das últimas seis classes passadas. O resultado é enviado para o contador [counter] e quanto mais notas “novas” surgem, maior é o índice de permeabilidade melódica.

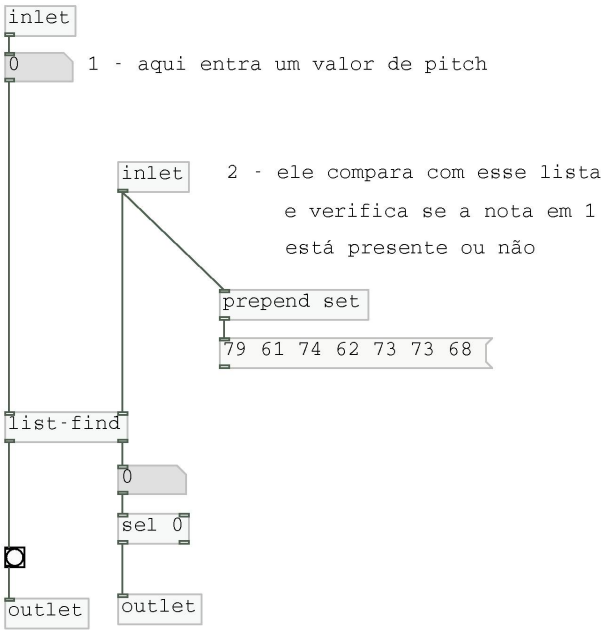

Fig. 8 – Abstração [colecao_vs_pitch].

Podemos pensar no conceito de permeabilidade como uma técnica composicional que pode ser quantificável, portanto passível de ser implementada em um programa de computador e agregada ao sistema que aqui se propõe. Esse conceito pode ser aplicável aos outros parâmetros do som e radicalizando esse pensamento podemos pensar no ato composicional como um controle dinâmico entre diversos níveis de permeabilidade em todos parâmetros do som.

ESTABILIDADE RÍTMICA

Uma das questões perseguidas é o fato do sistema ter conhecimento do atual nível de estabilidade rítmica. Isso pode ser alcançado por uma relação que considere variações de pulso e alternância e variações de padrões rítmicos de tamanhos diferentes. Um músico humano sempre realiza micro-variações de tempo e andamento (Rowe, 2004). Um dos desafios dessa pesquisa foi estabelecer uma forma do programa conseguir classificar a estabilidade de cada nova duração em relação as anteriores.

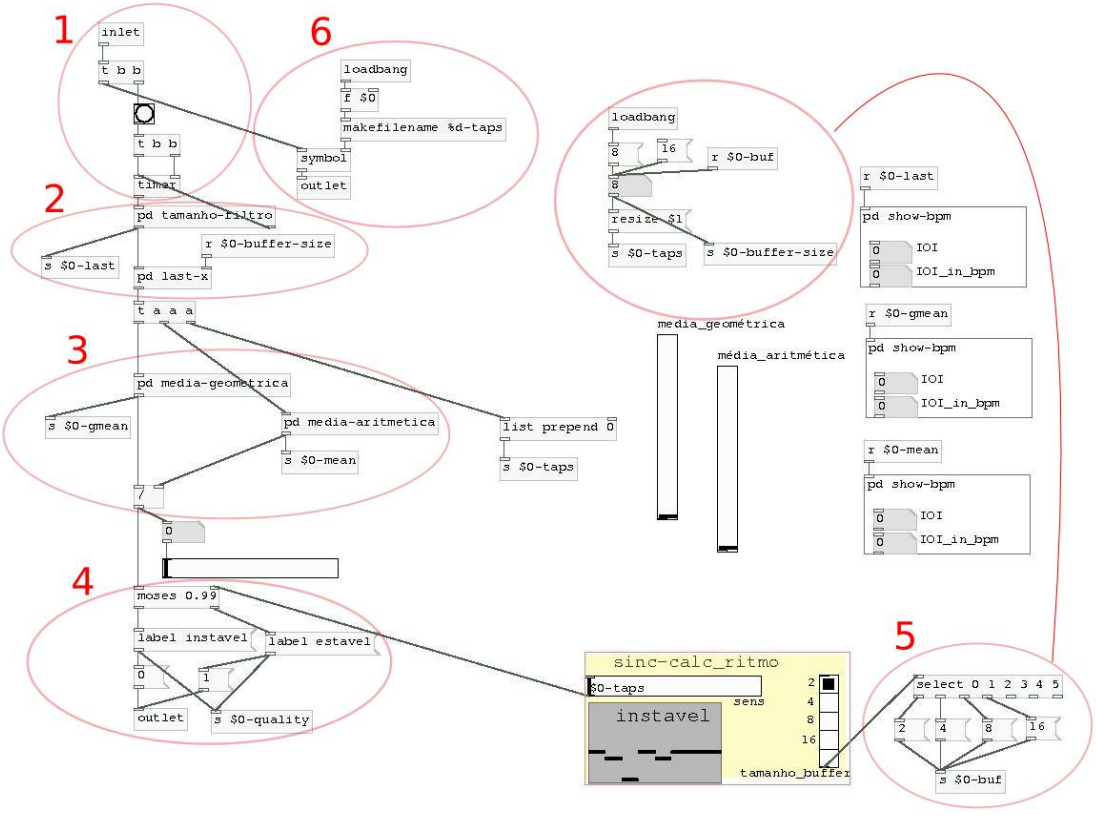

Fig. 9 – Patch de [sinc-calc-ritmo].

A técnica básica de análise rítmica em Pd, se dá através do uso do objeto [timer]. O objeto [timer] mede a distância em milisegundos entre "bangs" que chegam pela entrada da esquerda em relação ao que chega pela entrada da direita (WINKLER, 1993). Na abstração [sinc-calc-ritmo] é apresentado um método de classificação de estabilidade de durações entre notas. Primeiro são calculadas as durações entre cada ataque sonoro, esses valores são filtrado em [pd filter-range] e colocados em uma tabela dinâmica em [pd last-x]. O tamanho da tabela sempre pode ser redimensionado em tempo-real através da variável *$0-buffer-size* então é calculado ao mesmo tempo a média geométrica e a média aritmética e os resultados são executados na expressão definida por uma divisão da média geométrica pela média aritmética. Na seção 3 da figura 9 vemos em destaque a divisão das médias.

$$\frac{n\sqrt[n]{a1.a2....an}}{a1 + a2 + ...an}$$

Essa expressão é calculada a cada nota que chega, onde n é o tamanho do buffer. O resultado desse cálculo devolve um valor numa escala de 0 a 1. Esse valor representa o índice de instabilidade rítmica da última duração entre 2 notas, comparado com as últimas "n" durações. Nesse resultado dessa divisão é aplicado um filtro com [moses] onde se pode calibrar a sensibilidade do valor do índice final.

DENSIDADE RÍTMICA

Densidade rítmica é o índice que mede a quantidade de eventos sonoros dentro de espaços de tempo. Serve como ferramenta de análise e pode funcionar como um parâmetro estrutural em um sistema dedicado para análise musical.

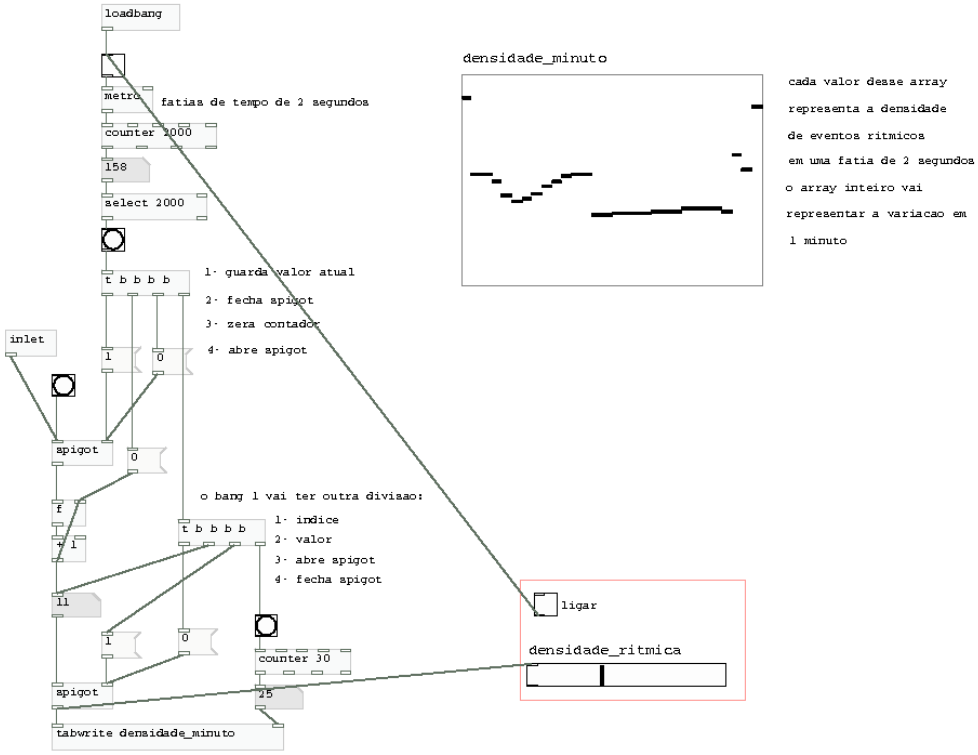

Fig. 10 - Patch de [sinc-densidade].

Na figura 10 vemos o conteúdo interno da abstração [sinc-densidade], onde são realizadas contagens de quantos ataques de eventos sonoros acontecem a cada 2 segundos de tempo. Densidade rítmica é um bom índice global de controle dos eventos sonoros por um instrumentista. A maneira de uso desse índice se transforma de um controle rítmico prático para um músico treinado.

Conclusão

As técnicas de análise aqui apresentadas representam bons métodos para criação de relações musicais interativas. As 5 abstrações mostradas são modulares e plenamente funcionais em um sinal de áudio monofônico. Algumas poucas alterações permitem que sejam usadas em projetos de computação gráfica e em combinação com outros projetos interativos.

Referências

Livros

ROWE, R. **Machine Musicianship.** Massachussets: MIT Press, 2004.

WINKLER, T. **Composing interactive music.** Massachussets: MIT Press, 1993.

Dissertações ou Teses

SAMPAIO, MARCOS DA SILVA. **Em torno da Romã: aplicações de operações de contornos na composição**. Dissertação de mestrado. Salvador: UFBA/ BA, 2008.

Artigos em Periódicos

LIGETI, G. "Transformações da Forma Musical", **Die Reihe nov/dez.** Tradução para o Português: Conrado Silva (1971), 1958.

PUCKETTE, M. "Patch for Guitar", **Proceedings PdCon 2007**. Montreal, 2007