Universidade do Estado do Rio de Janeiro
Centro de Tecnologia e Ciências
Faculdade de Engenharia

Heloísa Diná Félix Lima Bezerra

Hardware reconfigurável para geração de música harmonizada utilizando autômatos celulares

Rio de Janeiro
2013

Heloísa Diná Félix Lima Bezerra

Hardware reconfigurável para geração de música harmonizada utilizando autômatos celulares

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Orientadora: Prof^a. Dr^a. Nadia Nedjah
Co-orientadora: Prof^a. Dr^a. Luiza de Macedo Mourelle

Rio de Janeiro
2013

CATALOGAÇÃO NA FONTE
UERJ/REDE SIRIUS/CTC/B

B586 Bezerra, Heloísa Diná Félix Lima
Hardware reconfigurável para geração de música harmonizada utilizando autômatos celulares. – 2013.
170f. : il.

Orientadora: Nadia Nedjah.
Co-orientadora: Luiza de Macedo Mourelle.
Dissertação (Mestrado) – Universidade do Estado do Rio de Janeiro, Faculdade de Engenharia.
Bibliografia: f. 159 – 170.

1. Música. 2. Autômatos Celulares - Dissertação. 3. Reconfigurável. I. Nedjah, Nadia. II. Mourelle, Luiza de Macedo. III. Universidade do Estado do Rio de Janeiro. Faculdade de Engenharia. IV. Título.

CDU 510.5

Autorizo, apenas para fins acadêmicos e científicos, a reprodução total ou parcial desta dissertação, desde que citada a fonte.

Assinatura

Data

Heloísa Diná Felix Lima Bezerra

Hardware reconfigurável para geração de música harmonizada utilizando autômatos celulares

Dissertação apresentada, como requisito parcial para obtenção do título de Mestre, ao Programa de Pós-Graduação em Engenharia Eletrônica, da Universidade do Estado do Rio de Janeiro. Área de concentração: Sistemas Inteligentes e Automação.

Aprovado em 29 de Agosto de 2013

Banca Examinadora:

Prof^a. Dr^a. Nadia Nedjah (Orientadora)
Faculdade de Engenharia, UERJ

Prof^a. Dr^a. Luiza de Macedo Mourelle (Co-orientadora)
Faculdade de Engenharia, UERJ

Prof. Dr. Marcelo Soares Pimenta
Instituto de Informática, UFRGS

Prof^a. Dr^a Aura Conci
Instituto de Computação, UFF

Rio de Janeiro
2013

Agradecimentos

Agradeço a Deus, por me permitir alçar vôos cada vez mais altos.

Agradeço a Mamãe Ju, Papai Riba, Papai Madruga, Vovó Lalá e Tia Diná simplesmente por tudo.

Agradeço aos *Stormtroopers* Ana Luiza, Biel, Estanislau, Madureira, Natasha e Wajnsztajn e a todos os demais *Protetores da República* que sempre me apoiaram e incentivaram incondicionalmente durante esta jornada.

Agradeço as *Jedis* Adelene e Bianca, que tiveram a árdua tarefa de sequestrar meus *sabres de luz* e me manter longe do *lado obscuro da força*.

Agradeço a *Mestre* Dr^a. Nadia Nedjah (Orientadora) pela paciência e pelas broncas, ensinamentos e incentivos, sem os quais eu jamais teria zerado este nível.

Agradeço a *Mestre* Dr^a. Luiza de Macedo Mourelle (Co-orientadora) pelos incentivos, ensinamentos e pelas guloseimas que acalentaram esta difícil jornada.

Agradeço aos *Mestres* Dr. José Franco Machado do Amaral, Dr. Luiz Biondi Neto, Dr. Nival Nunes de Almeida e Dr. Pedro Henrique Gouvêa Coelho pelos maravilhosos ensinamentos durante esta jornada.

Agradeço aos *Jedis* Alexandre, Fábio, Luneque, Paulo e Rogério e aos *Padawans* Barbosa, Carlos, Carrara, Kleber, Leandro, Mathias, Nicola, Rafael e Rodrigo pelas brincadeiras, pizzas, biscoitos, piadas e pelo delicioso café que nunca provei, *rs*.

Agradeço ao *Padawan* Gabriel Cutrim, a quem espero um dia retribuir o grande auxílio.

Agradeço aos *Mestres* Dr^a Aura Conci e Dr. Marcelo Soares Pimenta, pela dedicação e pelas valiosas observações que corroboraram para a conclusão deste trabalho.

"...Ô minha madrinha..."
"...O que que é Lalá..."
— Tia Diná*(in memoriam)*

RESUMO

BEZERRA, Heloísa Diná Felix Lima. **Hardware reconfigurável para geração de música harmonizada utilizando autômatos celulares**. 2013. 170f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2013.

Desde os primórdios da humanidade, a descoberta do método de processamento cerebral do som, e consequentemente da música, fazem parte do imaginário humano. Portanto, as pesquisas relacionadas a este processo constituem um dos mais vastos campos de estudos das áreas de ciências. Dentre as inúmeras tentativas para compreensão do processamento biológico do som, o ser humano inventou o processo automático de composição musical, com o intuito de aferir a possibilidade da realização de composições musicais de qualidade sem a imposição sentimental, ou seja, apenas com a utilização das definições e estruturas de música existentes. Este procedimento automático de composição musical, também denominado música aleatória ou música do acaso, tem sido vastamente explorado ao longo dos séculos, já tendo sido utilizado por alguns dos grandes nomes do cenário musical, como por exemplo, *Mozart*. Os avanços nas áreas de engenharia e computação permitiram a evolução dos métodos utilizados para composição de música aleatória, tornando a aplicação de autômatos celulares uma alternativa viável para determinação da sequência de execução de notas musicais e outros itens utilizados durante a composição deste tipo de música. Esta dissertação propõe uma arquitetura para geração de música harmonizada a partir de intervalos melódicos determinados por autômatos celulares, implementada em *hardware* reconfigurável do tipo FPGA. A arquitetura proposta possui quatro tipos de autômatos celulares, desenvolvidos através dos modelos de vizinhança unidimensional de *Wolfram*, vizinhança bidimensional de *Neumann*, vizinhança bidimensional *Moore* e vizinhança tridimensional de *Neumann*, que podem ser combinados de 16 formas diferentes para geração de melodias. Os resultados do processamento realizado pela arquitetura proposta são melodias no formato *.mid*, compostas através da utilização de dois autômatos celulares, um para escolha das notas e outro para escolha dos instrumentos a serem emulados, de acordo com o protocolo MIDI. Para tal esta arquitetura é formada por três unidades principais, a unidade divisor de frequência, que é responsável pelo sincronismo das tarefas executadas pela arquitetura, a unidade de conjunto de autômatos celulares, que é responsável pelo controle e habilitação dos autômatos celulares, e a unidade máquina MIDI, que é responsável por organizar os resultados de cada iteração corrente dos autômatos celulares e convertê-los conforme a estrutura do protocolo MIDI, gerando-se assim o produto musical. A arquitetura proposta é parametrizável, de modo que a configuração dos dados que influenciam no produto musical gerado, como por exemplo, a definição dos conjuntos de regras para os autômatos celulares habilitados, fica a cargo do usuário, não havendo então limites para as combinações possíveis a serem realizadas na arquitetura. Para validação da funcionalidade e aplicabilidade da arquitetura proposta, alguns dos resultados obtidos foram apresentados e detalhados através do uso de técnicas de obtenção de informação musical.

Palavras-chave: Autômatos Celulares. Composição Musical. FPGA. MIDI. Musica Aleatória. VHDL.

ABSTRACT

BEZERRA, Heloísa Diná Felix Lima. **Reconfigurable hardware for music composition using cellular automata**. 2013. 170f. Dissertação (Mestrado em Engenharia Eletrônica) – Faculdade de Engenharia, Universidade do Estado do Rio de Janeiro, Rio de Janeiro, 2013.

Since the beginnings of mankind, questions about the method of sound processing in the brain, and hence the music, are part of the human imagination. Therefore, the researches related to this process constitute one of the largest fields of research in science. Among many attempts to understand the biological processing of sound, humans beings invented the automatic musical composition automatic process in order to verify the possibility of producing quality musical compositions, without the human interference, wich means, only by using the definitions and structures of existing pieces of music. This automatic music composition procedure, also known as random music, has been widely exploited over the past centuries, being used by greats musicians like *Mozart*. The advances in engineering and computing allowed the evolution of random music composition methods, making the application of cellular automata a viable alternative to determine the execution sequence of musical notes, as well as, other items used in this type of composition. This dissertation proposes a hardware architecture for random music composition using cellular automata to determine the melodic intervals order. A prototype has been implemented in FPGA. The proposed hardware architecture has four kind of cellular automata, modeled according to the one-dimensional *Wolfram* neighborhood, two-dimensional *Neumann* neighborhood, two-dimensional *Moore* neighborhood and three-dimensional *Neumann* neighborhood. These cellular automata can be combined in sixteen different ways for generating melodies. The processing result as conducted by the proposed architecture are melodies in the *.mid* format, through the use of two cellular automata, one for selecting notes and the others for choosing the instruments to be emulated, in accordance with the MIDI protocol. This hardware is composed by three main units, the frequency divider unit, which is responsible for the synchronizing of the tasks performed by the hardware architecture, the cellular automata set unit, which is responsible for the control and enabling of the cellular automata and the MIDI machine, which is responsible for organizing the iterations of the cellular automata and convert them to the MIDI protocol structure, generating the final musical composition. The proposed hardware architecture is parametrizable, so that the data settings that influence the final product, such as, the rules of the cellular automata, are informed by the user. There are no limits regarding the possible combinations to be performed in the hardware architecture. In order to validate the functionality and applicability of the proposed hardware, some results were presented and detailed through the use of techniques for musical information retrieval.

Keywords: Cellular Automata. FPGA. MIDI. Musical Composition. Random Music. VHDL.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

r	raio de vizinhança
bpm	batidas por minuto
CA	Autômato Celular
CAM	Cellular Automata Machine
CBM	Cellular Automata Machine Brain
ChaOs	Chemical Oscillator Celular Automatom
FPGA	Field Programmable Gate Arrays
Gb	GigaByte
GHCA	Greenberg-Hastings Cellular Automata
HDL	Hardware Description Language
Hz	Hertz
IHM	Interface Homem Máquina
jMusic	Java Music
LFSR	Linear Feedback Shift Register
LGCA	Lattice Gas Cellular Automata
LUT	Lookup Table
Mb	Megabytes
MHz	Megahertz
MIDI	Musical Interface Digital Instruments
MLS	Maximum Length Sequence
MP3	MPEG Layer 3
MPEG	Moving Picture Experts Group
PPQ	Pulses Per Quarter Note
RAM	Random Access Memory
ROM	Ready Only Memory
SMF	Standard MIDI Files
VHDL	Very High Speed Integrated Circuits Hardware Description Language
WAV	Waveform Audio Format

SUMÁRIO

Introdução

GRANDE parte da expressão humana no decorrer da vida é caracterizada por sons. Desde o momento do nascimento, cujo choro indica o correto funcionamento dos sistemas nervoso, respiratório e muscular (BRANCO et al., 2006) até o momento de falecimento, com o último suspiro, os seres humanos costumam esboçar suas ações e reações através do uso de expressões sonoras, como por exemplo, a fala, o choro, o grito e a música.

Esta forte identificação entre o ser humano e a expressão sonora atravessou os séculos, adquirindo com o passar destes definições e classificações, realizadas na tentativa de categorizar as sensações provocadas no corpo humano pelo processamento auditivo de um som (PEREIRA, 1997; IAZZETTA, 1997; PINTO, 2001). Esta busca por respostas acerca do funcionamento do sistema auditivo humano gerou resultados sobre o funcionamento do cérebro humano. Algumas das evidências destes resultados são os fatos de que (ANDRADE, 2004; HATEM; LIRA; MATTOS, 2006; BONINI-ROCHA et al., 2009; NOCITI, 2010):

- Bebês tem capacidade de processamento musical similar ao de adultos, respondendo melhor as escalas com distribuição desigual de tons, a melodias e acordes mais consonantes e também a estímulos rítmicos métricos, ou seja, baseados em pulsações regulares;
- Há indícios de que todos nascem com o chamado *ouvido absoluto*, que é a capacidade de identificar a altura exata das notas musicais, e com o decorrer do tempo, em função da falta do estímulo necessário, esta habilidade é perdida;
- Alguns componentes estruturais da música, como contorno melódico e intervalos são codificados automaticamente pelo cérebro humano;
- Estudos neurofisiológicos comprovaram que os princípios básicos de harmonia, consonância e dissonância são reflexos das propriedades anatômicas e fisiológicas dos circuitos auditivos humanos;

- Comparativos interculturais indicam que a prevalência de intervalos de *oitavas* e *quinta justa* em culturas muito diferentes podem ser resultado da forma como os ouvidos destas populações foram desenvolvidos;

- Existe uma degeneração auditiva específica, denominada *amusia*, que se refere a incapacidade de reconhecer melodias anteriormente consideradas familiares. Este tipo de degeneração pode ser adquirida ou congênita. Em contrapartida, existem patias ligadas diretamente a estimulação musical específica, como por exemplo a *epilepsia musicogênica*, na qual os ataques são incitados pela audição de determinadas peças musicais;

- Alguns portadores de anomalias cerebrais congênitas apesar de se tornarem, em sua fase adulta, completamente dependentes de auxílio na realização de tarefas, conseguem apresentar grande fluência linguística e musical, como é o caso dos portadores da *síndrome de Williams*;

- Existem, de forma idêntica ao senso espacial e motriz, redes neuronais essenciais específicas para processamento musical;

- Resultados de eletroencefalografia comprovam a dominância do hemisfério cerebral direito para a percepção melódica e o envolvimento de ambos os hemisférios cerebrais para o processamento local e analítico dos componentes musicais.

- Estudos da área de psicopedagogia comprovam a importância da música para o desenvolvimento das inteligências intelectual e social dos indivíduos. (CHIARELLI; BARRETO, 2005; ILARI, 2006)

Descobriu-se ainda que ao contrário da crença cultural popular, as respostas emocionais à música são relativamente homogêneas e compartilhadas por todos os membros de uma mesma cultura (ANDRADE, 2004). Também comprovou-se, parcialmente, que a forma de organização musical, sucessão melódica e variação, proporcionam ao ouvinte, independente das diferenças culturais, uma correlação imediata de itens subjetivos, como por exemplo tristeza e alegria, através da ativação de mecanismos neurais nas *regiões paralímpicas*, normalmente associadas as sensações de prazer e desprazer (ANDRADE, 2004).

E, dada esta importância da expressividade sonora no desenvolvimento da humanidade, há mais de 2 mil anos, a música, por ser uma expressão sonora universal, é utilizada e pesquisada como uma ferramenta auxiliar para resolução de diversos problemas humanos, nas mais

diferentes áreas, extrapolando a compreensão tradicional de seu uso apenas para divertimento e distração (ANDRADE, 2004; HATEM; LIRA; MATTOS, 2006; NOCITI, 2010).

Para fins de exemplificação da diversidade de aplicabilidade da música, vários estudos já comprovaram a eficácia do uso da música para efeito terapêutico em pós operatórios, como o estudo realizado por *Hatem et al.*, (HATEM; LIRA; MATTOS, 2006), que evidenciou a eficácia da aplicação de música como ferramenta auxiliar para o controle da dor e ansiedade, em crianças na faixa etária de 1 a 16 anos, durante o período pós operatório de cirurgias cardíacas, através da realização de sessões de musicoterapia com duração de 30 minutos. Também já foram comprovados os efeitos da musicoterapia para redução da demanda por administração de ansiolíticos em pacientes no período perioperatório, combatendo os efeitos colaterais provocados por este tipo de medicamento, como por exemplo o esquecimento das instruções realizadas antes da alta hospitalar (BORCHGREVINK, 1991; COSTA, 1992; NOCITI, 2010).

O objetivo deste trabalho foi o desenvolvimento de uma arquitetura para geração de música harmonizada através da combinação pseudo aleatória de intervalos melódicos, de forma a proporcionar uma ferramenta alternativa para a realização de estudos e pesquisas relacionados a área de música que demandem a utilização de música algorítmica. A pseudo aleatoriedade pretendida para organização dos intervalos na melodia foi obtida através da implementação de autômatos celulares em *hardware*. Para tal, desenvolveu-se este projeto em *hardware* reconfigurável do tipo FPGA (*Field Programmable Gate Arrays*) através de sua descrição em linguagem *VHDL*.

A escolha pela implementação de autômatos celulares para geração dos parâmetros musicais deveu-se a aplicabilidade da utilização destes para simulação de dinâmicas complexas, o que proporciona uma enorme gama de possibilidades para geração de produtos musicais, conforme evidenciado em alguns casos de sucesso na utilização de autômatos celulares para produção musical (MIRANDA, 2001; REINERS, 2004). Já a decisão pela implementação em *hardware* deveu-se a previsão da aplicação futura da arquitetura proposta em projetos que demandem por processamento em tempo real, como por exemplo algumas tarefas de entretenimento e aplicações nas áreas de medicina, como por exemplo shows e investigações do processamento musical por área cerebral, (ANDRADE, 2004; BIGAND, 2005), ao pequeno número de aplicações com autômatos celulares em *hardware* e a abertura para que ocorram futuras implementações da arquitetura proposta acoplada a equipamentos diversos (IAZZETTA, 2003; WANDERLEY, 2006; CORRÊA et al., 2008). De forma a proporcionar uma maior aplicabilidade da arquitetura proposta, os produtos musicais resultantes do processamento da

arquitetura são confeccionados conforme formatação definida pelo protocolo MIDI, que prevê a utilização de todas as notas e instrumentos musicais conhecidos internacionalmente, além de facilitar o armazenamento das melodias obtidas (STANDARD, 1999). Ainda com o intuito de maximizar o potencial de aplicação da arquitetura desenvolvida, foram definidos alguns parâmetros a serem configurados pelo usuário, como por exemplo a configuração inicial dos autômatos celulares e a fórmula do compasso.

O funcionamento da arquitetura proposta divide-se em três etapas. A primeira etapa é a de configuração, na qual o usuário define os parâmetros que deseja utilizar para geração de melodias. Na segunda etapa a unidade conjunto de autômatos celulares efetua o processamento das configurações realizadas pelo usuário gerando os dados necessários a composição musical. Na terceira etapa a unidade MIDI efetua a conversão dos dados, enviados pela unidade conjunto de autômatos celulares, nos códigos utilizados pelo protocolo MIDI, confeccionando ao final de seu processamento um arquivo *.txt*, que poderá ser convertido para a extensão *.mid*, através de um conversor hexadecimal. Após esta conversão o arquivo *.mid* poderá ser executado por qualquer interpretador MIDI.

De forma a proporcionar uma melhor compreensão das atividades desenvolvidas, esta dissertação está organizada em seis capítulos, além desta Introdução. Primeiramente, o Capítulo 1 aborda de forma sucinta a teoria dos autômatos celulares visando fornecer o embasamento teórico necessário para compreensão da aplicabilidade e importância destes para os estudos realizados no decorrer desta dissertação. Seguindo-se, o Capítulo 2 apresenta as definições básicas e terminologia de som e música, bem como as características do protocolo MIDI, utilizado nesta dissertação como linguagem de codificação da composição musical. Na sequência, o Capítulo 3 descreve algumas aplicações conexas com as motivações e objetivos deste trabalho, de forma a apresentar um panorama do desenvolvimento, norteando os estudos realizados. Subsequentemente, o Capítulo 4 apresenta a arquitetura proposta nesta dissertação, pontuando seus detalhes através de exemplificações de seu funcionamento. Depois, o Capítulo 5 trata dos resultados obtidos na síntese e na simulação da arquitetura proposta. Finalmente, esta dissertação encerra-se com as conclusões e as propostas de trabalhos futuros apresentados no Capítulo 6.

Capítulo 1

AUTÔMATOS CELULARES

NESTE capítulo, serão apresentados alguns conceitos da teoria de autômatos celulares, e, as definições e convenções que nortearam o desenvolvimento das atividades no decorrer de elaboração desta dissertação. Para tal, o capítulo é dividido estruturalmente em quatro partes. Na Seção 1.1 é apresentada uma breve ponderação histórica acerca do desenvolvimento da teoria dos autômatos celulares. Na Seção 1.2 é apresentada uma visão geral acerca da teoria de autômatos celulares. Na Seção 1.3 são apresentadas as definições básicas e terminologia dos autômatos celulares. Na Seção 1.4 são comentados e definidos alguns dos tipos de autômatos celulares existentes, através de exemplificação dos modelos de vizinhança.

1.1 Breve Histórico

Em 1936, o matemático *Alan Turing* desenvolveu um dispositivo teórico denominado máquina universal ou máquina de *Turing* (TURING, 1936, 1937; SETTI, 2009; SILVA, 2010), que consistia em um modelo matemático, baseado em autômatos finitos, composto por um grupo finito de símbolos, um número finito de estados e uma fita de comprimento ilimitado dividida em pequenas células organizadas em sequência. A máquina de *Turing* foi um artefato tão grandioso do ponto de vista da evolução da ciência, que, mesmo sendo apenas teórica é considerada como um dos fundamentos da Ciência de Computação e a base da teoria dos autômatos celulares e do conceito de autômato celular unidimensional. Além disto, até hoje, a aplicação de sua ideia de fita infinita é utilizada para nortear a resolução de problemas de borda para autômatos celulares a *n dimensões* (BAH; CHEN; CREUTZ, 1939; DILãO, 1993; CASTRO; CASTRO, 2008; GREMONINI; VICENTINI, 2008; HODGES, 1999).

Ao final da década de 40, com o surgimento de novas técnicas computacionais, tornou-se possível a realização de algumas modelagens matemáticas de sistemas complexos, até então, consideradas impossíveis, tais como a análise de trajetórias táticas para movimentação de

tropas em exercício e cálculos balísticos (FILHO, 2007). Tais modelagens permitiram aos pesquisadores da época explorar, elaborar, complementar e simular determinados comportamentos previstos em teoria (PASCOAL, 2005).

Em 1950 *John von Neumann*, matemático e pesquisador atuante na área de neurofisiologia, propôs um modelo matemático abstrato que simulava as rotinas de um robô com potencial de replicação a partir de um conjunto de peças separadas, denominado *Universal Copier and Constructor*, como solução para um problema de auto-reprodução. Este modelo foi criado a partir de uma sugestão feita por *Stanislau Ulam* a *von Neumann*. Esta sugestão foi a da utilização de um gráfico infinito de pontos, onde cada ponto possui um número finito de conexões com os pontos próximos, denominados vizinhança, e uma determinada correlação de estados finitos, criando-se assim o primeiro autômato celular bidimensional (BEYER; SELLERS; WATERMAN, 1985; LIMA, 2007; MELOTTI, 2009; BAIANU, 2010).

Na década de 70 *John Conway*, matemático e pesquisador da área de grupos finitos, através de seu jogo *Game of Life*, com seu modelo bidimensional de dois estados, popularizou os autômatos celulares bidimensionais. Na aplicação de *Conway*, um autômato celular bidimensional é utilizado para emular a evolução de um grupo de organismos vivos durante um determinado período de tempo através dos parâmetros de vida ou morte (CONWAY, 1970).

Na década de 90 e durante o início dos anos 2000, *Stephen Wolfram* definiu sistematicamente os autômatos celulares unidimensionais através de mecanismos determinísticos simples, pontuando os detalhes de cada uma das 256 regras que analisou, corroborando com a caracterização de diversos conceitos que fundamentam a teoria de autômatos celulares (WOLFRAM, 1994, 2002, 2006).

1.2 Visão Geral

Um Autômato Celular, *Celular Automata - CA*, é um modelo matemático discreto capaz de simular comportamentos complexos, a partir de regras simples, baseado na teoria descrita por *Turing* (TURING, 1950; MARIA, 2003). Em termos abstratos, um autômato celular pode ser definido como um sistema capaz de reproduzir comportamentos padronizados efetuados por um grupo homogêneo de organismos em contínua interação, de maneira que um observador consiga perceber, mesmo que minimamente, a existência de alguns dos seguintes conceitos fundamentais:

- Célula: Menor unidade do sistema capaz de realizar processamentos a partir de características predefinidas, ou seja, um autômato;

- *Grid* (reticulado ou *lattice*): É o agrupamento formado por células idênticas, com capacidade de interação predeterminada, ordenadas de forma estrutural, ou seja, um autômato celular;
- Vizinhança: É o conjunto de células do *grid*, capaz de provocar alterações comportamentais de forma unívoca em outra célula;
- Estado: É a demonstração do comportamento da célula, ou seja, a sua reação as atividades de sua vizinhança;
- Regra: É o conjunto de informações que determina o formato de evolução de cada célula com base na análise de sua vizinhança.

Todos estes cinco conceitos supracitados são interdependentes e serão ampliados e complementados no decorrer das próximas seções deste capítulo.

1.3 Definições Básicas e Terminologia

A comunicação entre as células vizinhas no *grid* acontece de forma uniforme, determinística e síncrona produzindo a evolução do sistema ao longo do tempo. O formato de ligação entre as células no *grid* irá definir a chamada *dimensão* do mesmo, considerando como base de aproximação visual, os sistemas de coordenadas conhecidos e o seu referenciamento na forma matricial.

Matematicamente, um autômato celular $\mathbb{A}$, com estrutura de *grid* em *n dimensões*, composto por um número finito de células é definido por (MELOTTI, 2009; OLIVEIRA, 2009; CARDOZO, 2006; VIEIRA JOãO PAULO SCHWARZ SCHüLER, 2012) :

- Um conjunto finito de estados, denominado $\mathbf{Q}$;
- Um conjunto finito de símbolos, denominado *alfabeto*, simbolizado por Σ;
- Um estado inicial, q_0, anterior a qualquer entrada do sistema;
- Um subconjunto de estados de $\mathbf{Q}$, denominado $\mathbf{F}$, cujos elementos são chamados estados finais;
- Uma estrutura $\mathcal{G}$, denominada *grid*, na qual estarão distribuídas ordenadamente todas as células de $\mathbb{A}$, cada qual interligada a vizinhança que lhe é pertinente.

Para estas definições das características de autômato celular sempre existirão as seguintes relações:

1. Toda célula do *grid* possuirá um q_0;
2. $\forall\ q_0$, q_0 é VERDADEIRO $\Leftrightarrow q_0 \subset \mathbf{Q}$;
3. $\mathbf{Q}$ será constituído por no mínimo dois elementos, e um destes será escolhido como q_0, como por exemplo: Se $\mathbf{Q} = \{q_1, q_2\}$ então $(q_0 = q_1) \vee (q_0 = q_2)$;
4. No decorrer de sua evolução, $\mathbb{A}$ sempre converge para $\mathbf{F}$;
5. Toda e qualquer mudança de estados é definida por uma função de transição de estados, δ, que será:
   - Para autômatos determinísticos: δ: $\mathbf{Q} \times \Sigma \rightarrow \mathbf{Q}$, onde δ é uma função total, para a qual existe somente uma resposta possível a uma dada combinação de entradas;
   - Para autômatos não-determinísticos: $\delta : \mathbf{Q} \times (\Sigma \cup \{\epsilon\}) \rightarrow \mathcal{P}(\mathbf{Q})$, na qual δ é uma função parcial, e a resposta estará contida em um conjunto de respostas possíveis para uma dada combinação de entradas;
6. Toda célula de $\mathbb{A}$ possuirá um conjunto de vizinhança, denominado $\mathbf{V}$, formado por algumas das células contidas no $\mathcal{G}$ de $\mathbb{A}$, que irá influenciar diretamente na evolução desta célula, fornecendo os dados de entrada para a função de transição δ;
7. Todas as células contidas em $\mathcal{G}$ são idênticas em constituição e funcionamento, existindo, para cada célula, um determinado grupo de elementos que constituem seu conjunto $\mathbf{V}$.

A partir destas definições torna-se possível a formalização dos modelos de classe, regra e dimensão que serão apresentados nas próximas seções deste capítulo e são inerentes ao desenvolvimento de qualquer CA .

1.3.1 Classes

Como resultado de sua pesquisa, *Stephen Wolfram* definiu a existência de quatro modelos diferentes de autômatos celulares a partir da análise do conjunto de estados iniciais, do modelo de regras e da evolução do sistema. Estes modelos são classificados em um padrão chamado *classe*, que é estruturado em quatro níveis (WOLFRAM, 2002).

1.3.1.1 Classe-1

Nesta *classe* enquadram-se os autômatos celulares mais simples. A evolução dos CAs pertencentes a esta *classe* é caracterizada pelo processo de *looping*. Neste caso, considerando o período de tempo necessário para a ocorrência de cada iteração do sistema, um observador que presencie a evolução do sistema ao longo de um período de tempo n, no qual n seja suficiente grande de forma a corresponder no mínimo a uma evolução completa do sistema, será capaz de:

1. Identificar de forma parcial ou total os elementos do conjunto $\mathbf{Q}$ e identificar o momento de *looping*;

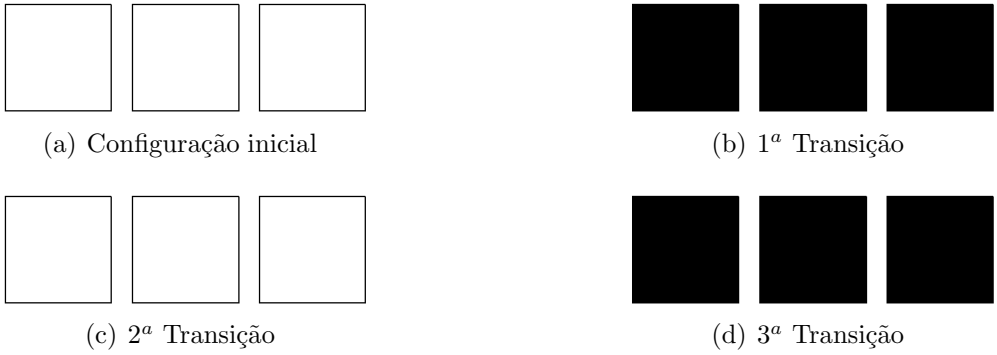
2. Predizer qual será o próximo elemento de $\mathbf{Q}$ a ser assumido pelas células do sistema.

Neste caso, todas estas respostas dependerão do conjunto $\mathbf{Q}$, pois, a análise de vizinhança não possuirá grande influência sobre o sistema. É importante observar que, como todas as células evoluem para os mesmos estados ao longo do tempo, caracterizando um comportamento homogêneo, este tipo de CA não transmite ao observador qualquer impressão de aleatoriedade, limitando a aplicabilidade desta classe. Para fins de exemplificação, dito que $\mathbb{A}$ é um autômato celular de Classe-1, com 3 células, $\{\mathbb{A}_1, \mathbb{A}_2, \mathbb{A}_3\}$, se o seu conjunto $\mathbf{Q}$ for igual a $\{q_1,\ q_2\}$, apresentados na Figura 1.

(a) q_1 (b) q_2

Figura 1: Conjunto de estados possíveis para as células de $\mathbb{A}$

A representação gráfica de sua evolução ao longo do tempo será a apresentada na Figura 2. Para este CA exemplo, em especial, considerando que $\mathbb{A}$ seja inicializado no estado q_1, e que todas as suas 3 células tenham o mesmo estado inicial, conforme é representado na Figura 2(a), como o conjunto $\mathbf{Q}$ é composto por apenas 2 valores possíveis, representados na Figura 1, o processo chamado de *looping* inicia-se após a segunda iteração. Na primeira iteração todas as células evoluirão do estado q_1 para o estado q_2, conforme Figura 2(c), retornando na segunda iteração para o estado q_1, conforme Figura 2(d), e assim sucessivamente até que ocorra o término do número de iterações predefinido ou algum tipo de interferência externa, parando o sistema.



(a) Configuração inicial (b) 1^a Transição

(c) 2^a Transição (d) 3^a Transição

Figura 2: Configuração inicial e primeiras transições de $\mathbb{A}$

1.3.1.2 Classe-2

Os CAs pertencentes a Classe-2 representam uma pequena evolução em relação aos da Classe-1. Nesta *classe*, o elemento escolhido para representar o estado inicial, q_0, de cada célula é independente, dentro dos possíveis valores de $\mathbf{Q}$, diferentemente da Classe-1. Contudo, após várias iterações este tipo de sistema tende a estabilidade comportamental, culminando, da mesma forma que a *classe* anterior, em um processo de *looping*. A Classe-2 ainda é marcada, em alguns casos, por não necessitar da análise da relação de vizinhança para correta compreensão do comportamento do sistema ao longo do tempo.

Para fins de exemplificação, se um CA $\mathbb{B}$, Classe-2, possuir 3 células, $\{\mathbb{B}_1, \mathbb{B}_2, \mathbb{B}_3\}$, contemplando como elementos possíveis do conjunto $\mathbf{Q}$ a quíntupla $\{q_1, q_2, q_3, q_4, q_5\}$ e considerando que a única regra presente na função δ é a sequência de transição de estados, que ocorre conforme apresentado na Figura 3.

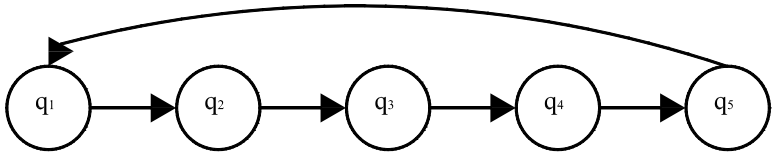


Figura 3: Sequência de transição de estados

Se este conjunto de estados possíveis for representado graficamente, de forma que cada estado possua um padrão gráfico próprio, neste caso as cores branco, preto, cinza, azul e verde respectivamente para representação dos estados $\{q_1, q_2, q_3, q_4, q_5\}$, conforme representado na Figura 4.

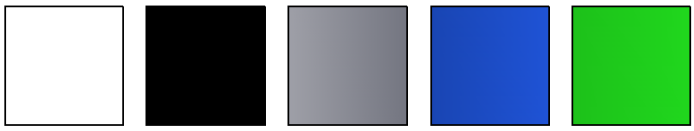
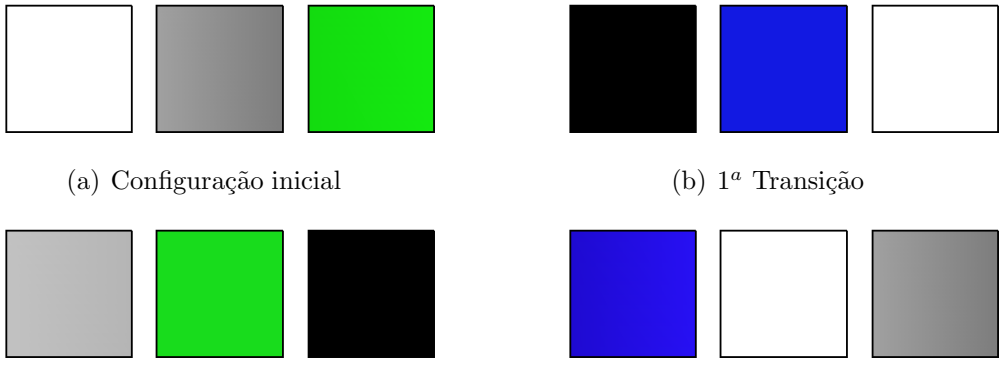


Figura 4: Conjunto de estados possíveis para as células de $\mathbb{B}$

Se q_1, q_3 e q_5, nesta ordem, forem escolhidos para inicialização de cada uma das células do CA em questão, as primeiras iterações do sistema resultam nas transições temporais apresentadas na Figura 5.



(a) Configuração inicial

(b) 1^a Transição

(c) 2^a Transição

(d) 3^a Transição

Figura 5: Configuração inicial e primeiras transições de $\mathbb{B}$

Ou seja, dada a função δ, um observador ao acompanhar a evolução do CA tende a considerar que este possui algum tipo de aleatoriedade em função do aumento na dificuldade de identificação das características do sistema, como por exemplo, os valores do conjunto $\mathbf{Q}$ e a previsão de quando começará o processo de *looping*. Este aumento de complexidade é baseado em dois fatores:

1. Número de células do CA;
2. A quantidade de elementos constituintes do conjunto $\mathbf{Q}$.

Identificando $f(n)$ como a função para o aumento de complexidade na classificação das características de um CA Classe-2, na qual n é o número de células do CA, a relação entre estes fatores será a apresentada na Equação 1:

$$f(n) = \mathbf{Q} \setminus n \tag{1}$$

Ou seja, para um CA Classe-2, quanto maior o número de estados e menor o número de células no *grid*, mais difícil será para um observador identificar as propriedades do sistema.

1.3.1.3 Classe-3 e Classe-4

Os CAs da Classe-3 introduzem a análise de *vizinhança* como um dos principais parâmetros para a transição de estados. Portanto, um CA de Classe-3 tem a sua evolução obrigatoriamente regida por dois fatores:

1. Sequência de transição de estados;
2. Análise dos estados atuais de cada um dos elementos pertencentes ao conjunto *vizinhança* da célula considerada.

Portanto, a composição do alfabeto Σ para esta *classe* será realizada através do uso dos conectivos lógicos SE e ENTÃO, como por exemplo, na lógica apresentada na Equação 2:

$$\delta = \begin{cases} Se\ vizinhos\ em\ estados\ iguais,\ Entao\ mudar\ o\ estado\ atual; \\ Se\ vizinhos\ em\ estados\ diferentes,\ Entao\ manter\ estado\ atual. \end{cases} \quad (2)$$

Consequentemente, o comportamento de um CA Classe-3 emula o comportamento de organismos em interação constante com o meio no qual estão inseridos. Para esta *classe*, a complexidade do sistema a ser projetado será nivelada em função do conjunto de proposições lógicas, que poderá evoluir de proposições simples, como por exemplo, {Se Sim X..., Se Não Y...} para proposições complexas, como por exemplo, {Se Sim X..., Se Não Y..., Mas caso ocorra Z..., Para Z = N e X = I...}.

Para fins de exemplificação do comportamento de um CA de Classe-3 será utilizado um CA genérico $\mathbb{C}$, cujo sistema obedecerá a configuração descrita na Tabela 1:

Tabela 1: Características de $\mathbb{C}$

Característica	**Utilizado**
Disposição de *grid*	Unidimensional
Número de Células	5
Configuração Inicial	Aleatória
Número de Estados Possíveis	2

Se a representação visual do conjunto **Q** para o CA $\mathbb{C}$ for a apresentada na Figura 6.

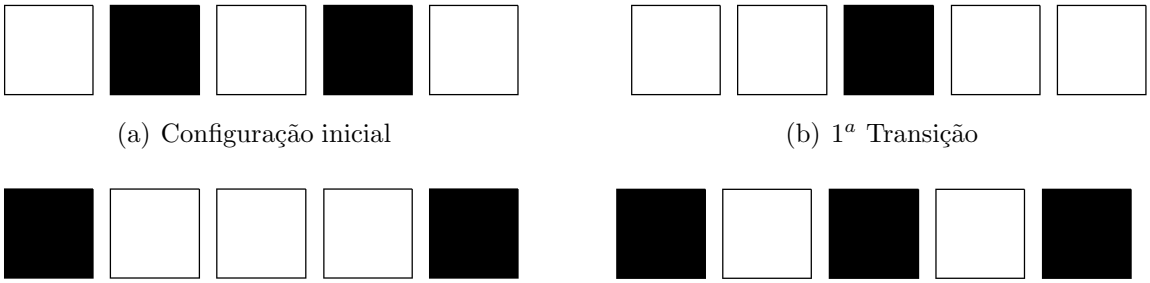
(a) q_1 (b) q_2

Figura 6: Conjunto de estados possíveis para as células de $\mathbb{C}$

Considerando para $\mathbb{C}$ a função de transição δ, descrita na Equação 3:

$$\delta = \begin{cases} Se\ vizinhos\ em\ estados\ iguais,\ Entao\ mudar\ o\ estado\ atual; \\ Se\ vizinhos\ em\ estados\ diferentes,\ Entao\ manter\ estado\ atual. \end{cases} \quad (3)$$

Dado que o conjunto *vizinhança* das células de $\mathbb{C}$ contém dois elementos, que são as duas células adjacentes a célula em análise, e considerando como condição de fronteira a referência circular, ou seja, a primeira célula de $\mathbb{C}$ é vizinha da última célula e vice-versa, se o estado inicial de $\mathbb{C}$ for o representado na Figura 7(a), as transições temporais resultantes da interação entre as células serão as apresentadas nas Figuras 7(b), 7(c) e 7(d).



(a) Configuração inicial (b) 1^a Transição

(c) 2^a Transição (d) 3^a Transição

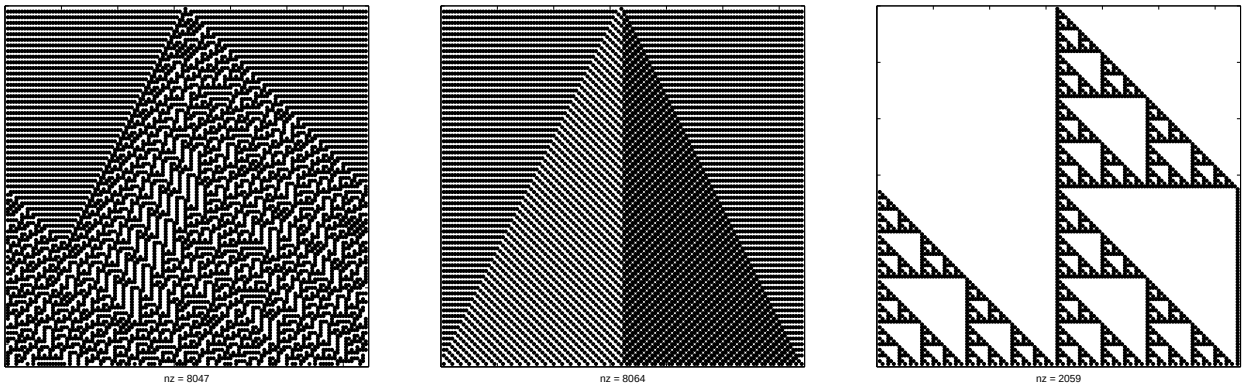
Figura 7: Configuração inicial e resultado das primeiras iterações de $\mathbb{C}$

Uma análise da primeira transição realizada pelo CA $\mathbb{C}$, com base na Equação 3, é apresentada na Tabela 2, onde as células são simbolizadas por $\mathbb{C}$ e a localização de cada célula no *grid* é representada pelo índice de $\mathbb{C}$. A evolução realizada pelo CA $\mathbb{C}$, que nas iterações apresentadas não entrou em *looping*, faz com que dificilmente um observador, sem o conhecimento prévio das proposições lógicas existentes para a função δ deste CA, do conjunto **Q** e da vizinhança de cada célula, consiga predizer a evolução realizada pelo CA ao longo do tempo. Por estes motivos, os CAs de Classe-3 são chamados de *caóticos* ou *irregulares*.

Tabela 2: Análise da 1^a transição de $\mathbb{C}$

Célula	Vizinhança	Estado atual	Estados vizinhança	Próximo estado
$\mathbb{C}_1$	$\mathbb{C}_5, \mathbb{C}_2$	q_1	q_1, q_2	q_1
$\mathbb{C}_2$	$\mathbb{C}_1, \mathbb{C}_3$	q_2	q_1, q_1	q_1
$\mathbb{C}_3$	$\mathbb{C}_2, \mathbb{C}_4$	q_1	q_2, q_2	q_2
$\mathbb{C}_4$	$\mathbb{C}_3, \mathbb{C}_5$	q_2	q_1, q_1	q_1
$\mathbb{C}_5$	$\mathbb{C}_4, \mathbb{C}_1$	q_1	q_2, q_1	q_1

Os autômatos celulares que se enquadram na Classe-4 possuem as mesmas características dos autômatos celulares da Classe-3, ou seja, possuem conjunto de regras formado por proposições lógicas que dependem diretamente da análise de *vizinhança*. Consequentemente, os CAs de Classe-4 podem ser considerados uma evolução dos métodos utilizados para composição dos CAs de Classe-3. A diferença entre estas duas *classes* é a complexidade geométrica que se forma no decorrer da evolução de um CA Classe-4. A Figura 8 apresenta alguns exemplos de padrões gráficos complexos. Estes exemplos são o espelho da evolução de um CA unidimensional, com *grid* composto por 128 células, através de três conjuntos de regras diferentes, mas com a mesma configuração inicial. Estas regras, que são as regras 45, 57 e 60 propostas por *Wolfram* (WOLFRAM, 1994), geram ao longo da evolução do CA padrões gráficos extremamente complexos, que são uma característica da evolução dos CA Classe-4 (SILVA; MARTINS, 2003; ASSIS, 2006; MULLER, 2008; SUHADOLNIK et al., 2012).



(a) Regra 45 (b) Regra 57 (c) Regra 60

Figura 8: Exemplos de padrões complexos de evolução

Este tipo de diversidade geométrica pode ser gerada a partir de técnicas complexas, como por exemplo, a aplicação de regras de vizinhança que considerem cálculos probabilísticos para realização ou não da transição de estados em função do comportamento da vizinhança da célula, ou, um conjunto de regras com proposições simples, como os utilizados para criação

da Figura 8, cujas proposições lógicas baseiam-se na conversão da base de numeração decimal para a base de numeração binária.

A similaridade entre as *classes* 3 e 4 torna extremamente complexo o enquadramento dos CAs em uma delas. Uma exemplificação desta complexidade de enquadramento é o fato de que sem uma criteriosa análise da evolução dos CAs apresentados na Figura 8 não é possível predizer quais conjuntos de regras produzem evoluções da Classe-3 e quais produzem evoluções da Classe-4. Em função desta ambiguidade na distinção entre conjuntos de regras, a classificação de CAs na Classe-4 é considerada controversa por alguns pesquisadores. Inclusive alguns pesquisadores consideram que a evolução caracterizada como Classe-4 é apenas transitória, pois após um determinado período de tempo a evolução difusa, característica da Classe-4, tende a convergir para uma evolução estável, que é uma característica das evoluções classificadas como outras classes, como o exemplo a evolução apresentada na Figura 8(c), (SILVA; MARTINS, 2003; ASSIS, 2006; MULLER, 2008; SUHADOLNIK et al., 2012).

Contudo, algumas metodologias para classificação em *classes* podem ser utilizadas pelos projetistas para resolução de alguns impasses de caracterização. Uma destas metodologias, proposta por *Wolfram* (WOLFRAM, 2006), é a realização da evolução de dois CAs com as mesmas características. Nesta metodologia, o primeiro CA evoluirá sem que ocorra qualquer alteração em sua estrutura. Já o segundo CA terá o estado de uma de suas células, escolhida aleatoriamente, alterado. Ao final de cada iteração, as evoluções são comparadas e a distância de *Hamming*, (MILIES, 2009), entre as evoluções é calculada, indicando a similaridade entre estas. Para este tipo de classificação a análise da distância de *Hamming* tem quatro possíveis resultados:

- Se tender a zero, após n iterações, considera-se que o comportamento é homogêneo, e o sistema será classificado como Classe-1;
- Se estabilizar em um valor ou em um conjunto de valores diferentes de zero, considera-se um comportamento periódico ou estável, e o sistema será classificado como Classe-2;
- Se apresentar valores diversos sem qualquer tipo de padrão, considera-se um comportamento caótico, e o sistema será classificado como Classe-3;
- Se assumir um valor estável após um longo período de valores diversos, considera-se que o comportamento é complexo, e o sistema será classificado como Classe-4.

Uma segunda metodologia de classificação foi proposta por *Li* e *Packard* (PACKARD, 1988; OLIVEIRA; OMAR; OLIVEIRA, 2010). Nesta classificação, ao invés de 4, existem 6

classes possíveis, separadas pelo tipo de conjunto regra utilizado. E, neste caso, os conjuntos de regra podem ser classificados como:

- Regras nulas, para as quais após n iterações entre células o comportamento do sistema converge para um padrão imutável formado apenas por células ativas ou inativas, 0s ou 1s;
- Regras ponto fixo, nas quais após n iterações entre células o comportamento do sistema continua constante mesmo que sejam excluídas as configurações de apenas 0s ou 1s;
- Regras de ciclo duplo, nas quais após $2n$ iterações, o comportamento do sistema retorna a um comportamento anterior;
- Regras periódicas, nas quais após n iterações, o comportamento do sistema se repete;
- Regras caóticas, cujo comportamento do sistema não possui nenhuma periodicidade;
- Regras complexas, cujo comportamento do sistema torna-se imprevisível em função do tempo para necessário para sua determinação.

Existem várias outras taxonomias que não foram citadas neste trabalho, mas que também são utilizadas para realização de tarefas de classificação de autômatos celulares. Contudo, é importante salientar que, mesmo com a existência de metodologias endossadas por diversos pesquisadores, não existe uma fórmula que consiga classificar todos os padrões de evolução de autômatos celulares. E, em alguns casos, a classificação por classes torna-se quase impossível, pois, o padrão de evolução de um CA dependerá diretamente de pelo menos três fatores (configuração inicial, conjunto de regras e vizinhança) (LI; PACKARD et al., 1990; WOLFRAM, 2002, 2006; MULLER, 2008; MELOTTI, 2009; OLIVEIRA, 2009; WEINERT, 2010).

1.3.2 Regras

O conjunto de regras, de forma idêntica a configuração inicial, é um dos fatores determinantes para o comportamento evolucionário de um CA. Para fins de exemplificação e explicitação da influência do conjunto de regras no comportamento evolucionário de um CA, serão utilizados algumas das regras elaborados por *Wolfram* (WOLFRAM, 2002), que se baseiam no sistema binário de numeração e que possuem as características definidas na Tabela 3 (WOLFRAM, 2002):

Tabela 3: Características dos CAs unidimensionais de *Wolfram*

Característica	**Definição**
Número de células vizinhas	2
Estados possíveis	2, ativo e inativo
Configuração inicial	Apenas a célula central ativa
parâmetros da regras	estado atual e vizinhança

Para fins de representação gráfica, nos exemplos desta seção, as células ativas terão a cor preta e valor lógico 1, e as células inativas terão a cor branca e valor lógico 0. A cada iteração, as células do CA verificam seu estado atual e os estados de sua vizinhança, expressando os resultados com base na conversão em números binários de suas variáveis de entrada. Em outras palavras, representando como:

- N, o número de combinações possíveis de entrada;
- W, o número de estados possíveis;
- r, o número de células vizinhas;
- P, o número de configurações possíveis para o sistema.

Matematicamente, os CAs elaborados por *Wolfram* podem ser definidos através das Equações 4 e 5 (WOLFRAM, 2002):

$$N = W^{2r+1} \tag{4}$$

$$P = W^{N} \tag{5}$$

Por conseguinte, um CA exemplo $\mathbb{D}$, cujas características foram anteriormente apresentadas na Tabela 3, terá para cada uma de suas células $N = 8$, e a representação de cada item de N conforme a descrição realizada na Tabela 4, na qual o índice de $\mathbb{D}$ indica a posição da célula no *grid*.

Tabela 4: Combinações possíveis para qualquer célula de $\mathbb{D}$

$\mathbb{D}_{n-1}$	$\mathbb{D}_n$	$\mathbb{D}_{n+1}$
0	0	0
0	0	1
0	1	0
0	1	1
1	0	0
1	0	1
1	1	0
1	1	1

Com relação aos possíveis conjuntos de regras que podem ser utilizados em $\mathbb{D}$, *Wolfram* previu um total de 256 regras possíveis para um CA unidimensional com $r = 1$, da regra 0 a regra 255. Estas regras são formadas a partir da correlação do número utilizado para nomear a regra na base decimal com o seu valor na base binária. No ato desta correlação, o valor binário da regra é distribuído ordenadamente pelas saídas possíveis do CA, neste caso 8, fazendo com que cada combinação de valores de entrada ative apenas uma das saídas possíveis. Para fins de exemplificação será apresentada a regra 30 de *Wolfram*. Esta regra, muito utilizada pelas áreas de matemática, engenharia e computação, principalmente em projetos de criptografia, possui como característica a formação de uma figura geométrica extremamente complexa no decorrer da sua evolução, proporcionando, em função da configuração inicial realizada, chaves criptográficas difíceis de serem quebradas (FILHO; HENRIQUES, 2011). Para obter as possíveis saídas geradas pela regra 30 é necessário apenas converter o número 30_{10} para a base binária, 00011110_2, e distribuir o resultado obtido ordenadamente em 8 saídas, aqui nomeadas de S_7 a S_0 conforme apresentado na Tabela 5. A cada iteração uma destas saídas será escolhida como parâmetro para evolução da célula.

Tabela 5: Resultados possíveis para a Regra 30 de *Wolfram*

Saída	S_0	S_1	S_2	S_3	S_4	S_5	S_6	S_7
Valor	0	1	1	1	1	0	0	0

Por conseguinte, a união das tabelas 4 e 5, fornecerá, para cada célula de $\mathbb{D}$, a partir das possíveis configurações de vizinhança e estado, as saídas apresentadas na Tabela 6.

Tabela 6: Combinações válidas para a regra 30 de *Wolfram*

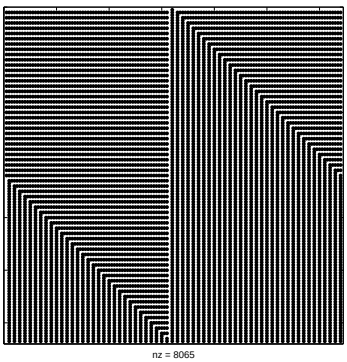
$\mathbb{D}_{n-1}$	$\mathbb{D}_n$	$\mathbb{D}_{n+1}$	**Saída**
0	0	0	0
0	0	1	1
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	0
1	1	1	0

Este formato de configuração é o mesmo realizado para todas as demais 255 regras propostas por *Wolfram*, conforme os exemplos destacados na Tabela 7, que apresenta os detalhamentos das regras 86, 160 e 227 de *Wolfram*.

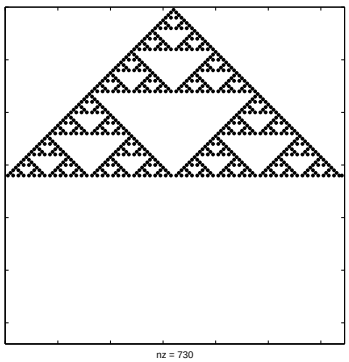
Tabela 7: Detalhamento de algumas regras de *Wolfram*

Regra	S_7	S_6	S_5	S_4	S_3	S_2	S_1	S_0
86	0	1	0	1	0	1	1	0
160	1	0	1	0	0	0	0	0
227	1	1	1	0	0	0	1	1

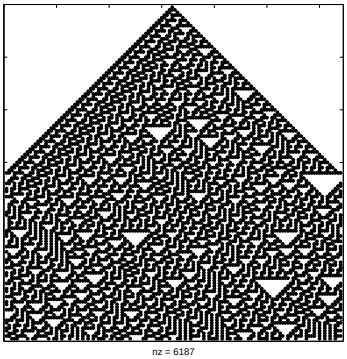
Este modelo de conjuntos de regras possui aplicabilidade a todos os CAs unidimensionais, que possuam as características apresentadas na Tabela 3. A única observação em relação a esta afirmação é a de que a aplicação terceira linha da Tabela 3 apresenta a configuração inicial que foi utilizada por *Wolfram* para realização dos experimentos, contudo, a configuração inicial será definida em função dos objetivos do sistema projetado. Retornando a exemplificação gráfica, algumas demonstrações da complexidade gráfica gerada por conjuntos de regras simples são apresentadas através dos espelhos gráficos de evolução de algumas regras de *Wolfram*. Estas evoluções, apresentadas na Figura 9, são resultado do processamento realizado pela arquitetura proposta das regras elaboradas por *Wolfram*. Para tal, utilizou-se a configuração inicial de apenas a célula central ativa, *grid* unidimensional composto por 128 células e o número de iterações configurado em 128. As regras utilizadas foram as regras 13, 18, 30 e 212, respectivamente apresentadas nas Figuras 9(a), 9(b), 9(c) e 9(d).



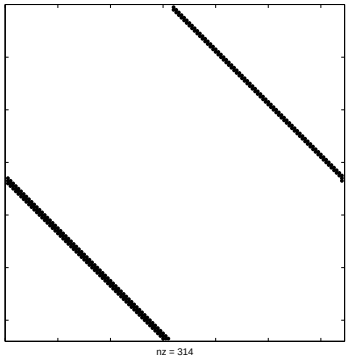
(a) Regra 13



(b) Regra 18



(c) Regra 30

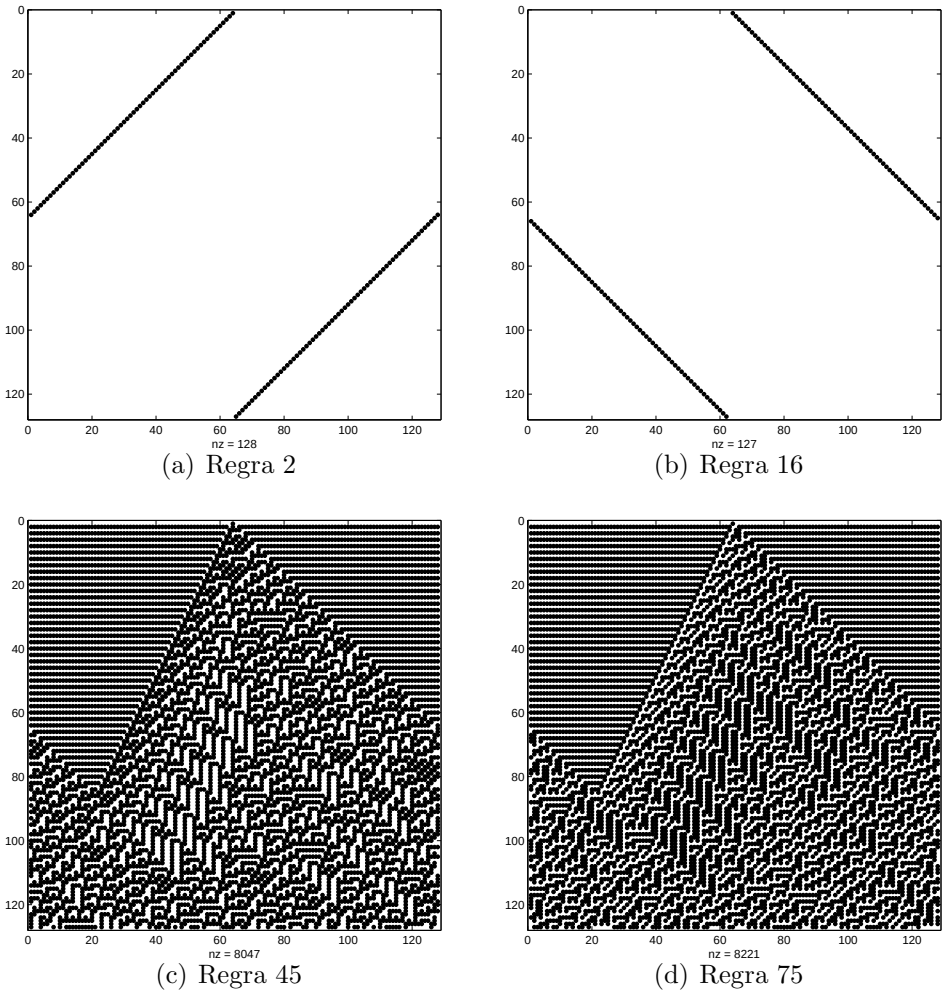


(d) Regra 212

Figura 9: Espelho da evolução de algumas regras de *Wolfram*

A análise das evoluções apresentadas permite a constatação de que a regra 13 assim como a regra 212, dado o tamanho do CA, reinicia após 64 iterações. Já a regra 30 ao longo das 128 iterações desenvolve um comportamento caótico, que é essencial para sua aplicação nos projetos de criptografias. E, a regra 18, dada a configuração inicial, limita a evolução do sistema as 64 primeiras iterações, e após estas, todas as células assumem o estado inativo, 0.

Determinados conjuntos de regras, principalmente os obtidos através de algoritmos simples, tendem a favorecer a ocorrência de regras equivalentes, como por exemplo, o caso das 256 regras de *Wolfram*, das quais apenas 88 são denominadas elementares, e as demais, pela própria definição das regras, equivalentes a estas, como por exemplo, o caso da regra 2, que possui equivalência com a regras 16 e a regra 45, que possui equivalência com a regra 75, conforme apresentado na Figura 10 (FREITAS, 2010; OLIVEIRA, 1999).



(a) Regra 2 (b) Regra 16

(c) Regra 45 (d) Regra 75

Figura 10: Exemplos de equivalência para as regras de *Wolfram*

Uma regra é considerada equivalente quando a sua tabela de transição é o complemento ou é idêntica a tabela de transição de outra regra. No caso das regras 45 e 75 é possível perceber,

após uma análise simples, que uma é o complemento da outra, pois a regra 45 possui a tabela de transição 0101101_2 e a regra 75 possui a tabela de transição 1001011_2. Para as regras 2 e 16, através da apresentação gráfica torna-se fácil, a compreensão de que uma também é o espelho da outra. A identificação de regras similares é realizada através de parâmetros de transformação, que não serão abordados neste trabalho. Estes parâmetros baseiam-se na análise do padrão estrutural e evolutivo de cada regra (OLIVEIRA, 1999). A correlação entre a regra elementar e suas equivalentes para as 256 regras de *Wolfram* é apresentada na Tabela 8, onde a primeira coluna indica a regra original, e a segunda coluna apresenta as regras que são equivalentes a esta.

Tabela 8: Relação de equivalência das regras de *Wolfram*(OLIVEIRA, 1999)

Regra	**Equivalente**	**Regra**	**Equivalente**	**Regra**	**Equivalente**
0	255	35	49,59,115	108	201
1	127	36	219	110	124,137,193
2	16,191,247	37	91	122	161
3	17,63,119	38	52,155,211	126	129
4	223	40	96,235,249	128	254
5	95	41	97,107,121	130	144,190,246
6	20,159,215	42	112,171,241	132	222
7	21,31,87	43	113	134	148,158,214
8	64,239,253	44	100,203,217	136	192,238,252
9	65,111,125	45	75,89,101	138	174,208,244
10	80,175,245	46	116,139,209	140	196,206,220
11	47,81,117	50	179	142	212
12	68,207,221	51		146	182
13	69,79,93	54	147	150	
14	84,143,213	56	98,185,227	152	188,194,230
15	85	57	99	154	166,180,210
18	183	58	114,163,177	156	198
19	55	60	102,153,195	160	250
22	151	62	118,131,145	162	176,186,242
23		72	237	164	218
24	66,189,231	73	109	168	224,234,248
25	61,67,103	74	88,173,229	170	240
26	82,167,181	76	205	172	202,216,228
27	39,53,83	77		178	
28	70,157,199	78	92,141,197	184	226
29	71	90	165	200	236
30	86,135,149	94	133	204	
32	251	104	233	232	
33	123	105			
34	48,187,243	106	120,169,225		

1.3.3 Estados

Os estados de um CA, como citado anteriormente, são uma reação da célula a uma interferência externa, permitindo para o projetista a possibilidade de que seu modelo de CA tenha quantos estados o projeto demandar, desde que todos os estados estejam definidos nos parâmetros de evolução do sistema. Portanto, quanto mais estados o sistema possuir mais complexa será a sua evolução, dificultando a análise de erros nos resultados obtidos.

Com relação a caracterização, os estados podem ser cores, números, odores, notas musicais, ou seja, qualquer grupo de itens em que seja possível evidenciar claramente a ocorrência de variações, sendo mantida a característica de que o número mínimo de estados para qualquer CA deverá ser 2 e que as restrições do número máximo de estados são inerente aos objetivos do projeto. Esta flexibilidade na criação de um modelo de CA permite que estes sejam utilizados para os fins mais diversos, como por exemplo, entretenimento, modelagem computacional e estudos médicos (OLIVEIRA et al., 2006; GOMES; FERRACIOLI, 2006; MELOTTI, 2009).

1.4 Vizinhança

Conforme citado anteriormente, vizinhança é o nome dado às células pertencentes a um CA, cujos estados atuais interferem na evolução de outra célula. A abrangência de uma vizinhança, ou seja, a quantidade de células que podem ser classificadas como vizinhas de uma determinada célula dependerá única e exclusivamente da vontade do projetista e do objetivo do projeto.

O formato estrutural do *grid* sempre irá coincidir com uma figura geométrica, e portanto, a escolha da vizinhança deverá resolver o problema denominado efeito de borda, enumerando as condições de fronteira. Efeito de Borda é o problema ocasionado pelo isolamento das células do extremo da figura, existindo dois tipos possíveis de solução para este problema (MELOTTI, 2009):

1. Limite fixo, no qual as células da extremidade possuem estados fixos durante toda simulação;

2. Limite reflexivo, ou referência circular, no qual as células localizadas na borda do *grid* são vizinhas entre si transformando o CA em um circulo, quando o *grid* é unidimensional, ou em um toroide/esfera no caso dos CAs n-dimensionais.

A escolha de uma destas soluções dependerá única e exclusivamente dos objetivos do projeto, que irá considerar, dentre outros itens o raio de vizinhança que será utilizado. Raio de vizinhança é a distância, em células, da extensão dos elementos integrantes do conjunto

vizinhança de uma célula. Se em um *grid* o raio de vizinhança for igual a 1, serão elementos do conjunto vizinhança todas as células adjacentes a célula em análise. Se neste mesmo *grid* o raio de vizinhança for igual a 2, serão elementos do conjunto vizinhança todas primeiras e segundas células imediatamente adjacentes a célula em análise. Alguns modelos de vizinhança foram definidos, de forma a facilitar esta escolha. Estes modelos, cujos conceitos são utilizados para classificação dos CAs em dimensões, são as vizinhanças de *Wolfram*, *Neuman*, *Moore* e Estocástica, que foram baseados na teoria proposta por *Turing* e serão apresentados na seção seguinte (WOLFRAM, 2002; WILLIS, 2002; GREMONINI; VICENTINI, 2008).

1.4.1 Vizinhança Unidimensional

Introduzindo um referenciamento matemático para a conexão estrutural entre as células de um *grid*, considera-se como vizinhança de *Wolfram*, $\mathbf{V}_W$, para uma determinada célula o conjunto composto pelas células lateralmente adjacentes a esta. No caso das células localizadas nas extremidades do *grid*, para composição do conjunto vizinhança será utilizado o conceito de referência circular. Este tipo de organização, irá transformar o CA em um vetor, considerando-se assim como CA unidimensional todos aqueles para os quais a descrição da conexão entre as células pode ser representada por um vetor. Neste caso, para não ocorrência do efeito de borda, o CA deverá ser considerado uma estrutura circular, na qual a primeira célula sempre será vizinha da última e vice-versa, conforme idealizado por *Turing* e exemplificado na Figura 11.

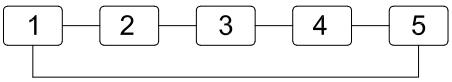


Figura 11: Referência circular para CAs unidimensionais

Para fins de exemplificação do conceito de CA unidimensional, será utilizado o CA $\mathbb{C}$, que foi apresentado na Figura 7, e que possui as características definidas na Tabela 1. Considerando que $\mathbb{C}$ seja unidimensional, pela representação gráfica o mesmo trata-se de um vetor horizontal, e as suas células podem ser organizadas da esquerda para a direita da forma $\{\mathbb{C}_{1,1}, \mathbb{C}_{1,2}, \mathbb{C}_{1,3}, \mathbb{C}_{1,4}, \mathbb{C}_{1,5}\}$.

Aplicando o conceito de referência circular, $\mathbb{C}_{1,1}$ terá como vizinha $\mathbb{C}_{1,5}$, e de forma idêntica $\mathbb{C}_{1,5}$ terá como vizinha $\mathbb{C}_{1,1}$. Em relação ao raio, se $r = 1$, $\mathbb{C}_{1,3}$ terá como vizinhança apenas as células $\mathbb{C}_{1,2}$ e $\mathbb{C}_{1,4}$, conforme apresentado na Figura 12(a) na cor preta. Já com $r = 2$,

a vizinhança de $\mathbb{C}_{1,3}$ se estenderá as células $\mathbb{C}_{1,1}$ e $\mathbb{C}_{1,5}$, conforme apresentado na Figura 12(b), também na cor preta.

(a) $r = 1$ (b) $r = 2$

Figura 12: Vizinhanças da célula $\mathbb{C}_{1,3}$

1.4.2 Vizinhança Multidimensional

Quando uma célula possui um conjunto *vizinhança*, $\mathbf{V}$, com mais de 2 elementos adjacentes no plano bidimensional, este CA é denominado *bidimensional*. Para este caso, a organização visual do sistema irá representar uma matriz $M_{i,j}$, para a qual i e j são diferentes de 1. Estes CAs utilizam as mesmas soluções para o efeito de borda citadas anteriormente, existindo como padrão 3 modelos diferentes de vizinhança: a de *Neuman*, a de *Moore* e a Estocástica. A vizinhança de *Neuman*, $\mathbf{V}_N$, de uma célula é formada pelas células vertical e horizontalmente adjacentes a esta, conforme exemplificado na Figura 13, na qual a célula em análise está em cinza e seu conjunto vizinhança é identificado em preto. Neste exemplo é representada a primeira vizinhança de *Neuman*, ou seja, as primeiras células adjacentes a célula em análise.

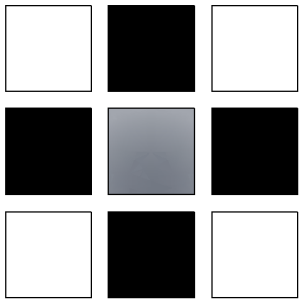


Figura 13: Modelo de vizinhança de *Neuman*

A vizinhança de *Moore*, $\mathbf{V}_M$, é formada por todas as células que façam fronteira no espaço bidimensional com a célula analisada, sejam estas verticais, horizontais ou diagonais, conforme exemplificado na Figura 14, na qual a célula analisada está em branco e seu conjunto $\mathbf{V}_M$ em preto. Neste caso é apresentada a primeira vizinhança de *Moore*, ou seja, as primeiras células adjacentes a célula em análise.

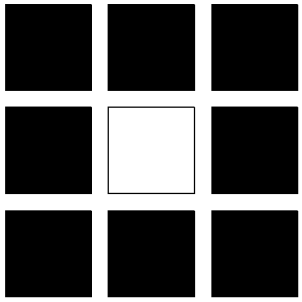


Figura 14: Modelo de vizinhança de *Moore*

Para ambos os casos, uma das possíveis soluções para o efeito de borda é o referenciamento circular, que para *grids* bidimensionais é realizado através da conversão da matriz no formato visual de um toróide, conforme exemplificado na Figura 15. Esta adaptação, realizada a partir do referenciamento circular de CA bidimensional de *grid* 4×4, faz com que todas as células da extremidade se tornem vizinhas, permitindo a identificação das vizinhanças relacionadas, como por exemplo, as células 5 e 8 e as células 9 e 12.

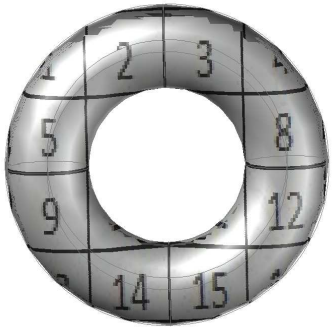


Figura 15: Referência circular para vizinhanças bidimensionais

A vizinhança estocástica, utilizada normalmente para confecção de modelos de CA Classe-3 e Classe-4, introduz o conceito de que uma célula pode possuir como vizinha qualquer outra célula do *grid*, independentemente de onde estejam localizadas. Uma exemplificação deste tipo de vizinhança é apresentada na Figura 16, na qual a célula analisada está em cinza, a vizinhança desta célula esta preto e as células que não compõem a vizinhança estão em branco. Neste caso, a definição de quais células farão parte do conjunto *vizinhança* fica a cargo de um algoritmo que utiliza variáveis probabilísticas e estatísticas. Ainda com relação a vizinhança estocástica, o algoritmo utilizado pode fazer com que ao longo da evolução do sistema os elementos do conjunto vizinhança sejam alterados.

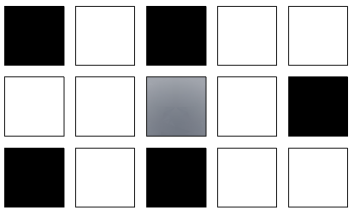


Figura 16: Exemplo de vizinhança estocástica

Em relação ao projeto de dimensão de um CA, podem ser confeccionados modelos com mais de duas dimensões, como por exemplo, o modelo de vizinhança tridimensional de *Neuman*, que é uma expansão do modelo bidimensional de *Neuman* para o ambiente tridimensional. Neste caso a vizinhança da célula é composta pelas células localizadas acima, abaixo, a esquerda, a direita, a frente e atrás da célula em análise formando para cada célula a estrutura de vizinhança apresentada na Figura 17.

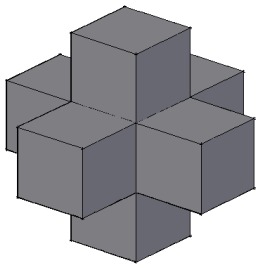


Figura 17: Modelo de vizinhança tridimensional de *Neuman*

Outro exemplo de vizinhança tridimensional é o modelo de vizinhança tridimensional de *Moore*. Neste modelo a vizinhança de uma célula é composta por todas as células nas direções cardeais adjacentes a esta no ambiente tridimensional, formando no entorno da célula a figura de um cubo, conforme apresentado na Figura 18.

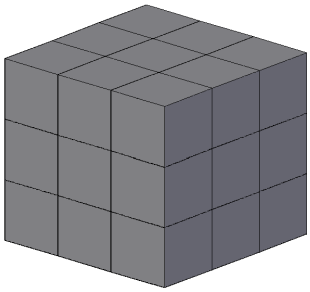


Figura 18: Modelo de vizinhança tridimensional de *Moore*

Os exemplos tridimensionais citados utilizam $r = 1$, mas para modelos com $r > 1$ basta efetuar a continuação da expansão da vizinhança. Para ambos os casos de vizinhança tridimensional apresentados, a resolução dos problemas de borda é realizada através da aplicação de referenciamento circular, que deverá ser realizado para todas as extremidades do *grid* tridimensional (GANDIN; RAPPAZ, 1997; ARATA et al., 1999; DAVIES, 2009).

Existem ainda outros modelos de vizinhança, como por exemplo, a vizinhança de *Margolus* e *Toffoli* (COLOMBO, 2011), que possuem diferentes parâmetros para definição dos elementos que pertencem ao conjunto vizinhança. Estes parâmetros podem ser, por exemplo, a expansão de modelos já existentes ou a combinação de diversos modelos. Nestes casos, caberá ao projetista a definição das formas para resolução dos problemas de efeito de borda.

A vantagem dos CAs *n dimensionais* sobre os CAs unidimensionais é o aumento no número de possibilidades de aplicação, uma vez que estes permitem a modelagem de dinâmicas com diversas variáveis, como por exemplo, sistemas imunológicos e de tráfego (FREITAS; JAFELICE, 2006; RANGEL; BALTA; VICENTE, 2006; LIMA, 2007; TAVARES, 2010; ALVES et al., 2012).

1.5 Considerações Finais do Capítulo

No decorrer deste capítulo foram apresentadas as definições de classe, estados, regras e vizinhança, abordando a relação entre estes e a importância da utilização de CAs no estudo de dinâmicas complexas, conceitos estes que foram necessários para moldar a arquitetura projetada no âmbito desta dissertação. O capítulo seguinte apresenta uma visão geral acerca de som e música e os detalhes do protocolo MIDI utilizados nesta dissertação.

Capítulo 2

SOM E MÚSICA

ESTE capítulo apresenta algumas definições básicas e terminologia de som e de música, e a importância destes para a evolução da humanidade. Em conjunto com estes itens são apresentados alguns conceitos utilizados na composição automática de música que serviram como base para o trabalho apresentado nesta dissertação. De forma a abordar os temas propostos este capítulo foi estruturado em três seções. A Seção 2.1 pontua a importância do som, e consequentemente da música para a humanidade, contextualizando a evolução da composição automática de músicas. A Seção 2.2 define o termo música e apresenta seus conceitos fundamentais abordando de forma generalista os elementos que influenciam na composição musical. Na Seção 2.3, os conceitos apresentados são complementados a partir das definições do protocolo MIDI e de seus parâmetros de escrita.

2.1 Breve Histórico

Desde seus primórdios, a humanidade vêm utilizando o som para atender suas demandas nas mais diversas aplicações, como por exemplo, defesa, comunicação, caracterização e descontração. Em virtude deste fato, com o passar dos anos o som tornou-se um dos parâmetros mais utilizados para identificação e contextualização de desenvolvimento e da expressão sociocultural de nações e grupos distintos (WISNIK, 2007).

A audição, assim como o olfato, pode ser considerada um sentido abstrato uma vez que cada ser humano, em função da época e do local em que reside, efetua uma classificação subjetiva de som e consequentemente de música. Este nível de abstração com o passar do tempo gerou alguns questionamentos, como por exemplo, por que a música consegue incitar tristeza e euforia ou por que o som de uma música tocada em um piano é diferente da mesma música tocada em uma flauta (MACHADO, 2009; LEVEK; SANTIAGO, 2012; MORI; DEUTSCH, 2005). De forma a resolver estes questionamentos, no decorrer dos séculos, foram definidos

conceitos para classificação de sons, objetivando a compreensão e a modificação de seu processo de geração (PINTO, 2001; WISNIK, 2007; FONSECA, 2007).

Deste aprendizado surgiram os conceitos musicais conhecidos atualmente e um destes conceitos, em especial, tem sido vastamente explorado pelos pesquisadores das áreas de Inteligência Artificial. Este conceito, que possui três denominações música algorítimica, aleatória ou música do acaso, consiste na geração de música harmonizada através da combinação pseudo aleatória de sons (TERRA, 2000; WISNIK, 2007).

O desenvolvimento e a automatização de ferramentas para composição de música aleatória sempre esteve presente no imaginário humano, tornando-se um vasto campo de estudos e também um desafio em função do tênue limiar entre os conceitos de música e ruído. A insistência na produção de métodos para este processo criativo deve-se, em grande parte, a curiosidade humana na descoberta e identificação do funcionamento do lado direito de seu cérebro, responsável, dentre outros itens, pelas decisões intuitivas e subjetivas, como por exemplo, a realização de algumas tarefas inerentes a interpretação musical. Uma das alternativas encontradas para geração de música aleatória é a utilização de algoritmos e outros raciocínios lógicos em substituição ao ser humano como compositor musical. Tais técnicas vêm sendo utilizadas a certo tempo conforme demonstram, dentre outros, os seguintes fatos (TERRA, 2000; MACEDO, 2009):

- Em 1787, *Amadeus Wolfgang Mozart* criou uma partitura composta por 196 fragmentos musicais numerados. Posteriormente, para criar as sequências musicais de sua obra intitulada *Musikalisches Würfelspiel*, ele uniu os fragmentos desta partitura, sorteados de forma aleatória através do lançamento de dados não viciados (RANSBEECK, 2009).

- Em 1946, *Guillaume Dufay* usou as proporções da nave, do cruzeiro, da abside e a altura da abóbada da catedral de *Santa Maria del Fiore* (Florença, Itália), respectivamente 6, 4, 2 e 3, para criar as proporções isorrítmicas de seu motete intitulado *Nuper Rosarum Flores* (CROSBY, 1999).

- Em 1957, *Lejaren Hiller* e *Leonard Isaacson* desenvolveram o projeto *Illiac Suite*, que consistiu na utilização de cadeias de *Markov* para escolha de notas musicais de forma pseudo aleatória, criando-se assim a primeira melodia composta em totalidade por um computador (HILLER; ISAACSON, 1957).

- Em 1980, *David Cope* publicou o EMI (*Experiments in Musical Intelligence*). Este trabalho apresentava uma ferramenta capaz de gerar composições em estilo clássico semelhante

ao de compositores consagrados, através de 3 passos (COPE, 1991; COPE; MAYER, 1996):

1. Análise da música em relação a melodia, estrutura e harmonia para posterior divisão da música em fragmentos.
2. Aplicação de uma técnica para reconhecimentos de padrões de forma a classificar o estilo do compositor em questão e a criar uma gramática representativa.
3. Criação de uma nova melodia com base nos padrões obtidos.

- Em 2004, Guéret, Monmarché e Slimane desenvolveram o projeto *AntMusic*, que consistiu na utilização da metaheurística de colônia de formigas para criação de música com o auxílio de computadores. Este projeto basou-se na utilização das probabilidades de tomada de decisão, através de escolha de caminhos, de forma a obter as notas musicais e as durações destas (GUÉRET; MONMARCHÉ; SLIMANE, 2004).

2.2 Definições Básicas e Terminologia

O som, conceitualmente, pode ser definido de duas formas:

1. Foneticamente, como ruído característico produzido por uma fonte sonora (AULETE; GARCIA; NASCENTES, 1986);
2. Fisicamente, como as variações de pressão que são captadas pela audição humana, ou seja, as que ocorrem na Faixa de 20 *Hz* a 20 *KHz* (FONSECA, 2007).

A partir destas duas definições, extrai-se o conceito de *fonte sonora*. Fonte sonora é qualquer objeto capaz de causar perturbações sonoras em um ambiente, dentro da faixa perceptível pela audição humana, como por exemplo, o impacto entre objetos ou o deslocamento de alguns animais. Portanto, sons são produzidos de forma voluntária e involuntária durante todo o tempo. Contudo, a audição de um determinado som por um ser humano dependerá diretamente das propriedades do meio e das características do sistema auditivo do ouvinte (CARMO, 1999; BARALDI; ALMEIDA; BORGES, 2007).

Uma peculiaridade no processo auditivo é o fato deste processo ser quase que em totalidade mecânico, fazendo com que a faixa audível não seja nem absoluta nem linear. Como o processamento humano de áudio inicia-se no canal auditivo e termina no cérebro, qualquer processo de identificação sonora é resultado de um criterioso processo de análise de algumas

propriedades físicas do som, que podem ser organizadas em 4 itens (OLIVEIRA, 2004; FONSECA, 2007):

- Timbre, que corresponde ao resultado da junção das harmônicas formadoras do som. O timbre também é conhecido como identidade sonora, pois é o que permite distinguir, por exemplo, o som de uma nota musical extraída de um violão do som desta mesma nota emitida por um violino;

- Intensidade, que corresponde a amplitude das oscilações da pressão do ar. A intensidade pode ser interpretada como o fluxo de energia por unidade de área, ou seja, a pressão causada pela velocidade das partículas no meio;

- Altura, também conhecida como tom, é a característica correspondente a frequência das oscilações, ou seja, permite distinguir entre um som grave (baixa frequência) e um som agudo (alta frequência);

- Duração, que indica o período de ocorrência das oscilações sonoras audíveis [1].

Para exemplificar a complexidade das tarefas realizadas pelo sistema humano durante o processamento sonoro, nas Figuras 19 e 20 são apresentados espectros de ondas sonoras, produzidas artificialmente através do uso do *software Audacity* (SOUSA; BESSA, 2008). O primeiro espectro é o apresentado na Figura 19.

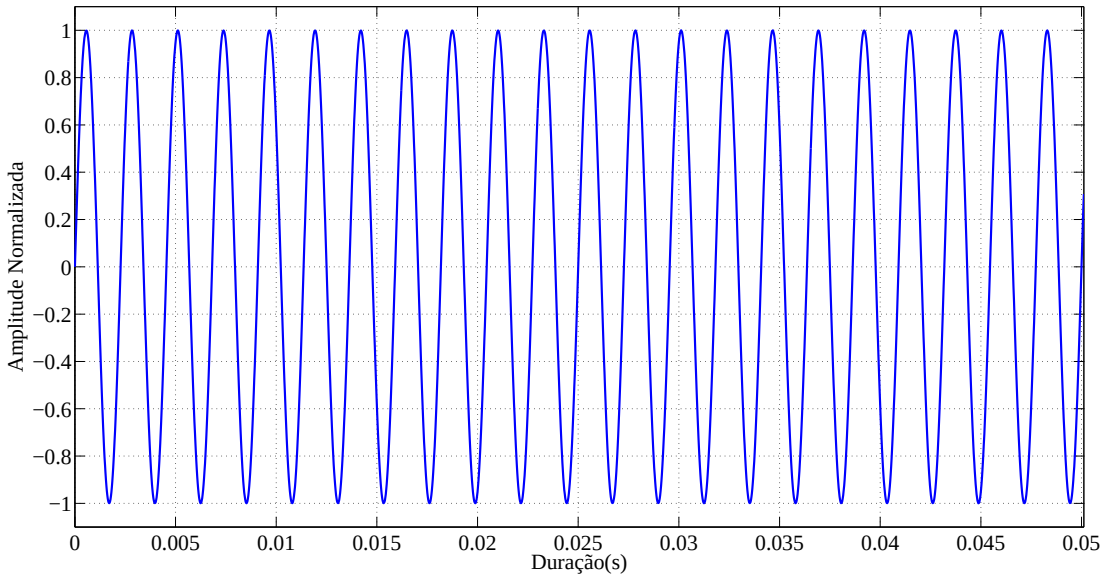


Figura 19: Espectro no domínio do tempo de uma onda sonora simples

[1] Alguns livros e artigos não aceitam a duração como característica sonora

Este espectro, apresentado na Figura 19, é de uma onda sonora simples, cujo som é denominado tom puro. Tom puro é o nome dado a sons que são formados por apenas uma frequência (ODA; BERNARDI; AZEVEDO, 2003). Para tal, a onda sonora, que possui duração de 0,05 segundos, foi confeccionada com intensidade de 0,2 a uma frequência de 440 *Hz*. Através deste gráfico, produzido no *software MatLab*,(MATHWORKS, 1992), é possível visualizar que um tom puro é contínuo ao longo do tempo, configurando-se uma sucessão de senoides de mesma amplitude.

Na Figura 20 é apresentado o espectro de uma onda complexa, obtido através do processamento computacional de um som conhecido pelo ouvido humano, o bater de palmas, cujo resultado é um timbre composto por diversas harmônicas de intensidade e altura variadas. A junção de harmônicos é o método utilizado para a produção de grande parte dos sons processados pelo sistema auditivo humano, como a fala, a música e o ruído. As tarefas realizadas pelo sistema auditivo para processar ondas sonoras simples ou complexas são similares. O grande diferencial entre indivíduos durante a realização do processamento sonoro é a fase de interpretação, que é realizada pelo cérebro. Esta fase depende diretamente das características do ouvinte, o que pode fazer com que duas pessoas ouçam o mesmo som e efetuem interpretações completamente distintas.

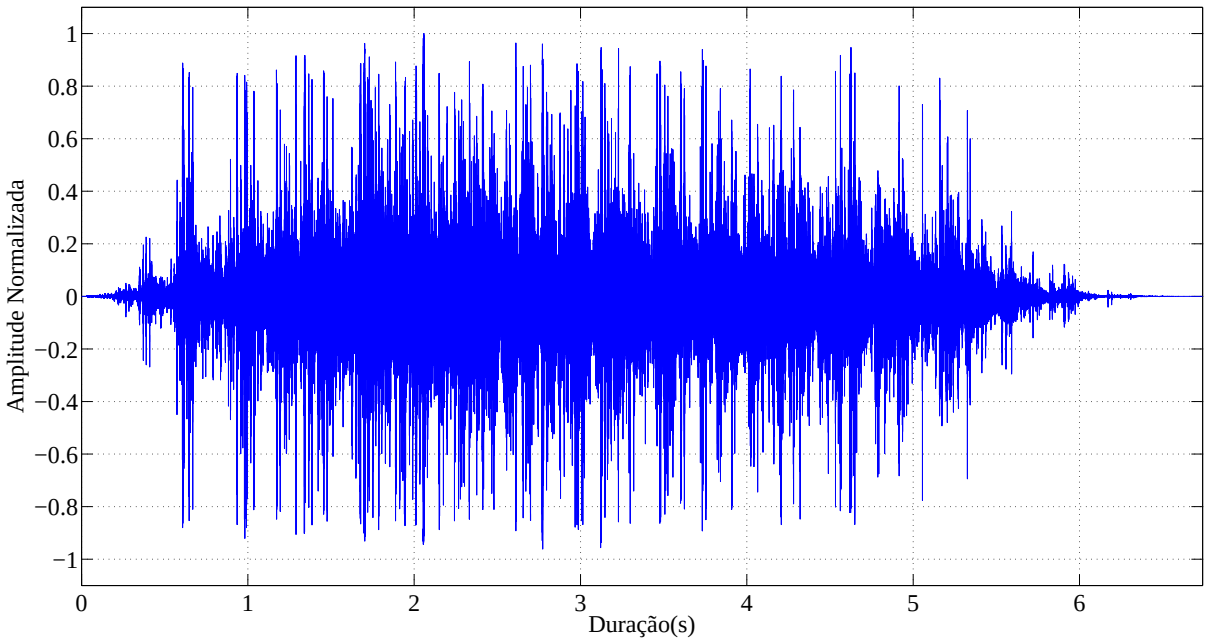


Figura 20: Espectro no domínio do tempo de uma onda sonora complexa

Em termos de classificação sensorial, o som harmônico comumente é dividido em duas categorias separadas por uma linha extremamente tênue: barulho e música. Esta separação é

dita como tênue porque, apesar de existirem alguns parâmetros estruturados para a classificação de um som harmônico em uma destas duas categorias, o real enquadramento dependerá apenas de critérios subjetivos estabelecidos pelo ouvinte. Em termo gerais, define-se barulho como o som conflitante ou constante exibido durante um determinado intervalo de tempo e música como som harmonioso exibido durante um determinado intervalo de tempo (AULETE; GARCIA; NASCENTES, 1986; LISBOA, 1998).

Como ambas as definições são relacionadas ao intervalo de tempo durante o qual o som é executado, em algumas composições ou apresentações sonoras é comum que os autores utilizam itens classificados como barulho para criar a estrutura do composto musical, como por exemplo, o zumbido de mosquito, o bater de palmas e o choque entre objetos.

De forma idêntica, alguns sons normalmente tidos como músicas, quando exibidos de forma pausada no decorrer de um longo intervalo de tempo, recebem a classificação de barulho, como por exemplo, o tilintar de copos e o impacto de gotas de água no solo (BRÄUTIGAM, 2005; CALDEIRA, 2007). Entretanto, para fins de desambiguação esta dissertação baseia-se nas definições fundamentais de barulho e música citadas anteriormente.

2.2.1 Música

Ampliando a definição de música realizada anteriormente, o conceito cientificamente conhecido de música é o de que, música é a combinação de sons e silêncio (ausência de som) capaz de ativar, durante seu período de execução, múltiplas áreas do cérebro humano, despertando sensações que normalmente são vinculadas a experiências anteriores (TERRA, 2000; IAZZETTA, 2001; PIGNATARI, 2005). Fisiologicamente, no decorrer da audição de uma música as áreas cerebrais ativadas liberam substâncias químicas capazes de desencadear as mais diversas e particulares sensações ao ouvinte (LISBOA, 1998; MUSZKAT; CORREIA; CAMPOS, 2000).

Em termos gerais um som pode ser considerado música quando é resultado de uma combinação harmoniosa dos chamados tons puros, ou seja, quando é formado por vários sons periódicos de apenas uma frequência, denominados notas musicais. O equilíbrio desta combinação de som e silêncio, apesar de produzir um resultado subjetivo no ouvinte, pode ser racionalizado através de alguns conceitos fundamentados em matemática e física, criando-se uma divisão estrutural da música. Esta divisão concentra-se no domínio do tempo em que cada nota será executada e na combinação de frequências tonais (WUENSCHE, 2006; MESQUITA, 2012).

2.2.2 Conceitos Básicos de Harmonia e Melodia Musical

Harmonia musical é a parte da teoria musical que descreve a forma com a qual os acordes serão encadeados. Acordes são sons simultâneos, ou seja, sons compostos por duas ou mais notas musicais, contudo, alguns teóricos definem o agrupamento de sons simultâneos como, (COSTA, 2005; RATTON, 1993):

- Intervalo, quando duas notas musicais são tocadas simultaneamente;
- Acorde, quando três ou mais notas musicais são tocadas de forma simultânea;

Um acorde com 3 notas, denominado *tríade*, divide-se em 3 partes:

- Nota Mais Grave, ou nota de referência, que por ter o som mais grave dá nome ao acorde;
- Nota Modal, ou 3^a, que irá definir se o acorde é maior ou menor e está localizada a uma *terça* da nota fundamental;
- Nota 5^a, ou nota de composição do acorde, localiza-se a uma *terça* da nota modal.

Existem 4 tipos de *tríade*, a *maior*, a *menor*, a *aumentada* e a *diminuta*. A *tríade maior* é formada pela superposição de duas *terças*, a primeira é a *terça maior* e a segunda é a *terça menor*, ambas em relação a nota de referência, como por exemplo, a combinação *Sol*, *Si* e *Ré*. A *tríade menor* é formada também pela superposição de duas *terças*, invertidas em relação a *tríade maior*, como por exemplo, a combinação *Dó*, *Mib* e *Sol*. A *tríade aumentada* é feita através da superposição de duas *terças maiores*, como por exemplo, a combinação *Dó*, *Mi* e *Sol#*. A *tríade diminuta* é formada pela superposição de duas *terças menores*, como por exemplo, a combinação *Dó*, *Mib* e *Solb* (ADOLFO, 1989; MATTOS, 2010; TADDEI, 2012).

Considerando que *intervalo* é a distância entre duas notas musicais, uma *terça musical* é um *intervalo* com distância de três notas musicais, uma *segunda musical* é um *intervalo* com distância de duas notas musicais e uma *quarta musical*, é um *intervalo* com distância de quatro notas musicais. Em termos de harmonia, um *intervalo musical* pode ser classificado como (MACHADO, 1941; ARCELA, 1994):

- Melódico, quando é formado por duas notas sucessivas, sejam estas ascendentes ou descendentes. Este tipo de *intervalo* subdivide-se em:
  1. Conjunto, quando é formado por duas notas consecutivas;
  2. Disjunto, quando é formado por notas não consecutivas;

- Harmônico, quando é formado por duas notas simultâneas;
- Simples, quando é formado por notas da mesma oitava;
- Composto, quando é formado por notas de oitavas diferentes.

Definidos estes conceitos, torna-se possível a utilização de uma definição formal para a relação existente entre os termos: música, intervalo, harmonia e melodia. Música é um conjunto formado por:

- Elementos horizontais, ou *urdidura*, que correspondem aos elementos sucessivos que formam as melodias;
- Elementos verticais, ou *trama*, que correspondem aos elementos sucessivos que formam as harmonias.

A relação entre as alturas gera um *intervalo* e o estudo de *intervalos simultâneos* é denominado *harmonia*. A variação entre *intervalos* é denominada *percurso sonoro* e qualquer indivíduo, que possua sistema auditivo completo e funcional, consegue realizar a identificação dos *movimentos melódicos*, que podem ser *ascendentes* ou *descendentes*, durante a audição de uma música. *Movimento melódico* ou *contorno melódico* é o resultado da sucessão de vários percursos sonoros. Quando um *movimento melódico* é constituído por variações lógicas e coerentes este movimento é denominado *melodia* (ANDRADE, 2004; BERTON, 2005; FALCÓN, 2012).

Os conceitos musicais explicitados neste capítulo tem apenas o objetivo de embasar a compreensão das demais seções, com o intuito do correto entendimento da grafia de um arquivo *.mid*, que é uma das bases desta dissertação, e do produto gerado pelo processamento realizado pela arquitetura desenvolvida no decorrer desta dissertação.

2.2.3 Tempo Musical

Notas musicais não possuem tempo fixo de execução, ou seja, um músico pode tanto executar uma nota durante apenas uma fração de segundos quanto durante vários minutos. Portanto, tornou-se necessária a criação de um identificador para que a reprodução exata de uma determinada melodia pudesse ser realizada por outro músico através apenas da leitura de uma partitura. A solução para este problema foi a criação dos elementos temporais da música, denominados *figuras musicais*, que são as *figuras de pausa* e as *figuras de nota* (MACHADO, 1941; CARDOSO; MASCARANHAS, 1973; SCHOENBERG, 1990; ADOLFO, 2002).

O *tempo musical* é calculado em batidas e a duração de uma nota é representada graficamente em uma partitura através das *figuras musicais*, cuja equivalência, baseada na *semínima*, é apresentada na Tabela 9.

Tabela 9: Relação de tempo, com base na *semínima*, entre as *figuras musicais*

Figura	**Duração**
semibreve	4 tempos
mínima	2 tempos
semínima	1 tempo
colcheia	1/2 tempo
semicolcheia	1/4 tempo
fusa	1/8 tempo
semifusa	1/16 tempo

Já o silêncio é representado pelo elemento *pausa*, cujo tamanho será idêntico ao da *figura* que este simboliza, como por exemplo, se uma *semibreve* tem 4 *tempos*, uma *pausa de semibreve* também terá 4 *tempos*.

Em teoria musical, as *pausas* e as *figuras* são representadas através de divisores da *semibreve*, que é a *figura musical* de maior duração. Contudo, na computação optou-se pela contagem de tempo a partir dos múltiplos e divisores da *semínima*, que corresponde a $\frac{1}{4}$ de *semibreve*. A esta contagem denominou-se PPQ - *Pulse Per Quarter Note* ou PPS - *Pulses Per Semínima*. A finalidade do PPQ é indicar a precisão com a qual uma nota deverá ser gravada. Como geralmente o valor máximo, em *softwares*, para PPQs é 480, cada *semínima* costuma ser dividida em 480 segmentos para que seja feita a sua gravação. Contudo, a *semínima* pode ser dividida em valores maiores ou menores em função dos objetivos do compositor. A Tabela 10 apresenta, para fins de exemplificação, uma correlação entre algumas *figuras de nota* e a sua correspondente em PPQ, utilizando como base uma *semínima* e a segmentação de um PPQ igual a 480 (LIMA; MACHADO; PINTO, 2002; CAMPOS; LIMA; PINTO, 2003).

Tabela 10: Correlação de PPQ entre as figuras musicais

Figura	**PPQ**
semínima	1
colcheias	2
semicolcheia	4
fusa	8
semifusa	16
quartifusa	32

Outro importante elemento na contagem de tempo musical é o *compasso*. O *compasso* é o elemento que fixa a quantidade de *batidas* através do uso de racionalização matemática,

dividindo quantitativamente um grupo de sons. Esta divisão pode classificar o *compasso* em três tipos:

- Binário, quando o *compasso* possui 2 *tempos*;
- Ternário, quando o *compasso* possui 3 *tempos*;
- Quartenário, quando o *compasso* possui 4 *tempos*.

Os *compassos* são ainda divididos em *simples* e *compostos*. Um *compasso simples* é aquele no qual a unidade de tempo determina a *fórmula de compasso*, como por exemplo, os *compassos* 2/4, 3/4, 4/4 que possuem a duração de uma semínima. O *compasso* composto é aquele cuja unidade de tempo é subdividida em 3 notas, como por exemplo, o *compasso* 6/8. Neste caso o tempo é o de uma *colcheia*, que possui $\frac{1}{8}$ de *semibreve*. Por conseguinte, no *compasso* 6/8 serão utilizadas 6 *colcheias* ($6 \times \frac{1}{8}$). Como cada pulso do *compasso* 6/8 é composto por 3 notas, e a *métrica* deste *compasso* é *binária*, seu tempo relativo será de 3 *colcheias* e não 6 [2] (SCHOENBERG, 1990; LIMA; MACHADO; PINTO, 2002; CAMPOS; LIMA; PINTO, 2003).

Exemplificando a leitura do *compasso*, considerando que em teoria musical os *tempos* são divididos em *semibreves*, um *compasso* 4/4 significa que cada unidade de tempo terá duração de uma *semínima*, que corresponde a $\frac{1}{4}$ de *semibreve*, e que este terá o total de 4 unidades de *tempo*. Neste caso uma *semibreve* ocupará todo o *compasso*, $4 \times \frac{1}{4}$, devendo as *pausas* e as *notas* durarem exatamente 4 unidades.

Existe ainda o conceito musical de *força*. Todos os *compassos compostos* são variações de *compassos simples*, e as definições de marcação de *tempo musical forte* e *tempo musical fraco* servem para ditar o andamento musical. Quando um *tempo musical* é denominado *forte* significa que ele é executado de forma mais rápida. De forma análoga o *tempo musical fraco* é executado de forma mais lenta. Os *compassos binários* podem ser *simples* ou *compostos*, desde que ocorra a *divisão binária*, e são formados por dois *tempos musicais*, o primeiro *forte* e o segundo *fraco*. O ritmo binário é normalmente utilizado em marchas, em algumas composições de música erudita e no *Jazz*. Contudo este *compasso* é mais conhecido por ditar o ritmo de músicas populares, como por exemplo, o Frevo, o Baião, o *Ska* e a Bossa Nova.

Já o *compasso ternário*, que pode ser simples ou composto, é normalmente utilizado em danças medievais e eruditas, como a Valsa e a Guarânia, mas também pode ser encontrado no

[2] As simbologias de numerador e denominador foram utilizados apenas como referência para o leitor, pois as numerações de *compasso* não seguem regiamente a lógica matemática de frações

Jazz. No *compasso quaternário*, em geral, o primeiro pulso é *forte*, o segundo e o quarto pulsos são *fracos* e o terceiro pulso é *médio*. Existem ainda os *compassos complexos* e os *compassos particulares* muito utilizados em *Rock* Progressivo e Rumba, que são formados a partir da justaposição de vários tipos de *compasso* (MASCARENHAS; CARDOSO, 1973; NOBRE, 2008; II, 2011; SCLIAR, 1986; LIMA; MACHADO; PINTO, 2002).

2.3 Interface Musical Digital de Instrumentos

Há muitos anos, os seres humanos aprenderam a registrar seus fatos e atos através de equipamentos eletro/eletrônicos. Contudo, o tamanho do arquivo resultante destas gravações era um dos maiores problemas para o registro de áudio na forma de ondas (LIMA; MACHADO; PINTO, 2002; CAMPOS; LIMA; PINTO, 2003).

As restrições nas capacidades de armazenamento e transmissão quase inviabilizaram a difusão da computação musical, pois, áudio digitalizado é um pacote de informações sonoras digitalizadas (ondas sonoras complexas) e frequentemente um arquivo de áudio simples ocupava todo espaço disponível em uma determinada mídia (HAANDEL, 2009; BRIA; SOSA; THOMAS, 1995; BEZERRA; NOGUEIRA, 2006).

A compreensão do processo sonoro possibilitou ao homem a criação de uma ferramenta capaz de produzir sons artificialmente, o sintetizador, provocando uma significativa redução no tamanho dos arquivos de áudio, uma vez que milhares de sons podem ser simulados a partir de sintetizadores (CASTRO et al., 2004; MILETTO et al., 2004). Um sintetizador é uma ferramenta eletrônica para geração artificial de sons através de técnicas diversas, tais como:

- Manipulação direta de correntes elétricas, neste caso o sintetizador é denominado analógico;
- Leitura de arquivos binários, ou, manipulação matemática de valores discretos, neste caso o sintetizador é denominado digital.

A criação de sintetizadores e de outras ferramentas musicais gerou um crescente interesse pela chamada *música eletrônica*, uma vez que com estes equipamentos um único músico conseguia executar apresentações que antes demandavam a presença de vários músicos. Porém, para conseguir realizar boas apresentações com estes recursos, o músico teria de conhecer alguns fundamentos de eletrônica e de configuração dos equipamentos, em função do complexo sistema de interligação existente. Somado a este fato, ainda existia o problema de que, apesar da grande

frequência de utilização dos sintetizadores, não havia um padrão de intercomunicação entre instrumentos de diferentes fabricantes (LIMA; MACHADO; PINTO, 2002; ZUBEN, 2004).

A não existência de um padrão de intercomunicação fez com que a indústria fonográfica da época não conseguisse absorver as novas tecnologias, já que cada fabricante elaborava seu próprio protocolo de comunicação. Este problema foi resolvido com a criação do padrão MIDI - *Music Interface Digital Instruments (Interface Musical Digital de Instrumentos)* (LIMA; MACHADO; PINTO, 2002; JUNIOR, 2007; HERRERA, 2002).

O padrão MIDI é um conjunto de especificações utilizadas por fabricantes de instrumentos musicais, que permitem a intercomunicação de diversos instrumentos, possibilitando composições sonoras complexas. As especificações MIDI, também denominadas *protocolo MIDI*, são divididas em cinco itens (SILVA, 2007):

- Interface MIDI, que é o equipamento capaz de permitir a comunicação entre dois ou mais componentes diferentes do sistema MIDI, com o uso de conectores padrões. Sistema MIDI é a denominação dada ao conjunto de itens utilizados para criação de música no padrão MIDI, como por exemplo, instrumentos, sequenciadores e placas de som;

- Conectores MIDI, são conectores DIN de cinco pinos [3], dos quais apenas os pinos 2, 4, 5 são utilizados para transmissão de dados;

- Mensagens MIDI, ou comandos MIDI, são as informações acerca de configuração, comandos ou notas musicais trocadas entre dispositivos que compõem um sistema MIDI;

- Cabos MIDI, é o cabeamento utilizado para transmissão de dados entre dispositivos MIDI, composto de três condutores, um condutor para enviar dados, um condutor para receber dados e um condutor utilizado como referência de ligação. O comprimento ideal destes cabos é de até 1,6 metros;

- Dispositivo MIDI, é todo e qualquer equipamento capaz de enviar e interpretar o padrão MIDI, como por exemplo, sintetizadores e módulos de som.

É importante salientar que em um sistema MIDI, durante a troca de informações, as mensagens enviadas contêm apenas comandos que deverão ser executados pela interface destino, como por exemplo, ativar ou desativar notas, alterar o volume de execução de uma nota e trocar de instrumento.

[3]Conectores DIN são modelos de conectores normalmente principalmente utilizados em equipamentos de áudio e que tiveram sua utilização expandida para periféricos de microcomputador, como por exemplo *mouse* e teclado

É nesta especificidade que se concentra um dos motivos da difusão do protocolo MIDI como ferramenta para codificação musical, pois uma vez que como não há tráfego de informação sonora, cada arquivo musical demandará no máximo de alguns *Kilobytes* para seu armazenamento, facilitando os processos de reprodução, gravação e edição musical.

O objetivo do MIDI é o compartilhamento de informação musical. Portanto, um arquivo MIDI pode conter informações acerca de instrumentos, notas musicais, tempo, letra da música (utilizado em karaokês), nomes de instrumentos, nomes de notas e as configurações necessárias a execução destes (MANCINI, 2003; SILVA, 2007; SELFRIDGE-FIELD, 1997).

2.3.1 Padrão de Arquivos MIDI

Para tornar possível a correta interpretação dos comandos enviados pelos mais diversos instrumentos, criou-se o padrão de linguagem estrutural internacional SMF - *Standard MIDI Files (Padrão de Arquivos MIDI)*. As informações contidas em um arquivo escrito na linguagem SMF registram ações que poderão ser compreendidas por qualquer interpretador MIDI. Interpretador MIDI é a denominação dada ao *hardware* ou *software* capaz de reproduzir arquivos *.mid* (ANDERTON, 1987; BRAUT; BROWN, 1994; SELFRIDGE-FIELD, 1997; STANDARD, 1999).

A ordem de escrita das informações de um arquivo SMF é a mesma em que a melodia será executada, respeitando-se os cabeçalhos e os códigos obrigatórios do padrão MIDI. Considerando que uma melodia nada mais é do que a interpolação de *figuras de pausa* e *figuras de nota*, a maior parte de um arquivo escrito em SMF é composto por sequências de ativação e desativação de notas. Portanto, para que seja possível executar uma nota musical, um arquivo escrito em SMF deverá conter as seguintes informações (ASSOCIATION et al., 1996; LIMA; MACHADO; PINTO, 2002):

1. Indicação da nota a ser tocada, como por exemplo o código *3C* utilizado para ativar a nota $Dó_5$;
2. Comando de ativação, como por exemplo, o código 90 utilizado para ativar uma nota ;
3. Indicação da altura de repercussão da nota musical, como por exemplo, o código 64 utilizado para indicar a *força* de *mezzo piano*[4];
4. Indicação do tempo de duração da nota, como por exemplo, o código 23 utilizado para indicar uma *semicolcheia*;

[4]Mezzo Piano é a definição musical para uma nota executada com força média

5. Fim da contagem de tempo, marcado pelo código 00;

6. Comando de desativação, como por exemplo, o código 80 utilizado para desativar uma nota;

7. Comando de fim da execução, nos quais normalmente é utilizado o código $2FFF00$.

A execução de um arquivo escrito em SMF é realizada em tempo real, e caso exista alguma informação incorreta no arquivo, desde que esta não seja parte integrante dos códigos essenciais, ela será ignorada até o recebimento de uma nova informação correta (FIGUEIREDO; JÚNIOR; TRAINA, 1997).

2.3.2 Contagem de Tempo

Nos arquivos descritos em SMF, a representação de tempo é realizada através do conceito *Delta Time*. Um *delta time*, de forma semelhante ao PPQ, também tem como base de cálculo o valor de uma semínima. A representação cada PPQ, em linguagem SMF, ocupa o tamanho de 1 *byte* no arquivo, sendo este *byte* utilizado da seguinte forma:

- O MSB é utilizado para determinação da continuidade ou não da contagem de tempo;
- Os outros 7 *bits* são utilizados para informar o tempo de execução da *nota* ou *pausa*.

A utilização do *bit* mais significativo para indicação de continuidade deve-se ao fato de que se todos os *bits* do *delta time* fossem utilizados para representar a contagem de tempo musical, as notas só poderiam ter o tempo máximo de duração que pudesse ser descrito em um *byte* (CAMPOS; LIMA; PINTO, 2003). A utilização do *bit* mais significativo para representação de continuidade permite que, quando uma nota demandar um tempo de execução maior do que o possível de descrever em um *byte* o sistema primeiro lerá todos os *bytes* que contém a informação de tempo referente a esta nota para depois executá-la.

Quando o valor lógico do MSB de um *delta time* é igual a 1, o interpretador MIDI identifica que existirão outros *delta times* em sequência a considerar para a contagem de tempo. Já quando o valor lógico do MSB for igual a 0, o interpretador MIDI encerra a contagem neste *byte*. Para fins de exemplificação, sendo $\mathbf{P}$ um vetor de 1 *byte*, no qual, $\mathbf{P_8}$ é o MSB e $\mathbf{P_1}$ o LSB, se $\mathbf{P}$ corresponder a um *delta time*, considerando os 4 casos exemplo, apresentados na Tabela 11, se a primeira informação recebida for o caso 2, seguido do caso 1 a duração da *nota* ou *pausa* será de uma semínima, uma vez que esta duração é calculada a partir da soma dos valores representativos de cada *bit* ativo, com exceção do MSB. Neste caso a soma será 256 +

128 + 64 + 32 = 480. Já se a informação recebida for o caso 3, seguido do caso 4 a duração será de uma *colcheia*, cujo cálculo é 128 + 64 + 32 + 16 = 240 (LIMA; MACHADO; PINTO, 2002; CAMPOS; LIMA; PINTO, 2003).

Tabela 11: Exemplos de *delta time*

Caso	P_8	P_7	P_6	P_5	P_4	P_3	P_2	P_1
1	0	1	1	0	0	0	0	0
2	1	0	0	0	0	0	1	1
3	1	0	0	0	0	0	0	1
4	0	1	1	1	0	0	0	0

Existe ainda a possibilidade da realização da contagem de *delta times* através do padrão *SMPTE timecode*, comumente utilizado para sincronização de áudio e vídeo em filmes. Contudo, este padrão de contagem não será utilizado neste trabalho.

2.3.3 Estrutura da Linguagem SMF

Além da contagem de tempo, outra característica importante do sistema MIDI é a diferenciação entre *Track* e *Canal*. Para que seja possível a reprodução com fidelidade de uma música gravada, um formato musical tem de possuir a capacidade de tocar todos os 16 canais de áudio considerados mundialmente no mercado musical (LIMA; MACHADO; PINTO, 2002; CAMPOS; LIMA; PINTO, 2003).

Cada canal de áudio equivale a um grupo de instrumentos diferentes, sendo o canal 10 normalmente reservado para os instrumentos de percussão, ou seja, se os 16 canais estiverem ativos ao mesmo tempo é possível ouvir vozes[5] de 16 instrumentos diferentes ao mesmo tempo. Já os *Tracks* são as linhas de gravação usadas para armazenar as composições realizadas (CAMPOS; LIMA; PINTO, 2003).

Com relação ao tipo de arquivo, existem 3 modelos de arquivo que podem ser utilizados para escrita em SMF, o MIDI-0, o MIDI-1 e o MIDI-2. A identificação de qual modelo será utilizado na linguagem SMF é realizada através da configuração de 4 *bits* do cabeçalho principal (SELFRIDGE-FIELD, 1997; LIMA; MACHADO; PINTO, 2002; CAMPOS; LIMA; PINTO, 2003). Os tipos de configuração existente são:

- *0000*, quando o arquivo for do tipo MIDI-0;
- *0001*, quando o arquivo for do tipo MIDI-1;
- *0010*, quando o arquivo for do tipo MIDI-2.

[5]Em Música, o som produzido por um instrumento é chamado de voz

A diferença entre cada um destes modelos constitui-se basicamente no número de *tracks* que podem ser inseridos no arquivo e na sua forma de execução pelo dispositivo MIDI. Portanto, a definição de cada um dos três modelos pode ser realizada da seguinte forma:

- MIDI-0, que é muito utilizado para comunicação entre instrumentos, dada a sua facilidade de leitura, uma vez que toda informação fica armazenada em apenas um *track*;
- MIDI-1, muito utilizado em estúdios de gravação e processamento de áudio. Neste modelo, a melodia é gravada no mesmo formato em que foi idealizada, ocupando quantos *tracks* forem necessários. Com relação a leitura do modelo ela é feita de forma sequencial;
- MIDI-2, que não é um padrão, mas sim uma extensão do formato MIDI-1. A diferença entre o formato MIDI-1 e o formato MIDI-2 é a de que, no formato MIDI 2 existe a opção da execução de alguns trechos de forma paralela e outros de forma sequencial.

Estas definições se tornam fundamentais para compreensão e interpretação de qualquer arquivo escrito no padrão SMF, pois contemplam a sua ordem estrutural. Para que seja possível a execução de arquivos *.mid* em máquinas MIDI virtuais, emuladas em ambiente computacional, os arquivos SMF são escritos no formato hexadecimal, de forma continua e respeitando a seguinte ordem:

1. *Header chunck* ou cabeçalho principal, é o trecho que identifica que o arquivo foi escrito no padrão SMF, apresentando o tamanho do arquivo, o número de *tracks* e o modelo MIDI utilizado;
2. *Track chunck* ou faixa, é o parte do arquivo que contém a informação a ser executada, ou seja, as notas musicais, os instrumentos, a assinatura de tempo, o metrônomo, etc.

Todo e qualquer arquivo no padrão SMF tem de conter estas duas partes e ambas as partes possuem campos fixos, tais como os indicadores de término de mensagem e o detalhamento do tipo de *Chunck* enviado, e campos variáveis que servem para registrar as informações musicais.

Em termos de abrangência, o padrão SMF prevê codificações para cada uma das 128 notas musicais e para cada um dos 128 instrumentos musicais internacionalmente aceitos, além das variações previstas para os instrumentos de percussão (SELFRIDGE-FIELD, 1997; STANDARD, 1999).

2.3.3.1 Cabeçalho Principal

A codificação no padrão SMF é estruturada de forma que qualquer erro no *header chunck* invalide a execução da composição da música registrada no arquivo. Portanto, além do código inicial de identificação obrigatório um *header chunck* deve possuir no máximo 14 *bytes* de comprimento, distribuídos estruturalmente da seguinte forma:

$$< Header\ Chunck > = < chunck\ type > <length > <format> <ntrks> <division>$$

A definição de cada um dos campos constituintes do *header chunck* é:

1. *Chunk Type* ou método, Mthd, contém o código que indica que o arquivo foi feito no padrão SMF;
2. *Length*, contém o código que indica o número de *bytes* contidos no restante do cabeçalho, ou seja, o total de *bits* a serem lidos para compreensão dos trechos *format*, *ntrks* e *division*;
3. *Format*, indica qual é o modelo de arquivo utilizado;
4. *Ntrks*, contém o código que indica o número total de *tracks* contidos no arquivo;
5. *Division*, contém a representação, em *delta time*, da quantidade de pulsos por semínima utilizada no arquivo.

Um exemplificação do preenchimento de cada um dos campos que compõem o *header chunck* é apresentado na Tabela 12. Neste exemplo é apresentado um cabeçalho principal com 6 *bytes*, utilizando o *formato* MIDI-0, com um *track* e 96 pulsos por semínima.

Tabela 12: Detalhamento do *Header Chunck*, com exemplificação (LIMA; MACHADO; PINTO, 2002)

estrutura	chunck type	length	format	ntrks	division
tamanho	4 bytes	4 bytes	2 bytes	2 bytes	2 bytes
exemplo	4D546864	00000006	0000	0001	0060

2.3.3.2 Faixa

Por conter todas as informações necessárias para execução de uma melodia, um *track chunck* é dividido em segmentos rígidos e flexíveis. Os segmentos rígidos contém as informações necessárias a interpretação do arquivo. Os segmento flexíveis contêm as informações acerca da melodia, inseridas pelo usuário, como por exemplo, notas musicais, instrumentos e duração.

A presença de erros nos segmentos rígidos invalida o arquivo, mas erros nos segmentos flexíveis invalida apenas o trecho aonde o erro está localizado, fazendo com que a máquina aguarde até o recebimento de um trecho válido. Estruturalmente, uma faixa ou *track chunck* dividi-se em:

$$\boxed{<\textit{Track Chunck}> = <\textit{chunck type}> <\textit{length}> <\textit{MTrk event}>}$$

A definição de cada um dos campos constituintes do *track chunck* é:

1. *Chunck Type* ou cabeçalho secundário, que contém o código que indica que este *track* é do tipo *track chunck*;

2. *Length*, que é a soma de todos os *bytes* que serão enviados após este campo, incluindo os 3 *bytes* de encerramento $FF2F00_{16}$;

3. MTrk Event, que contém os códigos das configurações a serem lidas ou executadas pelo interpretador.

A estrutura do *MTrk event* é composta por duas partes:

$$\boxed{<\textit{MTrk Event}> = <\textit{delta time}> <\textit{event}>}$$

Todos os intervalos de um arquivo escrito no formato SMF, são marcados por um *delta time*. Quando o *delta time* tem valor 00_{16} significa que o interpretador não precisará aguardar para processar o próximo comando. Já quando o valor do *delta time* é diferente de 00_{16}, o interpretador deverá ler todos os *delta times* para obter o valor do tempo necessário de espera para o processamento do próximo comando. Já o trecho *event* consiste no seguinte grupo de informações:

$$\boxed{<\textit{Event}> = <\textit{MIDI event}> \mid <\textit{sysex event}> \mid <\textit{meta event}>}$$

O campo *MIDI Event* é destinado a inserção dos códigos referentes as mensagens do canal, tais como ativar ou desativar notas. O campo *Sysex Event* é reservado para os códigos referentes a mensagens exclusivas de equipamento. O campo *Meta Event* destinado a inserção dos códigos referentes as informações denominadas *Não MIDI*, tais como a assinatura de tempo. É importante salientar que, apenas alguns equipamentos de áudio conseguem interpretar todos os códigos possíveis para *Meta Event*, mas, todos os equipamentos interpretam os códigos de *Meta Event* básicos, como por exemplo, a assinatura de tempo.

O tamanho dos campos *MIDI event*, *Sysex Event* e *Meta Event* dependerá única e exclusivamente da informação que será enviada através deles. O campo *Sysex event* deve sempre ser iniciado com o valor $F0_{16}$ e encerrado com o valor $F7_{16}$. Contudo, este campo não é uma estrutura obrigatória na mensagem, tendo em vista que o mesmo trata-se de comunicação exclusiva entre equipamentos.

Todo *meta event* deve ser iniciado com o valor FF_{16}, sendo a estrutura de um *meta event* definida através do seguinte formato:

'FF' < *type* > <*length*> <*text*>

Alguns dos principais *meta events* são apresentados na Tabela 13. Os campos cujo valor foi definido com a palavra variável, dependem única e exclusivamente do arquivo que esta sendo produzido, como por exemplo, a letra da música.

Tabela 13: Exemplo de *Meta Events* (LIMA; MACHADO; PINTO, 2002)

type	**descrição**	**tamanho (em bytes)**
51	metrônomo	3
58	formula do *compasso*	4
04	nome do instrumento	variável
05	letra da música	variável

2.3.4 Notação Musical

Um protocolo musical é considerado válido quando abrange o maior número de componentes musicais possíveis, que neste caso são as notas e os instrumentos. A definição de abrangência em relação as notas musicais é ligada diretamente ao tipo de escala tonal aceita pelo formato de codificação.

A *escala tonal* é a regra que descreve a forma como as notas são distribuídas, em termos de frequência, dentro de um intervalo pré-definido. Dentre os vários padrões existentes de escala tonal as escalas mais conhecidas são a escala oriental e a escala ocidental. A escala oriental possui cinco notas principais. Já a escala ocidental possui sete notas principais. Destas duas escalas surgiu a chamada escala temperada, que consiste na divisão exata de intervalos, entre as sete notas e os cinco acidentes musicais existentes na escala ocidental, constituindo-se assim doze notas principais. Os acidentes musicais podem ser ascendentes ou descendentes. O acidente ascendente, simbolizado por #, representa o *Sustenido* que é um efeito musical capaz de tornar o som mais agudo. O acidente descendente, simbolizado por *b*, representa o *Bemol*,

que é um efeito musical capaz de tornar a nota mais grave. A Tabela 14 apresenta as 12 notas existentes na escala temperada.

Tabela 14: Notas da música ocidental

Nome	**Símbolo**
Dó	C
Dó#/Réb	C#/Db
Ré	D
Ré#/Mib	D#/Eb
Mi	E
Fá	F
Fá#/Solb	F#/Gb
Sol	G
Sol#/Láb	G#/Ab
Lá	A
Lá#/Sib	A#/Bb
Si	B

Considerando que uma nota musical pode ser definida pelo número de repetições de uma determinada forma de onda no espectro sonoro audível, durante um segundo, e que este número de repetições/segundo é definido como frequência do sinal, cuja grandeza de medida é o *Hertz*, (*Hz*), a relação entre uma nota e outra na escala temperada é a apresentada na Equação 6, na qual F_{PN} é a frequência da próxima nota e F_{NA} é a frequência da nota atual.

$$F_{PN} = F_{NA} \times 2^{1/12} \tag{6}$$

Esta relação, torna matematicamente possível a classificação de todas as 128 notas musicais aceitas pelo padrão MIDI, derivadas das 12 notas principais da escala temperada. Estas 128 notas são estruturadas por frequência, em grupos de 12 notas denominados oitavas. A estruturação de uma oitava é a apresentada na Figura 21, na qual são identificadas as correspondências entre # e *b*.

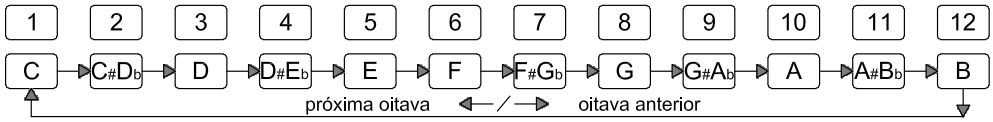


Figura 21: Formato de alocação das notas musicais em oitavas

Toda oitava começa com a nota C e termina com a nota B, com exceção da décima primeira oitava (oitava 9), que termina na nota G. A razão entre a frequências de uma oitava e a oitava seguinte é $\frac{1}{2}$. O mapa completo de codificação de notas do protocolo MIDI é

apresentado na Tabela 15, onde são indicados, por oitava os códigos utilizados na linguagem SMF para de cada uma das 128 permitidas (SELFRIDGE-FIELD, 1997; STANDARD, 1999; LIMA; MACHADO; PINTO, 2002; CAMPOS; LIMA; PINTO, 2003).

Tabela 15: Mapa de notas MIDI (LIMA; MACHADO; PINTO, 2002)

oitava	**C**	**C#**	**D**	**D#**	**E**	**F**	**F#**	**G**	**G#**	**A**	**A#**	**B**
-1	0	1	2	3	4	5	6	7	8	9	10	11
0	12	13	14	15	16	17	18	19	20	21	22	23
1	24	25	26	27	28	29	30	31	32	33	34	35
2	36	37	38	39	40	41	42	43	44	45	46	47
3	48	49	50	51	52	53	54	55	56	57	58	59
4	60	61	62	63	64	65	66	67	68	69	70	71
5	72	73	74	75	76	77	78	79	80	81	82	83
6	84	85	86	87	88	89	90	91	92	93	94	95
7	96	97	98	99	100	101	102	103	104	105	106	107
8	108	109	110	111	112	113	114	115	116	117	118	119
9	120	121	122	123	124	125	126	127				

2.3.5 Instrumentos Musicais

A tabela de instrumentos musicais no SMF engloba desde instrumentos convencionais, como violão e piano até efeitos musicais, como barulhos de chuva e vento. Cada instrumento, neste caso *voz*, é descrito por um código numérico sempre iniciado por zero, cujo tamanho é um *byte*. Os códigos utilizados variam da primeira *voz* que é o piano, 0000000_2, até a última *voz* que é o som de tiro, 01111111_2. Com relação a distribuição por canais, os instrumentos MIDI podem ser distribuídos no canais MIDI de 0 a 9 e de B_{16} a F_{16}. O canal A_{16} é normalmente reservado para instrumentos de percussão. Esta separação entre os instrumentos de percussão e os demais instrumentos deve-se ao fato de que instrumentos de percussão normalmente são capazes de emitir apenas uma nota musical, sendo fortemente marcados pela variação rítmica e não tonal, como por exemplo, o som das maracas e do *side kick* (SELFRIDGE-FIELD, 1997; STANDARD, 1999; LIMA; MACHADO; PINTO, 2002; CAMPOS; LIMA; PINTO, 2003).

Em termos de divisão sonora os instrumentos MIDI podem ser classificados em 16 classes estruturadas em codificação por faixa. A definição de cada classe de instrumentos MIDI é:

- Pianos, faixa de 0 a 7, composta pelos instrumentos acústicos e elétricos de teclas, tais como cravo e clavicórdio;
- Percussão Cromática, faixa de 8 a 15, composta pelos instrumentos de percussão que permitem afinação, como xilofone, caixa de música e vibrafone;

- Orgãos, faixa de 16 a 23, compostas pelos instrumentos musicais cujo som é produzido pela passagem de vento por tubos internos, tais como acordeon e bandoneon
- Guitarras, faixa de 24 a 31, que contempla todos os tipos de guitarra, como por exemplo, a elétrica, a abafada e a de cordas de *nylon*;
- Baixos, faixa de 32 a 39, que contempla todos os tipos de baixo, como por exemplo, o *thump*, o *pop* e o analógico;
- Cordas, faixa de 40 a 47, que contempla as tonalidades produzidas por instrumentos clássicos de corda, tais como harpa, viola e violino;
- Orquestra Sinfônica, faixa de 48 a 55, que contempla os timbres executados de forma similar a uma orquestra;
- Metais, faixa de 56 a 63, composta pelos instrumentos de sopro metálicos, tais como trombone e trompete;
- Palhetas, faixa de 64 a 71, composta pelos instrumentos de sopro cujo som é produzido pela vibração de palhetas no ato da passagem de ar, tais como saxofone e oboé;
- Flautas, faixa de 72 a 79, composta pelos instrumentos cujo som é produzido através da passagem de ar por orifícios ou desníveis, como por exemplo o flautim e a ocarina;
- Solos sintéticos, faixa de 80 a 87, composta por instrumentos capazes de produzir sons das maneiras mais diversas o possível, como por exemplo, a *calliope* que tem como base de geração sonora o funcionamento de uma locomotiva;
- Fundos sintéticos, faixa de 88 a 95, que engloba os sons criados artificialmente, muito utilizados em vinhetas e propagandas;
- Efeitos parte 1, faixa de 96 a 103, que contempla os efeitos de fenômenos naturais como chuva de gelo e tilintar de cristais;
- Instrumentos étnicos, faixa de 104 a 111, que contempla os instrumentos característicos de um povo ou nação, como por exemplo, o *shamisen*;
- Percussão não cromática, faixa de 112 a 119, composta pelos instrumentos de percussão que não permitem afinação direta, mas que possuem capacidade de produzir notas diferentes como o agogô e o *taiko*;

- Efeitos parte 2, faixa de 120 a 127, segunda parte de efeitos especiais abordando sons curiosos como o de um telefone tocando e o barulho de um tiro de arma de fogo;

Deve ser observado o fato de que no ato da escolha de um determinado instrumento dentro do protocolo MIDI, o que está sendo realizado é uma solicitação para que o interpretador execute a nota ou o grupo de notas da mesma forma que seria executado naquele determinado instrumento. Por conseguinte, nem todas as notas são audíveis nos 128 instrumentos MIDI. Outro fato importante é o de que os códigos utilizados para os instrumentos de percussão são os mesmos já utilizados por outros instrumentos, fazendo com que o sistema compreenda que se o código foi utilizado no canal A, deverá ser emulado o instrumento de percussão.

2.3.6 Qualidade Acústica do SMF

As principais críticas relacionadas ao padrão SMF, e consequentemente aos arquivos de extensão *.mid*, referem-se a qualidade sonora. O grande motivo destas reclamações é que apesar do sistema prever toda a gama possível de instrumentos e notas musicais, boa parte dos módulos timbrais e placas de som não possuem as características necessárias para reprodução de tons com a mesma qualidade dos instrumentos acústicos. Esta critica é tão latente, que durante muito tempo, apesar de ser um protocolo extremamente robusto, o MIDI só era utilizado para aplicações de pequeno porte, como por exemplo, melodias para telefones celulares e fundo musical em jogos de videogame (ROSSUM; JOINT, 1995; MANCINI, 2003; IAZZETTA; KON, 1998; SILVA, 2007).

Algumas soluções para este tipo de problema foram idealizadas, sendo grande parte concentrada em *software* devido ao alto custo da aquisição de boas placas de som e módulos timbrais, como por exemplo:

- Aquisição de programas com sintetização de timbres em boa qualidade;
- Utilização de *SoundFonts*, apenas para quem possui placas da *Sound Blaster* ou similares;
- *Download* de sintetizadores, como o XG da *Yamaha*, que conseguem fazer uma aproximação sonora de timbres.

2.4 Considerações Finais do Capítulo

No decorrer deste capítulo foram apresentadas as diferenças entre som, barulho e música, pontuando os conceitos musicais relevantes para compreensão do protocolo MIDI e as definições deste protocolo. Também foi explicitado que o protocolo MIDI, que será utilizado como

codificação para as melodias geradas, armazena apenas informações musicais restringindo a qualidade sonora ao *hardware*/*software* que o executa. Foram definidas ainda as simbologias, apresentadas na Tabela 14 e Figura 21, que serão utilizadas para referenciar a notação musical ao longo deste trabalho. O capítulo seguinte apresenta alguns trabalhos relacionados aos objetivos desta dissertação.

Capítulo 3

TRABALHOS RELACIONADOS

ESTE capítulo, organizado em quatro seções, apresenta alguns trabalhos relacionados aos objetivos propostos, de forma a validar e endossar os elementos desenvolvidos no decorrer desta dissertação. Na Seção 3.1 são apresentadas algumas aplicações diversificadas implementadas em *software*, que exploraram a aplicação de CAs como alternativa para a resolução de problemas complexos. Na Seção 3.2 são apresentadas implementações de CA em *hardware* como alternativa para a redução de custo computacional e tempo de resposta em tempo real. A Seção 3.3 apresenta algumas aplicações que envolvem a utilização de CAs para composição automática de melodias, corroborando para o desenvolvimento da ideia central deste trabalho. Na Seção 3.4 são sintetizadas algumas considerações finais acerca dos trabalhos citados.

3.1 Aplicações em Software

Por permitirem a modelagem e a análise de dinâmicas extremamente complexas, através da emulação do comportamento coletivo, os CAs possuem aplicabilidade nas mais diversas áreas de pesquisa, tendo uma excelente divulgação e aproveitamento em *softwares* diversos, desenvolvidos com objetivo de demostrar resultados para questões que envolvam análise de demandas complexas de evolução. Esta seção mostrará três exemplos diversificados de utilização de CAs para este fim, de forma a proporcionar ao leitor a compreensão do alcance e aplicabilidade desta linha de pesquisa no auxílio à resolução de problemas complexos.

3.1.1 Análise e Controle de Epidemias

Este trabalho, (MIRANDA et al., 2008), foi resultante da necessidade de uma proposta de modelagem de CA para análise comparativa entre dois métodos para coleta de moluscos, que são:

- Puça, no qual são realizadas capturas sucessivas de moluscos em um determinado ponto de coleta, utilizando redes de *nylon* com 40 *cm* de profundidade, de formato similar a um filtro para coar café;

- Concha Metálica, no qual são realizadas capturas sucessivas de moluscos em um determinado ponto de coleta, utilizando conchas metálicas perfuradas com 25 *cm* de profundidade, de formato similar ao de uma peneira.

Esta coleta refere-se ao período entre Novembro/06 e Outubro/07, quando foi realizado um levantamento malacológico na praia da Carne de Vaca (Goiana/PE) como parte integrante de uma pesquisa sobre a reprodução e urbanização da esquistossomose, com base em inquéritos epidemiológicos.

A modelagem de CAs foi utilizada para geração de cenários para os dois processos, de forma a fornecer dados acerca da eficácia de uma técnica sobre a outra em determinadas situações de coleta. Como características o processo utilizou o modelo de *vizinhança* de *Moore* e um *grid* bidimensional, organizado de forma a simular uma coleção hídrica de 2.000 moluscos, distribuídos aleatoriamente, conforme uma distribuição normal, onde cada célula representa um molusco capaz de continuar parado, quando não houver outro em sua proximidade, ou movimentar-se de forma aleatória até molusco mais próximo.

Como não havia interesse na classificação de crescimento populacional, os estados de nascimento e morte não foram utilizados. Para obtenção do conjunto de regras foram elaboradas as sete variáveis, descritas na Tabela 16.

Tabela 16: Variáveis utilizadas no processo

Variável	**Definição**
N_1	número de moluscos coletados com puçá
N_2	número de moluscos coletados com concha
T_a	total de vizinhos acima de um molusco
T_b	total de vizinhos abaixo de um molusco
T_e	total de vizinhos a esquerda de um molusco
T_d	total de vizinhos a direita de um molusco
T_v	total de vizinhos de um molusco

O número total de iterações foi configurado como dez, e, no decorrer da emulação das técnicas de Puça e Concha Metálica foram simuladas coletas em diferentes pontos do *grid*, escolhidos de forma aleatória. A organização da vizinhança da célula X é apresentada na Figura 22. Nesta estrutura, cada célula vizinha é representada como C_n, onde n representa o índice de referência para deslocamento.

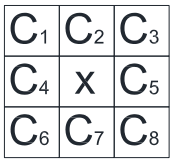


Figura 22: Organização da vizinhança de *Moore* para a célula X

As variáveis configuradas permitiram a geração de um conjunto de nove regras, apresentadas a seguir, que permitem o acompanhamento dos possíveis locais de movimentação dos moluscos durante a coleta, e consequentemente, a averiguação dos possíveis desvios de dados acerca do quantitativo populacional:

1. Se $T_v = 8$, não há espaço para deslocamento e o molusco continua na posição em que está;

2. Se $T_v = 0$, há 50% de chances do molusco se deslocar para a posição C_2 ou para a posição C_7;

3. Se $(T_e = T_d = 1) \wedge (T_a = T_b = 0)$, haverá deslocamento para C_7 e aferição da próxima regra;

4. Se $(T_e = T_d = 0) \wedge (T_a = T_b = 1)$, se houver disponibilidade haverá deslocamento para C_5, se não aferição da próxima regra;

5. Se as regras 4 e 5 não forem satisfeitas, haverá 50 % de probabilidade de deslocamento para as posições C_3 ou C_8, se estas não estiverem ocupadas;

6. Se $(T_e = T_d = 1) \wedge (T_a = T_b = 1)$, se houver disponibilidade, haverá deslocamento para C_8, e aferição da próxima regra;

7. Se $T_v = 4$, $T_e = T_a = T_d = 2$, ou seja, se existirem moluscos nas células C_1, C_3,C_6 e C_8, haverá deslocamento para C_1;

8. Se as regra 6 e 7 não forem satisfeitas, o deslocamento poderá ser realizado para qualquer espaço vazio;

9. Se $T_v = 1$, 3, 5, 6 ou 7, o deslocamento poderá ser realizado para qualquer espaço vazio;

Os resultados obtidos no decorrer das simulações, totalizando 156 moluscos capturados no método Puçá e 23 no método Concha Metálica, comprovaram a eficácia do método Puçá para coletas gerais, fornecendo dados reais que poderão ser utilizados em futuras pesquisas.

3.1.2 Evolução de Câncer

Este trabalho, (MARTINS; GROSSA, 2010), utilizou um modelo de CA bidimensional juntamente com o algoritmo *Hoshen* e *Kopelman* (HOSHEN; KOPELMAN, 1976), de análise de *clusters* em fenômeno de percolação, para modelar os aspectos de crescimento e tratamento quimioterápico de tumores e simular o funcionamento de um sistema de vigilância imunológica, que identifica e elimina agentes patogênicos e células tumorais.

No algoritmo de *Hoshen* e *Kopelman* os *clusters* são identificados por rótulos, ou números, através de algumas decisões, como por exemplo:

- Se um determinado sitio ocupado não possuir vizinho será iniciado um novo *cluster*, ou seja, este receberá um rótulo ainda não utilizado;
- Se um sitio possuir um vizinho com rótulo, a esquerda ou acima, receberá identificação deste;
- Se um sitio possuir dois vizinhos com rótulo, um a esquerda e outro acima, receberá identificação de menor número;

Este processo decisório, aplicado no sistema proposto, determina os passos da evolução do câncer ou da reação do sistema imunológico. O Câncer pode ser definido como um grupo de doenças que possuem como característica o crescimento celular incontrolável. Este crescimento se traduzirá, no corpo do paciente, em tumores ou neoplasias através de um processo chamado tumorigênese. Neste caso, as células atingidas desenvolvem resistência aos processos realizados para inibição de crescimento.

Para pleno desenvolvimento, um tumor necessita de nutrição e consequentemente de um novo abastecimento sanguíneo da região em questão, criando novos vasos sanguíneos (angiogênese). Quando um tumor atinge o estado denominado maligno, os inibidores de crescimento não possuem mais eficiência e o tumor torna-se capaz de invadir e se espalhar nos tecidos próximos (metástase).

Em função da demanda por soluções e explicações, dado o aumento no número de casos, nas últimas décadas foram publicados inúmeros trabalhos conceituando modelos para estudos de tumores. Dentre estes modelos foi escolhido, para o trabalho em questão, o modelo proposto por *Qi et al.* (QI et al., 1993), com base em autômatos celulares com inserção de atraso.

O atraso imposto reflete no intervalo durante o qual o sítio é afetado por suas regras iniciais e no tempo de atualização das variáveis, interferindo na proliferação celular. Já o

crescimento do sistema é realizado através do modelo de *Gompertz*, (FINCH; PIKE, 1996), considerando a metástase e algumas características microscópicas, tais como a proliferação das células cancerígenas, a atividade citotóxica e o mecanismo de pressão interna do tumor. Utilizou-se ainda o número de *clusters* como parâmetro para início da metástase.

No decorrer deste trabalho o comportamento macroscópico foi analisado através de características microscópicas, sendo realizados estudos acerca de alguns aspectos da dinâmica de crescimento de um câncer, da análise da metástase e do efeito da inserção de atrasos no tempo de proliferação, propiciando a utilização deste modelo para estudos complexos acerca da evolução desta doença, conforme objetivo proposto.

3.1.3 Simulação de Tráfego Urbano

Este trabalho, (TAVARES, 2010), teve como objetivo a construção de um simulador de tráfego veicular urbano, capaz de emular as principais características do trânsito atual, como por exemplo, vias de múltiplas faixas, diferentes tipos de veículos, recuos, sinalização, defeitos, acidentes e congestionamentos. Para tal, foi utilizado um CA estocástico bidimensional, no qual cada célula representa o espaço necessário para emulação de um veículo médio popular, que pode assumir dois estados, vazio ou ocupado, em relação aos veículos ao seu redor, simulando assim uma via de tráfego. Cada veículo possui as seguintes propriedades:

- Direção primária, *pd*, que é o trecho contínuo no qual o motorista não altera sua direção ou faixa;
- Direção secundária, *sd*, que é o trecho no qual o motorista muda de direção ou de faixa;
- Limite de velocidade, *vmax*, que indica a progressão máxima dos veículos pelas células a cada iteração, refletindo a média de células por segundo, ou seja, a velocidade do veículo. Este cálculo será realizado a partir da relação ($1 célula/segundo = 20\ Km/h$);
- Indicador de ocupação, *o*, que indica se o espaço está ocupado ou não.

Para indicação da indisponibilidade de célula para trânsito ou final de via, os veículos assumem, respectivamente, os valores das quadruplas (0, 0, 0, 0) e (9, 9, 9, 9), considerando os parâmetros supracitados. Em regiões onde não há possibilidade de alteração de faixa ou de direção, têm-se $pd = sd$.

Cada sentido de deslocamento possui um código *d* relativo ao valor de deslocamento do veículo nas direções dos eixos x e y, conforme Figura 23. Os códigos *d7* e *d3* representam

deslocamentos apenas no eixo y e os códigos *d1* e *d5* representam deslocamentos apenas no eixo x. Os demais códigos representam deslocamentos com pares ordenados dos eixos x e y.

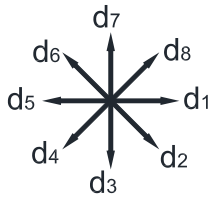


Figura 23: Identificação das direções possíveis para deslocamento de um veículo

Cada direção d representa uma das coordenadas direcionais que são apresentadas na Tabela 17.

Tabela 17: Códigos de deslocamento nos eixos x e y

Código	1	2	3	4	5	6	7	8
x	1	1	0	-1	-1	-1	0	1
y	0	-1	-1	-1	0	1	1	1

A elaboração do sistema proposto contemplou as dez seguintes etapas:

1. Parada de ônibus, que é a etapa na qual são inseridas as configurações de rota de parada de ônibus;

2. Controle de semáforos, que é a etapa na qual são inseridas as configurações de semáforo para controle de cruzamentos, controle de velocidade e paradas;

3. Movimentação veicular, que é a etapa mais importante, na qual é realizada a dinâmica do modelo (controle de velocidade, mudança de rota e de faixas);

4. Remoção veicular, na qual é realizada a remoção dos veículos iniciados com o valor (9, 9, 9, 9);

5. Configuração inicial, na qual são preparados os modelos a partir da obtenção dos parâmetros iniciais necessários. Nesta etapa todas as células do modelo que permitem trânsito estão vazias. Também é nesta etapa que são definidas as funções de densidade de probabilidade para a próxima etapa;

6. Criação de novos veículos, que é a etapa na qual são configurados os veículos que aparecerão no modelo;

7. Registro de veículos, que é a etapa na qual são utilizados dados das células sensores. As células sensores são células dispostas no início e fim de cada trecho de via/quarteirão, com a finalidade de obter e registrar os dados estatísticos sobre a dinâmica do modelo;

8. Geração de incidentes, etapa na qual é possível injetar no sistema bloqueios totais ou parciais das vias impedindo o trânsito durante um determinado intervalo de tempo;

9. Contabilização, etapa na qual são analisados os dados registrados nos sensores objetivando a quantificação do comportamento do sistema;

10. Finalização da simulação.

Cada uma destas etapas possui internamente um conjunto de regras locais que ditam a evolução do sistema. No mapa elaborado, modelado em *MatLab* (MATHWORKS, 1992), foram programados 8 quarteirões de circulação, e os valores de entrada propostos, condizentes com valores reais de tráfego urbano, produziram respostas em conformidade com a Teoria de Fluxo de Tráfego, (PEREIRA, 1988), confirmando que o sistema comportou-se conforme o pretendido, visto o conhecimento empírico do problema.

3.2 Aplicações em Hardware

Devido a complexidade de ligação entre células, que varia em função das dimensões do *grid* e do modelo de conjunto *vizinhança* adotado e das restrições naturais, causadas pela prototipação, tais como a vinculação do *hardware* a resolução de um grupo determinado de problemas e a necessidade do encapsulamento do CA dentro de um equipamento, as implementações de CA em *hardware* são mais raras do que as em *software*. Contudo, as implementações de CA em *hardware* tem se demonstrado extremamente úteis quando há necessidade de aceleração de processamento, resposta em tempo real e redução do custo computacional. Esta seção apresentará 3 exemplos de implementação de CAs em *hardware* que possuem como semelhança a característica de incentivar estudos futuros.

3.2.1 Cérebro Artificial

Este trabalho, (KORKIN et al., 1997), descreve uma arquitetura capaz de evoluir milhares de módulos de uma rede neural, permitindo a simulação de um bilhão de neurônios cerebrais artificiais em tempo real. Este *hardware* denominado *Cellular Automata Machine - Brain Machine* (CBM), (GARIS, 1994), é um componente essencial do projeto *Cellular Automata*

Machine - Brain Project, que tem por objetivo construir, ampliar e desenvolver um cérebro artificial constituído de um bilhão de neurônios.

O CBM é formado por módulos contendo 4096 células organizadas em um *grid* tridimensional, utilizando para tal a plataforma FPGA *XC6264-* 2PG299C da *Xilinix* (XILINX, 2013a). O objetivo deste projeto é criar uma plataforma que possibilite futuras ampliações, de forma a conseguir incluir a quantidade necessária de neurônios artificiais a fim de controlar diretamente um robô em tempo real, configurando-se assim um cérebro artificial.

Inicialmente, dada a estrutura proposta neste projeto, foi necessária a criação de uma estratégia para possibilitar a implementação de grandes quantidades de CAs bidimensionais e tridimensionais. A estratégia determinada foi a da utilização de cerca de 1 *Gb* de memória RAM para armazenar os dados oriundos da arquitetura, considerando que cada estado de um célula ocupa no máximo 2 *bits*.

Uma restrição inicial deste projeto foi a escolha da utilização da ferramenta CAM (Cellular Automata Machine) (TOFFOLI; MARGOLUS, 1987; DOOLEN, 1991). Esta restrição deve-se ao fato de que o CAM-8 só possui a capacidade de atualização de 200 milhões de células por segundo. Apesar de extremamente alta, esta taxa de atualização impossibilitava a implementação da estrutura proposta, uma vez que esta objetivava uma taxa atualização no valor de 105 bilhões de células por segundo. A solução para esta restrição foi a troca do modelo CAM-8 pelo CBM, que possui um maior poder de atualização em função da combinação de autômatos celulares e algoritmos genéticos. O modelo CBM é composto por cinco componentes principais:

- Módulo CA, que é o componente central do CBM. Este componente tem o objetivo é acelerar a velocidade de evolução do cérebro artificial, a partir de massiva atualização paralela de estados das células do *grid*. Este componente consiste em dois *grids* tridimensionais idênticos, totalizando 8192 células ($16 \times 16 \times 16$ cada unidade). A resolução dos problemas de borda no *grid* utilizado é realizada a partir do uso de referenciamento circular, explicitado no Capítulo 1;

- Memória genótipo/fenótipo, é o componente responsável por armazenar as informações de orientação e localização dos neurônios dentro de cada módulo;

- Unidade de análise de *fitness*, é o componente responsável pelo cálculo da evolução dos módulos em função do objetivo a ser alcançado;

- Unidade de algoritmo genético, é o componente que efetua o controle do algoritmo genético a cada medição realizada pela unidade de análise de *fitness*, de forma a selecionar os 10 melhores módulos para a próxima iteração do algoritmo;

- Módulo de interconexão de memória, é o componente que permite a transmissão de dados entre os módulos da arquitetura.

Apesar de a arquitetura proposta apresentar alguns problemas nos primeiros testes, a média obtida de 25 atualizações por segundo foi condizente com os primeiros objetivos propostos para realização de controle de um robô em tempo real. Os resultados parciais oriundos da análise da estrutura desenvolvida permitiram a confirmação de seu funcionamento e da sua aceptabilidade a expansões, possibilitando a previsão de seu aperfeiçoamento através do desenvolvimento da pesquisa.

3.2.2 Evolução de CAs Unidimensionais

Este trabalho, (WEINERT et al., 2007), apresentou uma proposta para implementação comportamental da evolução de CAs unidimensionais utilizando técnicas de desenvolvimento de *hardware* reconfigurável. Sua motivação deriva-se da dificuldade de execução de tarefas de classificação da densidade, que são tarefas computacionais utilizadas para mineração de dados (MARTINS; FYNN; OLIVEIRA, 2011). Seu objetivo é avaliar a utilidade da computação reconfigurável como acelerador de *hardware* para evolução de CAs, em comparação com esta abordagem em *software*.

Para o desenvolvimento do trabalho foram utilizados: a linguagem de descrição de *hardware VHDL*, o dispositivo *FPGA* (*Field Programable Gate Array*) *Cyclone EP1C6Q240C8*, a linguagem de programação *C++* e um computador pessoal. O computador pessoal e a linguagem *C++* foram utilizados na criação e compilação do *software* utilizado para geração de regras de transição. Estas regras são enviadas para a *FPGA* através de comunicação paralela, sendo a execução dos processos implementados na *FPGA* realizada da seguinte forma:

- Um conversor série/paralelo converte os 128 *bits* gerados pelo *software* armazenando-os em um registrador;

- Um decodificador de comandos recebe as instruções contidas no arquivo enviado pelo *software*, que podem as instruções de *reset*, reiniciar *MLS*[1], limpar regra do registrador,

[1] *MLS(Maximum Length Sequence/Sequência de Comprimento Máximo) é um tipo de gerador aleatório de bits que utiliza o máximo de espaço possível disponível, ou seja,* $2^m - 1$ *onde* m = *número máximo de registradores do tipo LFSR*

iniciar a evolução do CA e encerrar regra de transição. Estes comandos são destinados aos demais blocos do sistema proposto;

- Um bloco, denominado evolução do CA, recebe previamente a regra de transição e uma configuração inicial gerada pelo *MLS*, efetuando um número pré-determinado de iterações, neste caso 200. Para tal, no interior deste bloco existem duas máquinas de estados. A primeira máquina de estados controla o número de iterações do CA. A segunda máquina de estados controla a execução de 10.000 CAs unidimensionais, com uma configuração randômica inicial.

- Uma calculadora de precisão verifica se a configuração final é a esperada, através da realização de tarefas de classificação da densidade. Se a resposta for positiva, o valor da regra de acordo com a avaliação é incrementado. Durante as execuções a calculadora contabiliza o número de acessos e ao final divide-o pelo número de rodadas a fim de encontrar a taxa de precisão;

- Um cronômetro registra o tempo total gasto para gerar o comportamento dos 10.000 CAs;

- Um multiplexador seleciona dados entre a taxa de precisão e o tempo de processamento enviando-os para o computador pessoal.

Para avaliação do sistema, foram realizadas cinco experiências. Para cada das experiências realizadas foram gerados 10.000 CAs unidimensionais, de configuração inicial aleatória, que evoluíram durante 200 iterações, utilizando uma única regra conhecida como co-evolução, muito utilizada para tarefas de classificação de densidade (JUILLE; POLLACK, 1998). A configuração do CA proposto neste trabalho é apresentada na Tabela 18, na qual são detalhadas as características utilizadas.

Tabela 18: Configuração do CA proposto

Característica	**Definição**
r	3
grid	unidimensional
número de células	149
transições possíveis	2^{128}
estados possíveis	2
definição dos estados	0 e 1

Com a finalidade de comparar os resultados obtidos, foi elaborado um programa utilizando a linguagem *C++*, compilado sem qualquer tipo de otimização, com as mesmas ca-

racterísticas das tarefas executadas no *FPGA*. Como resultado desta comparação, obteve-se a resposta de que a computação reconfigurável tem um excelente desempenho para este tipo de tarefa.

3.2.3 Geração Automática

Este trabalho, (LIMA; FERREIRA, 2013), propõe a geração automática de autômatos celulares em arquitetura reconfigurável, através do projeto de um sistema digital e de uma interface gráfica amigável, com a finalidade de permitir, de forma simplificada, ao usuário um controle supervisório do sistema através de um computador pessoal, utilizando a plataforma *Atlys Digilent*, a FPGA *Spartan-6 XC6SLX45*, a linguagem de descrição de *hardware Verilog* e o microprocessador *MicroBlaze*. A motivação deste trabalho é o fato de que, por possibilitar análises de dinâmicas complexas, a geração de resultados de um CA em *software* demanda um elevado custo computacional e um grande período de tempo de processamento. Os modelos utilizados foram extraídos de duas implementações:

- *FPGA GHCA*, uma implementação do modelo estocástico do CA de *Greenberg-Hastings*. O modelo *GHCA* é utilizado para simulação da difusão de ondas em um meio ativo (VLASSOPOULOS et al., 2010);
- *FPGA LGCA*, uma implementação para estudo da propagação de ondas sonoras. *LGCA* são modelos de CA utilizados para emular dinâmica em fluídos e gases, como por exemplo, a colisão de partículas (SHAW; COCKSHOTT; BARRIE, 1996; WOLF-GLADROW, 2000).

O sistema projetado divide-se em três partes: a interface usuário/máquina realizada em *software*, o núcleo do CA implementado em FPGA e o processador *MicroBlaze*. A interface do usuário permite a configuração inicial do CA, uma leitura do mapa atual de estados e o controle da operação do *hardware*, utilizando para tal modelos construídos na linguagem *Verilog*. As características presentes no CA proposto para escolha do usuário são:

- Tipo de vizinhança, que poderá ser de *Moore* ou *Neuman*;
- Tamanho do *grid*, limitado a valores múltiplos de 8;
- Dimensão do *grid*, 1D ou 2D;
- Número de *bits* por célula;

- Regra.

Existe ainda a possibilidade de simulação paralela de diferentes CAs unidimensionais, através da configuração de vários *grids*. Neste caso, cada *grid* é regido por sua própria regra. O processador *Microblaze* é utilizado para transferência dos estados atuais das células, via barramento *RS-232*, acessando para tal o núcleo do CA. O núcleo do CA é dividido nos módulos caracterizados como necessários para o controle do CA, de forma que seja possível realizar a inicialização dos CAs e efetuar a leitura, o armazenamento e a análise dos estados atuais de cada célula. Dentro do núcleo do CA existe um módulo, denominado Lógica CA, que não pode ser reconfigurado durante a execução. Este módulo é responsável por armazenar todas as células programadas. Uma especificidade da arquitetura proposta é que cada linha de células é gerada de forma independente, a partir de um trecho de *HDL* diferente, mas todas as linhas são interligadas com suas respectivas vizinhanças no *grid*. Os demais módulos do sistema proposto são utilizados para registro e transferência de dados entre as células, permitindo o correto arranjo de vizinhança, e transferência de dados entre o *software*, o *Microblaze* e o núcleo CA.

Para incrementação do projeto, utilizou-se o *CoDec AC'97* para geração de melodias, de acordo com o mapa de estados do CA. O funcionamento do gerador de melodia consiste no recebimento e análise do mapa de estados de cada linha de células. A análise do mapa de estados consiste na determinação do *bit* de densidade. Esta determinação do *bit* de densidade é realizada através da divisão do número total de células ativas, valor 1, pelo número total de células contidas na linha. O valor, 0 ou 1, encontrado no *bit* de densidade indicará a frequência da nota a ser tocada, através da geração de uma onda quadrada.

Como resultado, comprovou-se que o tempo de processamento do *hardware* proposto não depende do tamanho do *grid*, mas sim da frequência do *clock* e da complexidade de cada célula. Também se comprovou a relação entre aumento de velocidade e tamanho do *grids* para os CAs FPGA-GHCA FPGA-LGCA. Apesar de este trabalho ter se mostrado funcional e flexível, não foi possível evidenciar como foi realizada a divisão de recursos dentro da FPGA, não inferindo qualquer garantia de que as células tenham sido uniformemente distribuídas podendo ter ocorrido algum grau de degradação de desempenho ou desperdício de recursos.

3.3 Música Algorítmica

A utilização de CAs como base de aquisição de dados para geração musical, principalmente em *software* tem se tornado uma alternativa recorrente para geração de música aleatória. Esta

difusão baseia-se na diversidade de resultados que podem ser gerados a partir do aproveitamento de dados coletados no decorrer de uma ferramenta de inteligência artificial em combinação com os métodos clássicos de produção musical. Esta seção traz alguns destes projetos, de forma a apresentar os tipos existentes de modelagens para geração de música algorítmica com base em ferramentas de inteligência computacional.

3.3.1 AntMusic

Este projeto, (GUÉRET; MONMARCHÉ; SLIMANE, 2004), apresenta uma proposta para geração de música algorítmica através da simulação da movimentação de formigas artificiais ao longo de um mapa. Para tal, projetou-se um grafo no qual cada vértice representa um MIDI *event* e cada aresta representa uma possibilidade de transição entre notas. Durante a inicialização do sistema proposto, as formigas artificiais são alocadas nos vértices do grafo. Ao longo da evolução do sistema, as formigas se movimentam pelo grafo escolhendo as arestas nas quais depositarão feromônio. Quanto mais formigas escolherem uma determinada aresta, maior será a quantidade de feromônio nesta aresta e, por conseguinte, maior será a probabilidade de outras formigas também depositaram feromônio nesta aresta. Os deslocamentos realizados pelas formigas para mudança de vértices irão compor, ao longo da evolução do sistema, a melodia.

As regras de transição para que uma formiga se desloque de um vértice para outro são baseadas no desejo de transição da formiga e na distância entre o vértice atual e o vértice desejado. Neste projeto a distância entre vértices corresponde ao número de semitons entre duas notas. Ao longo do tempo o percentual de desejo de transição incentivará as formigas a escolherem notas próximas a sua posição atual, determinando as notas que irão compor a melodia. Para controle do estilo musical é utilizado um parâmetro denominado β. Quando o valor de β é alto, a movimentação das formigas será com base no valor do desejo de transição. Quando o valor de β é baixo, a movimentação será realizada a partir da quantidade de feromônio contido em cada aresta. Um dos detalhes deste projeto é que, de forma que similar a um ambiente real, o feromônio depositado pelas formigas evapora lentamente com o passar do tempo. Para inicialização do sistema, o usuário deverá configurar nove parâmetros, que são a base fundamental dos processos decisórios para movimentação das formigas. Estes parâmetros são:

- Instrumento, conforme padrão MIDI;
- Escolha do intervalo de notas, conforme padrão MIDI;

- Comprimento máximo da melodia, que poderá variar de 1 a 25;
- Volume das notas;
- Tempo de duração de execução das notas;
- Parâmetro de evaporação;
- Quantidade de feromônio por formiga;
- Valor do parâmetro β;
- Tipo de formiga, que pode ser configurado em silêncio e composição.

A estrutura proposta permite também a utilização de alguns parâmetros de forma a inferir maior complexidade das melodias compostas. Um dos parâmetros que pode ser alterado é o número de instrumentos utilizados na composição musical. Este acréscimo de instrumentos é realizado através da construção de outro grafo idêntico ao existente com a determinação da quantidade de formigas artificiais por grafo. Um arquivo *.mid* pode ser utilizado para inicializar o sistema, de forma a limitar o tamanho do grafo. Assim, é possível obter uma melodia similar a contida no arquivo de inicialização.

Os resultados obtidos demonstraram a influência dos parâmetros de evaporação e de desejo de transição no produto musical obtido. Também foi evidenciado que a razão entre os tipos de formiga, formiga silêncio e formiga composição, também influencia diretamente no resultado obtido. Como conclusão, observa-se que a utilização de algoritmos de colônia de formigas, apesar de alguns erros e pontos a serem melhorados, demonstrou-se uma excelente alternativa para composição de música algorítmica, figurando este trabalho como uma proposta inovadora nesta área de pesquisa.

3.3.2 Automatous Monk

Este pacote que contém três aplicativos, (REINERS, 2004), utiliza a biblioteca *jMusic*, (BROWN; SORENSEN, 2000), e o modelo de CAs unidimensionais elaborados por *Wolfram* para realizar composições musicais. Para tal, durante seu funcionamento o sistema proposto efetua uma correlação entre os estados atuais de cada célula do CA e as notas musicais previstas pelo protocolo MIDI. Ao término das iterações previstas, o usuário pode optar entre, realizar a audição do arquivo musical obtido dentro do próprio aplicativo ou exportá-lo para outros interpretadores MIDI.

Durante a fase de inicialização, o usuário configura o sistema, escolhendo a regra que será utilizada como base para a composição musical, dentre as 256 regras detalhadas por *Wolfram*, (WOLFRAM, 1994), o número de iterações que deverá ser realizado, o instrumento a ser utilizado, dentre os 3 disponíveis, a escala tonal, dentre as 7 disponíveis, e a clave musical. O usuário também deverá o tipo de inicialização, que poderá ser randômica ou simples. Na inicialização randômica, o aplicativo determina os estados iniciais de cada célula do *grid*. Já na inicialização simples, apenas a célula central do *grid* é ativada, tornando a evolução do sistema idêntica as evoluções detalhadas por *Wolfram*, (WOLFRAM, 1994).

De forma a permitir maior aplicabilidade do sistema proposto, a criação de tons pode ser realizada de três métodos distintos. No primeiro método, denominado Multi Voz, cada célula do CA corresponderá a um tom diferente, configurando-se assim a correspondência direta entre as células do CA e as notas musicais do protocolo MIDI, detalhadas no Capítulo 2 desta dissertação. O segundo método, denominado Binário, utiliza as iterações do CA para determinação dos tons. Neste método, a cada iteração, os estados atuais de cada célula do *grid*, que podem ser 0 ou 1, são organizados em um vetor, de forma que cada vetor corresponda a uma nota no protocolo MIDI. No terceiro método, denominado Binário Cumulativo, o usuário poderá determinar quantos vetores serão utilizados para gerar uma nota musical. Ainda com relação aos métodos para criação de tons, para os métodos Binário e Binário Cumulativo, o usuário poderá determinar o modo como o sistema efetuará a leitura dos vetores obtidos. Esta leitura poderá ser realizada de 3 formas, da esquerda para a direita, da direita para a esquerda e duas contagens. No formato duas contagens o sistema realiza duas leituras ao mesmo tempo, uma da esquerda para a direita e a outra da direita para a esquerda. O usuário também pode determinar o andamento musical a partir da definição do número de *bpms* [2].

O resultado audível deste aplicativo é extremamente interessante pois permite ao usuário a distinção clara entre as configurações realizadas. Contudo, apesar de ser uma aplicação robusta, não é explicitado claramente como são combinados os estados das células para a geração musical ou como os valores obtidos no vetores são analisados pelo sistema para escolha das notas musicais.

3.3.3 CAMus

Este aplicativo, (MIRANDA, 1993, 2001), apresenta uma proposta de composição musical a partir da utilização dos resultados da evolução simultânea de dois modelos diferentes de CA.

[2]Batidas por Minuto, *bpm*, é um termo bastante utilizado na área musical, principalmente em música eletrônica, para diferenciar o andamento musical. Quanto maior o valor de *bpm*, mais rápida será a composição.

Os modelos de CA utilizados são o autômato celular bidimensional *Demon Cyclic Space* de *Grif-feath* e o autômato celular bidimensional *Game of Life* de *Conway* (CONWAY, 1970; MIRANDA, 1993; BURRASTON et al., 2004; NIERHAUS, 2009). O autômato celular de *Conway* é utilizado para seleção dos tons musicais, através da implementação de um modelo de projeção no plano cartesiano no qual a coordenada do eixo abcissas indica o primeiro intervalo musical e a coordenada do eixo das ordenadas indica o segundo intervalo musical, criando-se assim uma sequência de três notas, que podem ser executadas de forma simultânea ou sequencial. A duração das notas é especificada pelo usuário com base em uma fórmula de distribuição e uma taxa de velocidade. Cada célula possui também uma tipologia interna diferente, que varia em função da vizinhança e estabelece os tons que podem ser assumidos pela célula ao longo da evolução do sistema. O autômato de *Grif-feath* é o responsável por determinar qual canal MIDI, de 1 a 9, será utilizado para a alocação das triplas geradas.

Em relação aos controles do usuário, esta ferramenta é bastante interativa, permitindo a especificação do modelo de distribuição, da quantidade de repetições que o sistema deverá executar, da velocidade e do conjunto de regras do CA de *Conway*. A interface ainda possui duas telas para visualização de evolução, uma para cada modelo de CA e uma barra de visualização de coordenadas, a ser utilizada na configuração inicial do CA. A única especificidade é que a visualização da evolução do modelo de *Grif-feath* não pode ser realizada em tempo real, sendo apresentado apenas o resultado final desta.

Os resultados sonoros deste aplicativo são melodias básicas, tornando-o uma excelente ferramenta para estudos musicais. Contudo, apesar de promissor, este aplicativo foi descontinuado limitando a sua aplicabilidade de mercado.

3.3.4 ChaOSynth

Este aplicativo, (MIRANDA, 1995, 2001), é um sistema complexo que utiliza a síntese granular e autômatos celulares para composição musical, configurando-se praticamente uma ilimitada ferramenta para produção de sons complexos e incomuns. Para tal, o *ChaOSynth*, ao contrário de outros sistemas de síntese granular, utiliza ao invês de metodologias matemáticas complexas o modelo de CA *Chemical Oscillator Celular Automatom - ChaOs*, (CARPENTER, 1984; ECCLES, 1958), que é um modelo metafórico de um fenômeno neurofisiológico conhecido como circuito de reverberação neural. Neste trabalho, cada célula do *grid* assume um valor, determinado pelo usuário, que é associado a uma coloração e a uma frequência tonal. No decorrer das iterações do CA, a tela do aplicativo varia de coloração, em função das transições

de estado de cada célula, proporcionando ao observador a impressão de estar assistindo a um filme de animação.

Já o processo de síntese granular é realizado através da geração de sucessões rápidas de sons curtos, chamados grânulos, como por exemplo, sons com 40 milissegundos de duração. Esta técnica causa no ouvinte a sensação de fluidez sonora, gerando neste a interpretação sonora de que a melodia alterna-se infinitamente, até que a fonte sonora seja desligada. Para fins de compreensão, este efeito pode ser comparado com a sobreposição sucessiva de diferentes imagens, em velocidade maior que a capacidade de escaneamento da retina humana, proporcionando uma quase hipnótica sensação de movimento. Desta forma, os sons são criados com base na definição de que cada grânulo é formado por várias componentes espectrais. Cada componente espectral é definida a partir dos parâmetros de frequência, amplitude e duração. Enquanto o usuário determina como será controlada a amplitude, durante a fase de inicialização do sistema, o aplicativo realiza o controle dos valores de frequência e duração de cada grânulo. A duração dos sons obtidos é determinada pelo número de iterações configurado pelo usuário e da duração de cada componente espectral, determinada pelo sistema, como por exemplo, se forem configuradas 100 iterações de 35 milissegundos cada, a melodia resultante possuirá 3,5 segundos de duração.

Uma das grandes críticas a este aplicativo é o fato de que dada a sua capacidade ilimitada de produção e a versatilidade de funcionalidades disponibilizadas ao usuário, existem inúmeros questionamentos acerca da real capacidade do sistema, dificultando a expansão de sua utilização no meio musical.

3.3.5 Wolfram Tones

Este aplicativo, (WOLFRAM, 1994, 2009), de forma semelhante ao *Automatous Monk*, utiliza CAs unidimensionais para geração musical. A estrutura visual deste sistema divide-se em 3 seções distintas, configurando-se de duas seções para configuração do usuário e uma seção para visualização de resultados. A primeira seção, denominada Painel de Estilos, permite ao usuário a escolha do estilo musical que será utilizado como característica rítmica da melodia. Para tal o usuário deverá escolher 1 dos 16 estilos disponíveis, ou, solicitar que o sistema defina, de forma randômica, o estilo a ser utilizado. A segunda seção, denominada Controles de Composição, apresenta ao usuário as opções de configuração de tempo, escolha de instrumentos, escolha do conjunto de regras e permite a visualização do mapeamento tonal da melodia. Dentre as configurações disponíveis na seção Controle de Composição o usuário deverá determinar o tipo

de regra a ser utilizado, dentre os 16 conjuntos disponíveis, ou, solicitar que o sistema efetue a escolha da regra randomicamente. Com relação aos instrumentos musicais este aplicativo permite ao usuário a escolha de até cinco instrumentos simultâneos e um modelo de percussão. Já com relação às notas musicais o usuário deverá optar entre uma das mais de vinte escalas tonais disponíveis, ou, definir as notas musicais que serão utilizadas. O número de notas por batida também é definido pelo usuário através da escolha número de *bpm* e do tempo de duração da execução de cada nota musical. A terceira seção apresenta ao usuário, de forma gráfica, a evolução realizada pelo autômato celular e as opções de tocar a parar que fazer parte do controle musical.

Com relação ao funcionamento deste sistema, após as configurações realizadas pelo usuário durante a fase de inicialização, no decorrer das evoluções programadas os estados atuais de cada célula do CA são coletados e armazenados pelo sistema. Uma especificidade deste sistema é o fato de que, apesar da coleta e armazenamento do estados de cada célula ao longo da evolução do CA, apenas os dados de um número restrito de células localizadas na parte central do *grid* será utilizado para composição musical.

A diferença entre o aplicativo *Wolfram Tones* e o aplicativo *Automatous Monk* consiste nos conjuntos de regras disponíveis e no número de configurações que podem ser realizadas pelo usuário. Contudo, apesar do aplicativo *Wolfram Tones* apresentar uma melhor configuração harmônica, em relação ao aplicativo *Automatous Monk* não é possível identificar em totalidade a forma como os resultados de iteração das células CA são utilizados na composição musical.

3.4 Considerações Finais do Capítulo

Os trabalhos, apresentados neste capítulo, comprovaram a aplicabilidade da utilização de CAs para resolução de problemas relacionados a evoluções complexas e a validade da utilização dos resultados de evolução dos CAs como base para composição de música algorítmica ratificando a validade do trabalho proposto nesta dissertação. O capítulo seguinte apresenta, em detalhes, a arquitetura proposta nesta dissertação.

Capítulo 4

ARQUITETURA PROPOSTA

NESTE Capítulo é apresentada a Arquitetura de Composição Algorítmica com base em Autômatos Celulares e padrão de arquivos MIDI - ACH2M, que foi a arquitetura desenvolvida no decorrer deste trabalho. Para tal, este capítulo foi dividido em cinco seções. A Seção 4.1 apresenta uma visão geral da macro-arquitetura através do detalhamento de seus sinais de interface. As demais seções abordam cada uma das três unidades da arquitetura proposta, que são: o Divisor de Frequência, o Conjunto de Autômatos Celulares e a Máquina MIDI.

4.1 Macro-Arquitetura

A arquitetura proposta, denominada `ACH2M`, que objetiva a composição musical algorítmica através da utilização de autômatos celulares foi estruturada em três unidades principais. A primeira unidade é o Divisor de Frequência, `DVC`, que é o componente responsável pela distribuição do *clock* para a arquitetura proposta. A segunda unidade é o Conjunto de Autômatos Celulares, `CCA`, que é o componente responsável pela geração dos dados necessários à composição musical. Esta unidade contém os modelos de autômatos celulares que foram utilizados neste trabalho, constituindo-se de autômatos celulares baseados nos modelos unidimensional de *Wolfram*, bidimensional de *Neuman*, bidimensional de *Moore* e tridimensional de *Neuman*. Esta unidade também contém o componente responsável pelo controle dos autômatos celulares. A terceira unidade é a Máquina MIDI, `MMIDI`, que é o componente responsável pela conversão dos dados gerados pelo componente `CCA` em produto musical. Esta unidade contém as tabelas do protocolo MIDI necessárias para a composição musical, como por exemplo, os códigos referentes as notas e instrumentos musicais.

A macro-arquitetura proposta é apresentada na Figura 24. A representação dos sinais de tamanho 1 *bit* é efetuada através da utilização de linhas finas e os demais sinais, cujos tamanhos serão identificados no decorrer deste capítulo, são apresentados em linhas grossas.

Durante o período de inicialização da arquitetura, ativado pelo sinal Reset, os componentes CCA e MMIDI recebem as configurações escolhidas pelo usuário através dos sinais Regra, CI, Ciclos, DUser, User. Após a inicialização da arquitetura, os resultados das iterações dos autômatos celulares contidos no componente CCA são enviados para o componente MMIDI através do sinal Comp. Quando o componente MMIDI processa todos os resultados liberados pelo CCA é gerado o produto musical, que deverá ser convertido no formato *.mid* e a arquitetura encerra seu processamento.

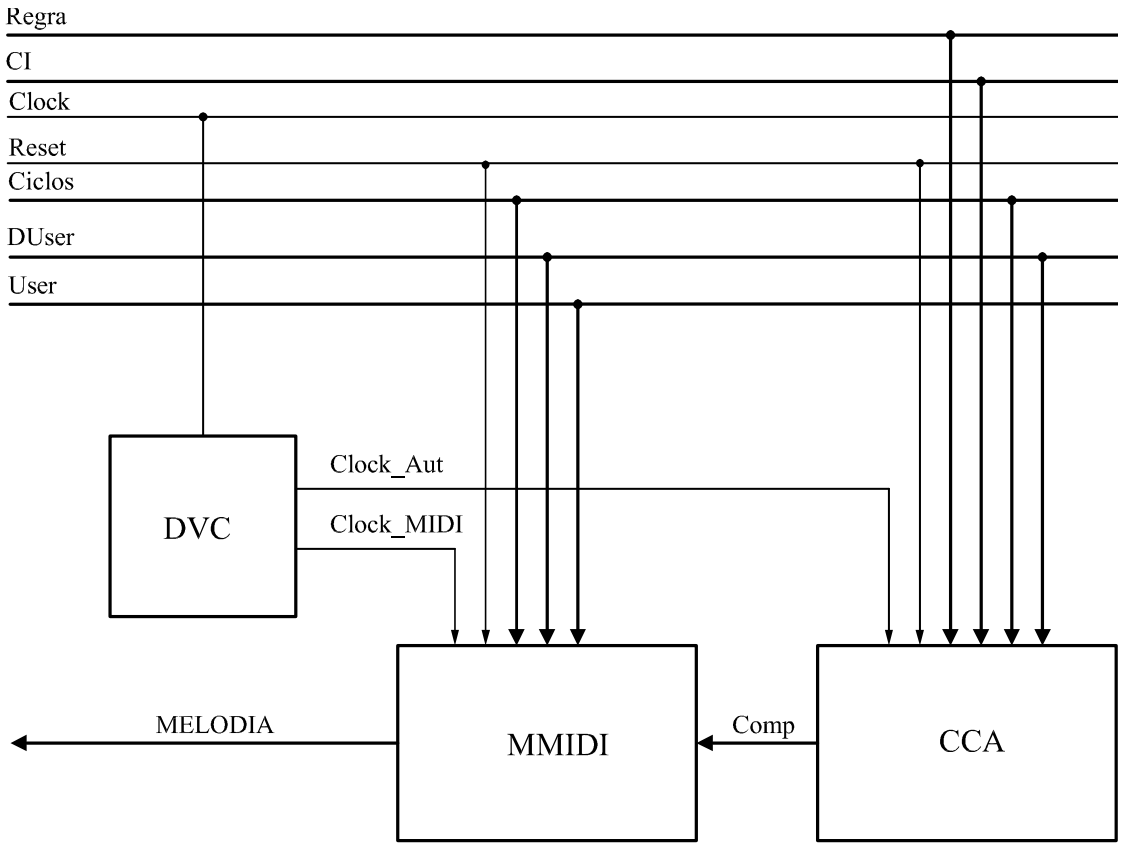


Figura 24: Arquitetura proposta

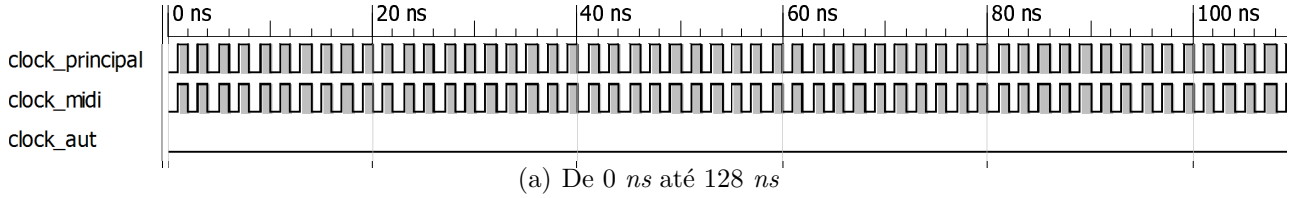
As próximas seções deste capítulo apresentarão em detalhes cada um dos sinais de interface e a descrição dos componentes utilizados na arquitetura, descrevendo o funcionamento e importância dos mesmos para a geração de produto musical.

4.2 Divisor de Frequência

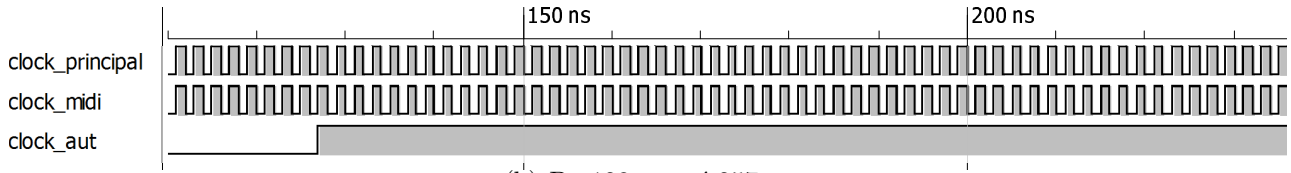
O componente Divisor de Clock, DVC, permite a sincronização das operações efetuadas pelos componentes CCA e MMIDI. A implementação deste componente foi realizada em função da razão de $\frac{1}{128}$ entre a frequência de operação do componente CCA e a frequência de operação do componente MMIDI. Tal diferença na frequência de operação deriva-se do fato de que ao

término de cada iteração do componente CCA, este libera o resultado das iterações para o componente MMIDI através do sinal Comp. Somente após o recebimento dos dados contidos no sinal Comp é que o componente MMIDI inicia seu ciclo de operação. Durante seu ciclo de operação o componente MMIDI analisa e converte os dados contidos no sinal Comp, e, para que esta conversão possa ser realizada, faz-se necessária a divisão dos dados contidos no sinal Comp em blocos de 128 *bits*, uma vez que o processamento do sinal Comp é realizado de forma serial pelo componente MMIDI, conforme será explicitado na Seção 4.5. O componente DVC atua durante toda a operação da arquitetura ACH2M recebendo o *clock principal* e o transformando em dois sinais. O primeiro sinal é o Clock_midi, que possui a mesma frequência do *clock principal* e é enviado aos componentes da unidade MMIDI. O segundo sinal é o Clock_aut, que possui frequência 128 vezes menor que o *clock principal*, é enviado aos componentes da unidade CCA.

A Figura 25 apresenta a simulação de funcionamento do componente DVC. Nesta simulação configurou-se 2 *ns* como o período de duração para cada ciclo do *clock principal*. No início da simulação, apresentado na Figura 25(a), o componente DVC recebe a frequência de *clock principal*, através do sinal Clock, e a repassa ao componente MMIDI, através do sinal Clock_midi. Somente após 64 ciclos do *clock principal* é que o sinal Clock_aut tem a sua transição positiva, apresentada Figura 25(b). Após mais 64 ciclos do *clock principal* o sinal Clock_aut tem a sua transição negativa.



(a) De 0 *ns* até 128 *ns*



(b) De 128 *ns* até 257 *ns*

Figura 25: Simulação da operação do componente DVC

4.3 Conjunto de Autômatos Celulares

A arquitetura do Conjunto de Autômatos Celulares, CCA, é formada por nove componentes diferentes. Cada autômato celular presente na unidade CCA é formado por 128 células, totalizando

as 1024 células presentes na arquitetura proposta ACH2M.

A arquitetura do CCA é apresentada em destaque na Figura 26. O componente Gestor de Autômatos Celulares, GCA, é responsável pelo controle dos oito autômatos celulares existentes na unidade CCA, pelo armazenamento temporário dos resultados que serão utilizados na composição musical e pelo controle do número total de iterações que deverá ser realizado pelos CA habilitados. Os componentes Unidimensional de *Wolfram* para geração de notas, UWN, e Unidimensional de *Wolfram* para geração de instrumentos, UWI, são CAs unidimensionais elaborados a partir das características propostas por *Wolfram* (WOLFRAM, 1994). Os componentes Bidimensional de *Neuman* para geração de notas, BNN, e Bidimensional de *Neuman* para geração de instrumentos, BNI, são CAs bidimensionais elaborados a partir das características propostas por *Neuman* (GREMONINI; VICENTINI, 2008). Os componentes Bidimensional de *Moore* para geração de notas, BMN, e Bidimensional de *Moore* para geração de instrumentos, BMI, são CAs bidimensionais elaborados a partir das características propostas por *Moore* (GREMONINI; VICENTINI, 2008). Os componentes Tridimensional de *Neuman* para geração de notas, TNN, e Tridimensional de *Neuman* para geração de instrumentos, TNI, são CAs tridimensionais elaborados a partir da aplicação das características propostas por *Neuman* para o ambiente tridimensional (FOLINO et al., 2006; OLIVEIRA et al., 2006).

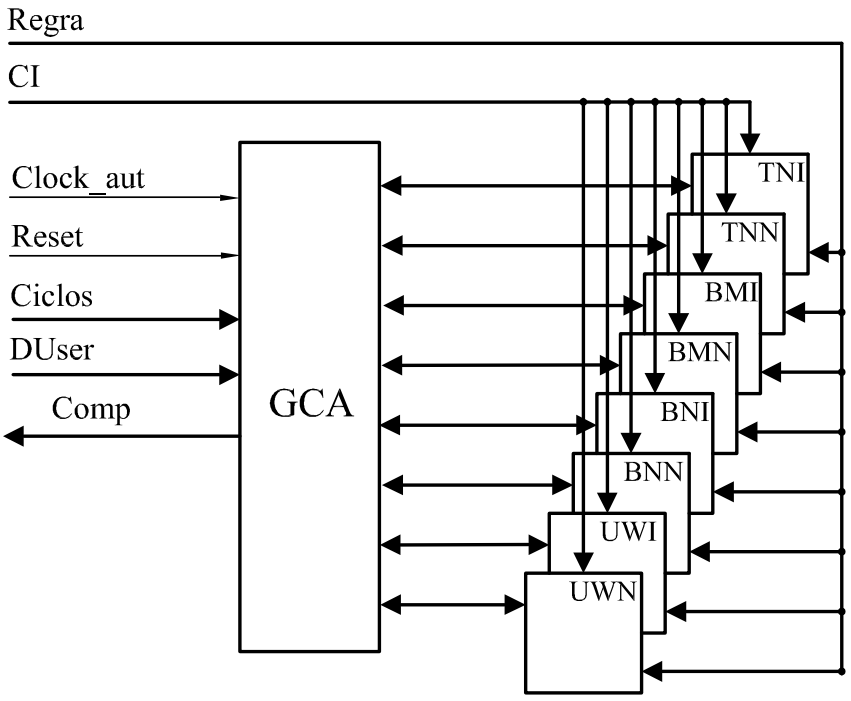


Figura 26: Arquitetura da unidade Conjunto de Autômatos Celulares

As regras utilizadas para composição do conjunto *vizinhança* de cada CA serão detalhadas no decorrer desta seção. O fluxo de sinais da unidade `CCA` é composto pelo sinal Clock_aut e por outros seis sinais externos. O sinal Reset permite a inicialização da unidade CCA. O sinal Ciclos, cujo valor é configurado pelo usuário, contém o número total de iterações que as células deverão executar para geração dos resultados necessários a composição da melodia. O sinal Regra, contém os conjuntos de regras que serão utilizados pelos CAs habilitados durante processo de composição musical. Seu tamanho, 128 *bytes*, é determinado pelos modelos de *vizinhança* utilizados nesta dissertação, sendo reservado para cada CA um segmento do sinal Regra, conforme apresentado na Tabela 19. No ato da configuração do sistema o usuário insere os valores, em numeração binária, do conjunto de regras que serão utilizados nos CAs habilitados, respeitando o número de dados necessários a ativação de cada CA.

Tabela 19: Divisão estrutural do sinal Regra

Segmento	Regra$_{UWN}$	Regra$_{UWI}$	Regra$_{BNN}$	Regra$_{BNI}$
Alocação de *bits*	1359 – 1352	1351 – 1344	1343 – 1312	1311 – 1280
Tamanho do Segmento	8	8	32	32

Segmento	Regra$_{BMN}$	Regra$_{BMI}$	Regra$_{TNN}$	Regra$_{TNI}$
Alocação de *bits*	1279 – 768	767 – 256	255 – 128	127 – 0
Tamanho do Segmento	512	512	128	128

A Tabela 20 apresenta uma exemplificação do processo de configuração do sinal Regra através do segmento Regra$_{UWN}$. Considerando a utilização da regra 25 de *Wolfram*, (WOLFRAM, 1994), os *bits* do sinal Regra referentes a configuração do componente `UWN`, apresentados na segunda linha da Tabela 20, armazenariam o vetor 00011001 no formato apresentado na terceira linha desta mesma tabela. O usuário só efetuará a configuração dos segmentos referentes aos CAs que deseja habilitar. Os segmentos referentes aos CAs que não serão habilitados são marcados pela arquitetura, que não efetuará qualquer tipo de análise sobre estes segmentos.

Tabela 20: Exemplo de configuração do segmento Regra$_{UWN}$

Regra$_{UWN}$								
bit	1359	1358	1357	1356	1355	1354	1353	1352
valor	0	0	0	1	1	0	0	1

Para habilitação dos CAs, o usuário utiliza o sinal DUser. Este sinal, de tamanho 8 *bytes* identifica quais CAs serão habilitados para composição musical, sendo reservado 1 *bit* para cada módulo CA existente na unidade `CCA`, conforme apresentado na Tabela 21.

Tabela 21: Divisão estrutural do sinal DUser

Segmento	DUser_{UWN}	DUser_{UWI}	DUser_{BNN}	DUser_{BNI}
Alocação de *bits*	7	6	5	4

Segmento	DUser_{BMN}	DUser_{BMI}	DUser_{TNN}	DUser_{TNI}
Alocação de *bits*	3	2	1	0

Durante a inicialização do sistema, o `CCA` realiza um chaveamento que parametriza o acréscimo de um determinado CA, através da utilização de portas `AND` e do sinal DUser. Nesta etapa, o usuário deverá escolher dois CAs, um para geração de notação musical e outro para geração de notação instrumental. O esquema de chaveamento realizado pelo componente `GCA` e a ordem de configuração do sinal DUser são apresentados na Figura 27.

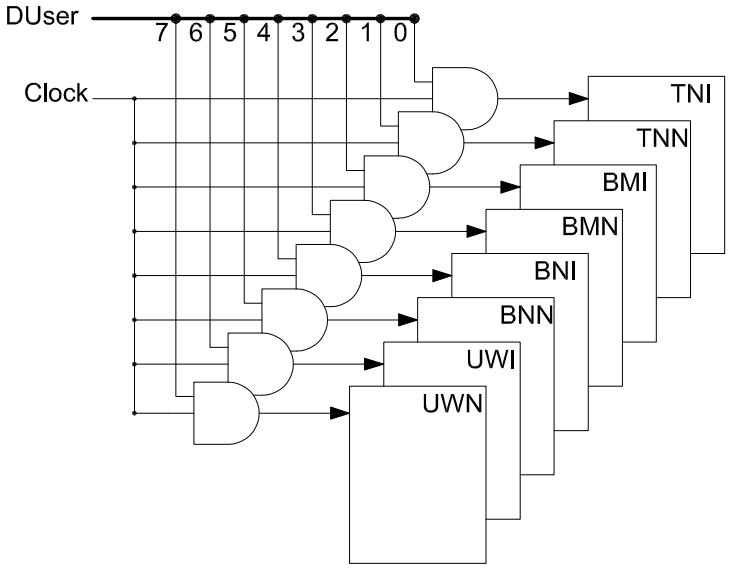


Figura 27: Visão amplificada do modelo de habilitação dos CAs contidos na unidade `CCA`

Alguns exemplos de configurações possíveis para o sinal DUser, com a indicação dos CAs que serão habilitados na arquitetura, são apresentados na Tabela 22.

Tabela 22: Exemplos de Configuração do sinal DUser

DUser	**Notação Musical**	**Notação Instrumental**
10010000	`UWN`	`BNI`
00000011	`TNN`	`TNI`
00011000	`BMN`	`BNI`

O sinal CI é o portador dos estados iniciais de cada uma das células integrantes dos CAs habilitados. Seu tamanho, 128 *bytes*, é determinado pelo número total de células da arquitetura.

Para cada *grid* existente é reservado um segmento de 16 *bytes*, conforme apresentado na Tabela 23. Dentro destes segmentos cada *bit* representa um estado inicial, q_0, definido pelo usuário que deverá ser assumido por uma célula do *grid*, de acordo com a posição desta no *grid*. Este estado inicial poderá ser 0 ou 1.

Tabela 23: Divisão estrutural do sinal CI

Segmento	CI_{UWN}	CI_{UWI}	CI_{BNN}	CI_{BNI}
Alocação de *bits*	0 – 127	128 – 255	256 – 383	384 – 511
Tamanho do Segmento	128	128	128	128

Segmento	CI_{BMN}	CI_{BMI}	CI_{TNN}	CI_{TNI}
Alocação de *bits*	512 – 639	640 – 767	768 – 895	896 – 1023
Tamanho do Segmento	128	128	128	128

O sinal Comp é o portador dos resultados da iteração corrente de cada uma das células dos CAs habilitados. Seu tamanho, 128 *bytes*, de forma idêntica ao sinal CI é determinado pelo número total de células da arquitetura, sendo reservado um segmento de 16 *bytes* para cada módulo CA, conforme apresentado na Tabela 24.

Tabela 24: Divisão estrutural do sinal Comp

Segmento	$Comp_{UWN}$	$Comp_{UWI}$	$Comp_{BNN}$	$Comp_{BNI}$
Alocação de *bits*	1023 – 896	895 – 768	767 – 640	639 – 512
Tamanho do Segmento	128	128	128	128

Segmento	$Comp_{BMN}$	$Comp_{BMI}$	$Comp_{TNN}$	$Comp_{TNI}$
Alocação de *bits*	511 – 384	383 – 256	255 – 128	127 – 0
Tamanho do Segmento	128	128	128	128

Dentro destes segmentos, cada *bit* registra o estado atual de uma célula. Ao final de cada iteração corrente, os estados atuais dos CAs habilitados são inseridos no sinal Comp que será enviado para a unidade `MMIDI`. Todos os CAs obedecem a mesma sequência de operação e cada iteração dos CAs, da análise de vizinhança até a liberação do resultado, demanda um ciclo do sinal Clock_aut. O único período de funcionamento atípico da unidade `CCA` é durante a inicialização do sistema, conforme será abordado durante a simulação do componente `CGA`.

O funcionamento do sistema depende diretamente da configuração correta dos sinais Regra, DUser e CI, pois estes sinais definem o funcionamento da unidade `CCA` e consequentemente de toda a arquitetura.

A arquitetura do `CCA` é composta por 9 componentes que podem ser agrupados em 5 unidades funcionais: Gestor de Autômatos Celulares, Autômato Celular Unidimensional de

Wolfram, Autômato Celular Bidimensional de *Neuman*, Autômato Celular Bidimensional de *Moore* e Autômato Celular Tridimensional de *Neuman*. A diferença entre o número de componentes e o número de unidades funcionais existe porque cada dupla de autômato celular, formada por um autômato celular para geração de instrumentos e um autômato celular para geração de Notas, como por exemplo `BMN` e `BMI` possui arquitetura interna idêntica. Nas próximas seções serão apresentados os detalhes de cada uma destas cinco unidades funcionais através da descrição de sua função, de forma a caracterizar atuação de cada uma destas unidades na geração de resultados.

4.3.1 Gestor de Autômatos Celulares

O componente Gestor de Autômatos Celulares, `GCA`, é responsável pelo controle dos 8 CAs existentes no `CCA`, realizando portanto todas as funções inerentes a este controle. As funções do `GCA`, por ordem de execução, são:

1. Habilitar os CAs;
2. Controlar o número de iterações que os CAs realizarão;
3. Armazenar temporariamente os resultados da iteração corrente de cada uma das células dos CAs habilitados;
4. Enviar o resultado dos valores de iteração corrente a unidade `MMIDI`;
5. Desabilitar os CAs.

A arquitetura do `CGA` é apresentada na Figura 28. A unidade de parametrização, `UPar`, é o componente responsável por transmitir o sinal Clock_aut, enviado pelo componente `DIV`, e o sinal Reset para os demais componentes da unidade `CCA`. Este componente também é responsável pela habilitação dos CAs, apresentada na Figura 27, e pelo desligamento da unidade `CCA` ao término das iterações programadas. Os registradores, `RUWN`, `RUWI`, `RBNN`, `RBNI`, `RBMN`, `RBMI`, `RTNN` e `RTNI`, de tamanho de 16 *bytes*, são os componentes responsáveis pelo armazenamento temporário dos valores resultantes de cada iteração corrente das células referentes aos CAs habilitados e pela organização do sinal Comp. O registrador `RCiclos` é o componente responsável por armazenar o valor do sinal Ciclos durante o período de inicialização do sistema e pelo armazenamento temporário dos valores gerados pelo componente `Count`. O componente `Count` é o componente responsável por realizar o decremento do valor inicial armazenado pelo registrador `RCiclos` durante o funcionamento do `CCA` e pelo informe do término das iterações programadas

ao componente UPar. Quando o componente Count informa ao componente UPar que o número de iterações programadas foi atingido este desabilita os CAs através da desativação do sinal de *clock*.

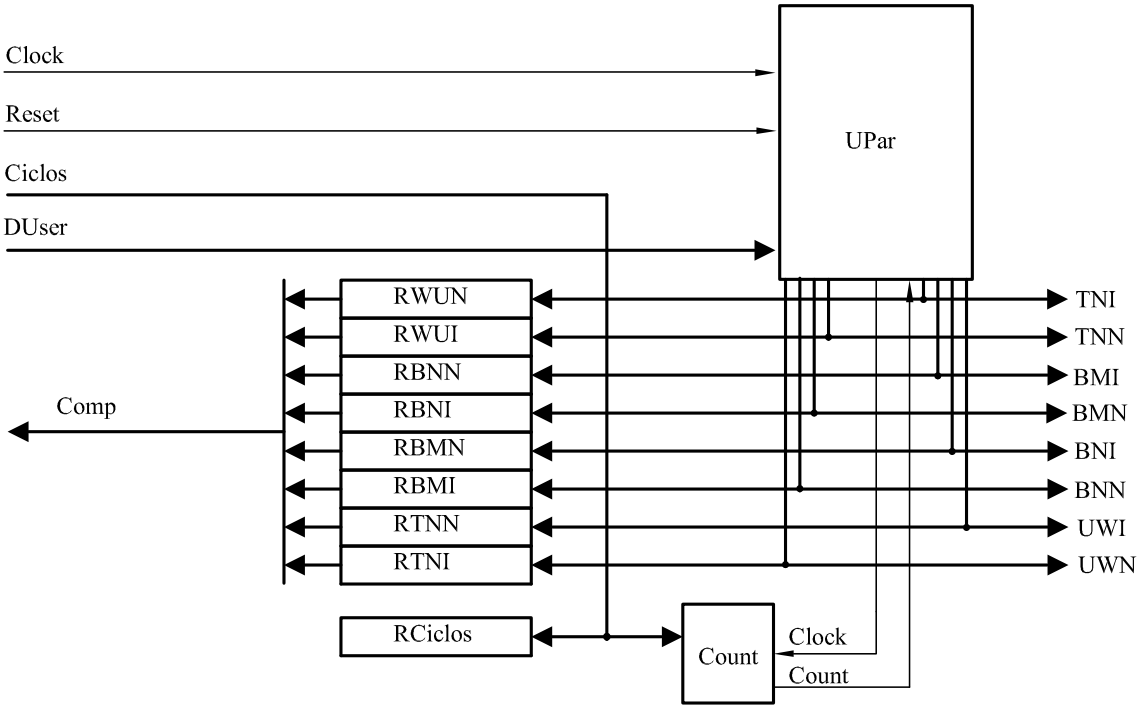


Figura 28: Arquitetura do componente CGA

O primeiro ciclo de operação do componente CGA é apresentado na Figura 29. Durante este período, no qual o sinal Reset é igual a 1, o CGA recebe o sinal Clock_aut e o sinal Reset transmitindo-os para os CAs habilitados, neste exemplo os componentes UWN e BNI, que recebem respectivamente os sinais Clock_uwn, Reset_uwn, Clock_bni e Reset_bni. A identificação dos módulos CA que deverão ser habilitados é realizada através do sinal DUser, que na simulação foi configurado como 10010000_2, habilitando os componentes UWN e BNI, conforme apresentado na Figura 27. É durante este período de inicialização que os sinais Regra, CI, são recebidos e transmitidos aos CAs habilitados, neste exemplo através dos sinais Regra_uwn, Regra_bni, CI_uwn e CI_bni. Para esta simulação foi definido 6 como o número total de iterações a serem realizadas. Este número total de iterações é identificado pelo CGA através do sinal Ciclos. A diferença entre as tarefas executadas durante de inicialização e as tarefas executadas no decorrer do funcionamento da arquitetura é que durante a inicialização são realizadas duas liberações de valores no sinal Comp. O primeiro valor liberado no sinal Comp é a configuração inicial do CA, já o segundo valor liberado é o resultado da primeira iteração. Conforme identificado nas Tabelas 23 e 24, os *bits* dos sinais CI e Comp são processados de forma oposta arquitetura,

como por exemplo, os dois últimos *bytes* do sinal CI_bni, 01111111_2, são distribuídos para as oito primeiras células do *grid* na forma 11111110_2. Na primeira transição positiva do *clock* os CAs habilitados assimilam a configuração inicial e efetuam a primeira iteração.

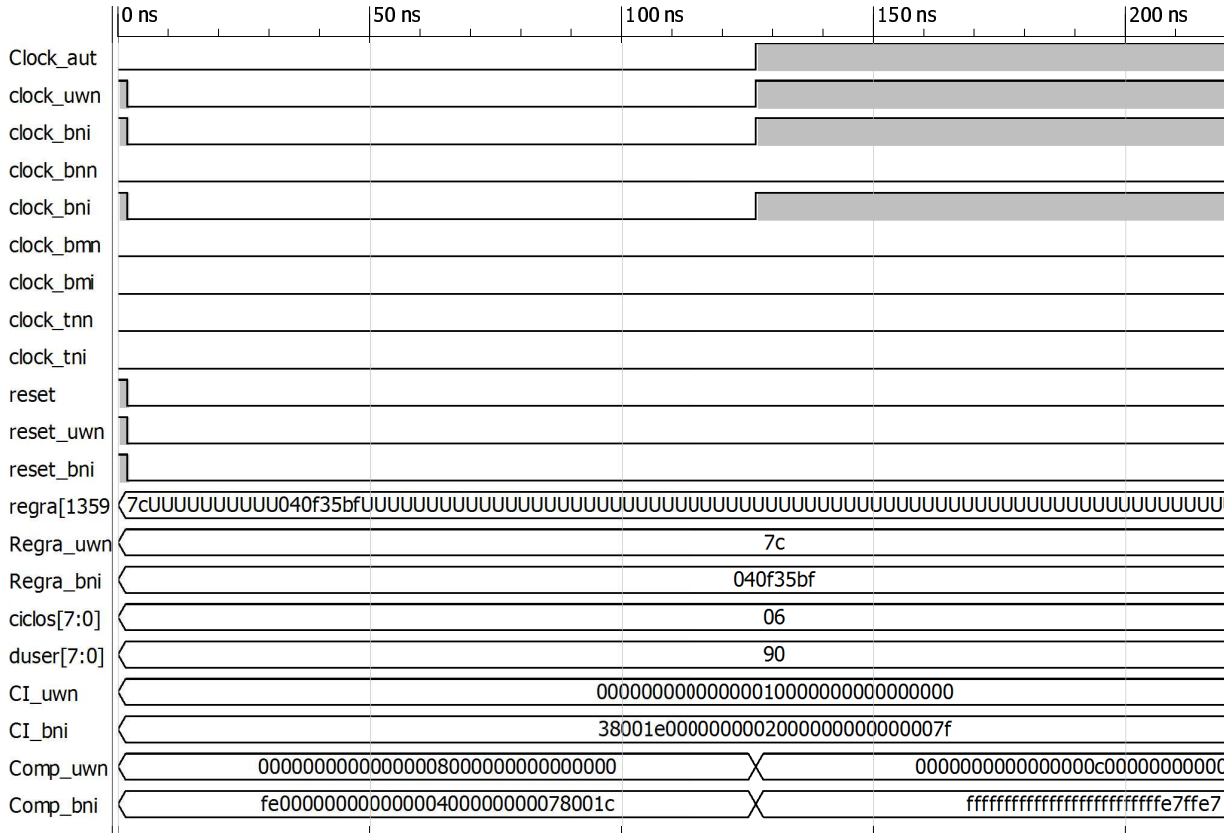


Figura 29: Diagrama do primeiro ciclo de operação do componente `CGA`

4.3.2 Módulo de Autômatos Celulares

O módulo de autômatos celulares, denominado neste trabalho de módulo CA, é composto por 4 unidades funcionais contendo um total de 8 componentes idealizados a partir dos modelos de conjunto *vizinhança* de *Neuman*, *Wolfram* e *Moore*, descritos no Capítulo 1. Estes autômatos celulares foram organizado na unidade `CCA` da seguinte forma:

- 2 CAs, `UWN` e `UWI`, com 128 células cada, modelo de *vizinhança* de *Wolfram*, $r = 1$, e distribuição de *grid* 1×128, denominados Unidimensionais de *Wolfram*.
- 2 CAs, `BNN` e `BNI`, com 128 células cada, modelo de *vizinhança* de *Neuman*, $r = 1$, e distribuição de *grid* 8×16, denominados Bidimensionais de *Neuman*.
- 2 CAs, `BMN` e `BMI`, com 128 células cada, modelo de *vizinhança* de *Moore*, $r = 1$, e distribuição de *grid* 8×16, denominados Bidimensionais de *Moore*.

- 2 CAs, `TNN` e `TNI`, com 128 células cada, modelo de *vizinhança* tridimensional de *Neuman*, $r = 1$, e distribuição de *grid* $4 \times 8 \times 4$, denominados Tridimensional de *Neuman*.

As arquiteturas dos autômatos celulares elaboradas no decorrer desta dissertação foram baseadas no fato de que todo *grid* de autômato celular pode ser definido como uma matriz *n-dimensional*. Esta definição permite que todos os elementos de um conjunto *vizinhança* possam ser identificados a partir de funções que utilizem como parâmetro o índice de localização da célula no *grid*. Para esta dissertação, que utiliza *grids* de até 3 dimensões, com valor de $r = 1$, as funções gerais para localização dos elementos do conjunto *vizinhança* são as apresentadas nas Equações 7 e 8:

$$f(x, H) = \begin{cases} H & \text{se } x = 1; \\ x - 1 & \text{senão;} \end{cases} \tag{7}$$

$$g(x, H) = \begin{cases} H & \text{se } x = 1; \\ x + 1 & \text{senão;} \end{cases} \tag{8}$$

Considerando (i, j, k) como a simbologia representativa dos índices para localização matricial e (M, N, P) como a simbologia representativa para indicação da ordem matricial, a localização dos valores do conjunto *vizinhança* de uma célula irá sempre variar em função do modelo de *grid* adotado e da correlação entre as triplas (i, j, k) e (M, N, P) e a dupla de funções $f(x, H)$ e $g(x, H)$. Esta correlação será apresentada e exemplificada no decorrer das próximas seções.

4.3.2.1 Autômato Celular Unidimensional de Wolfram

As regras propostas por *Wolfram*, conforme explicado no Capítulo 1, retratam características de um autômato celular unidimensional. O resultado da aplicação destas características é um CA cujo *grid* comporta-se graficamente como um vetor. Neste trabalho, no qual se utiliza $r = 1$, o conjunto *vizinhança* de uma determinada célula é sempre composto pelas duas células adjacentes a esta. Portanto, no caso do modelo de *grid* unidimensional de *Wolfram* horizontal, utilizado nesta dissertação, as triplas (i, j, k) e (M, N, P) podem ser substituídas respectivamente pelas triplas $(1, j, 1)$ e $(1, N, 1)$.

Considerando que o conjunto *vizinhança* utilizado contém dois elementos, a correta localização dos elementos pertencentes ao conjunto *vizinhança* de cada célula no modelo de *Wolfram* dependerá diretamente dos valores de j e N, sendo o método para identificação dos

elementos que compõem o conjunto *vizinhança* o apresentado na Equação 9:

$$V_j = \{V_{i,f_{(j,N)}}, V_{i,g_{(j,N)}}\} \tag{9}$$

A função V_j engloba a resolução do problema de borda para o CA proposto, através de ligação entre a primeira e a última célula, eliminando possíveis variações de resultado causadas por localizações incorretas dos elementos do conjunto *vizinhança*. Um exemplo do modelo de interligação utilizado através da aplicação da função V_j é apresentado na Figura 30. Neste exemplo a ligação entre células é apresentada na cor preta.

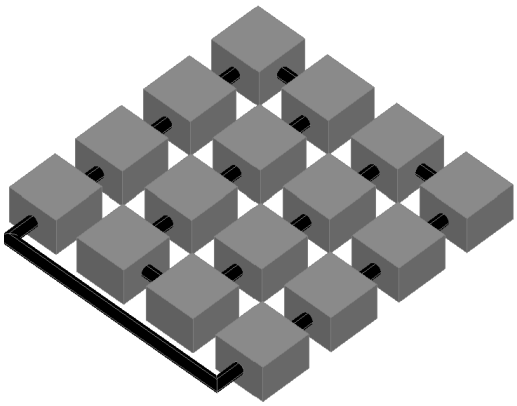


Figura 30: Organização estrutural do componente `UWN`

Na Figura 30, que representa um CA de 16 células e $r = 1$, é possível visualizar a sequência completa da ligação de vizinhança e a estratégia utilizada para resolução do problema de borda, configurando-se um exemplo de aplicação prática da fita infinita de *Turing*. A aplicação da função V_j neste CA exemplo torna possível a localização dos elementos integrantes do conjunto *vizinhança*, como por exemplo:

1. $V_{1,1} = \{V_{1,16}, V_{1,2}\}$;
2. $V_{1,3} = \{V_{1,2}, V_{1,4}\}$;
3. $V_{1,7} = \{V_{1,6}, V_{1,8}\}$;
4. $V_{1,10} = \{V_{1,9}, V_{1,11}\}$;
5. $V_{1,15} = \{V_{1,14}, V_{1,16}\}$.

A arquitetura do módulo `UWN` é apresentada na Figura 31, na qual as células são representadas pela denominação $C_{i,j}$. Quando este módulo é habilitado, ele recebe os sinais

Clock_uwn, Reset_uwn. Durante o período de inicialização, os valores contidos nos sinais Regra_uwn e CI_uwn são armazenados para que as células contidas neste componente possam efetuar as operações necessárias a realização das iterações programadas. Ao término de cada iteração o estado atual de cada célula é liberado para o componente GCA através do sinal Comp_uwn.

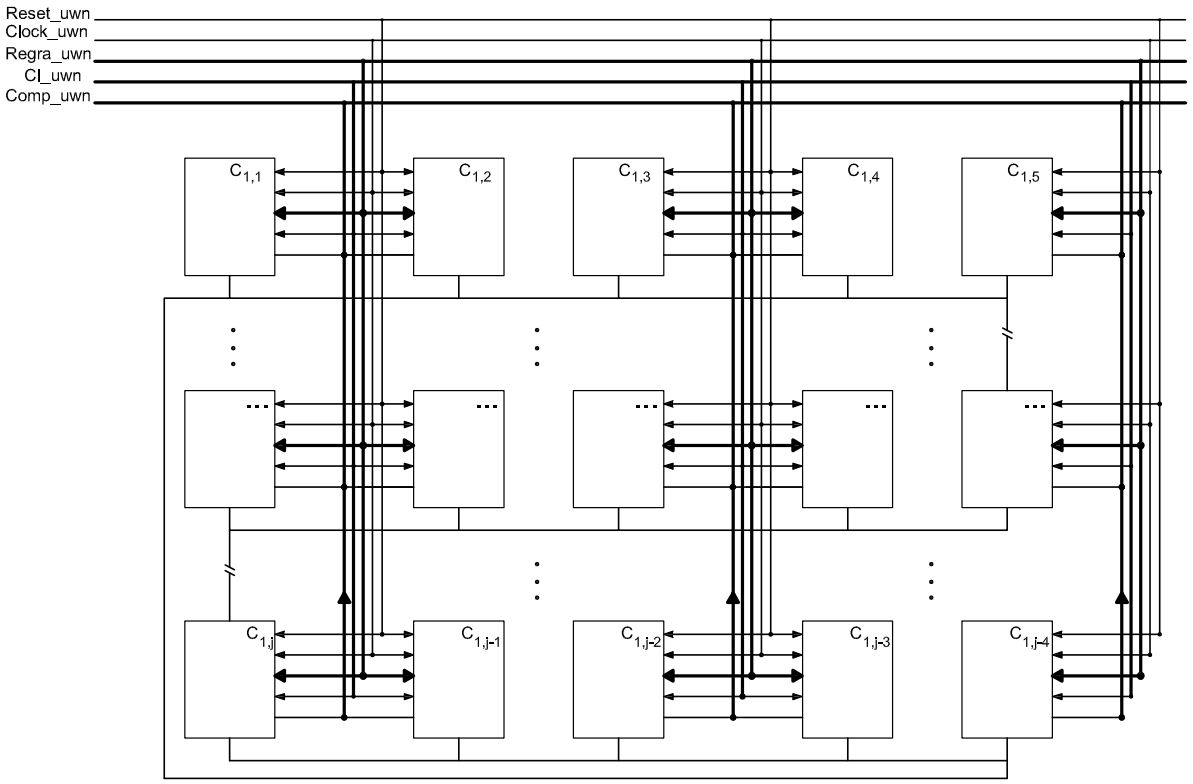
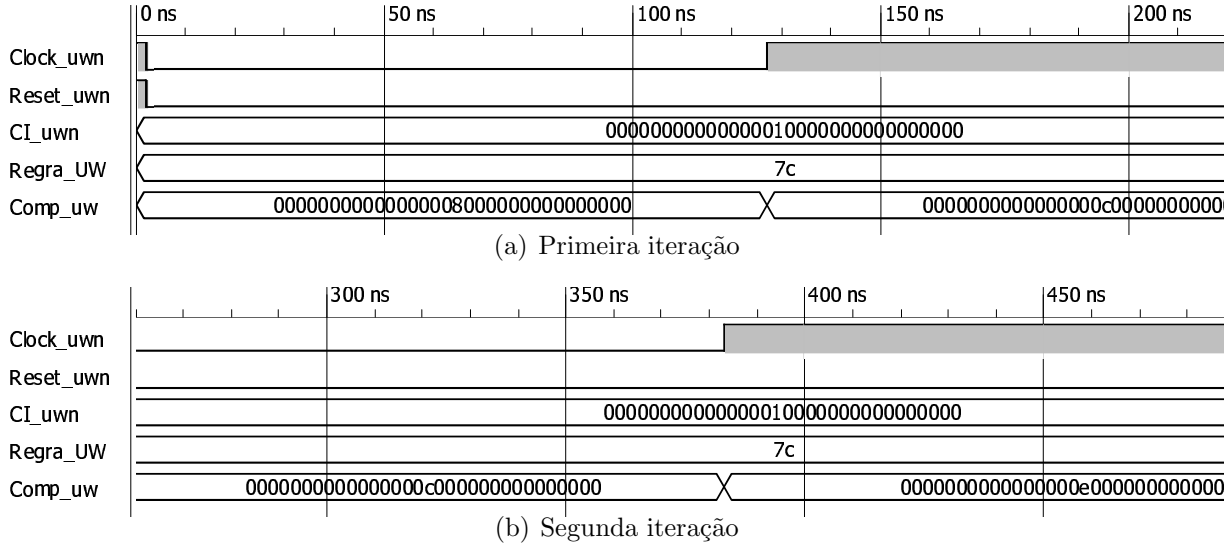


Figura 31: Arquitetura do componente UWN

Os dois primeiros ciclos de operação do componente UWN são apresentados na Figura 32. Quando habilitado, durante o período de inicialização, o UWN recebe os sinais de Clock_uwn e Reset_uwn enviados pelo componente CGA interpretando os sinais Regra_uwn e CI_uwn configurados pelo usuário, conforme apresentado na Figura 32(a). Durante o período em que o valor do *clock* é igual a 0, as células deste componente realizam as operações de armazenamento necessárias a iteração, como por exemplo o registro da configuração inicial e do conjunto de regras. Na primeira transição positiva do sinal Clock_uwn as células deste CA liberam os resultados da primeira iteração realizada. Estes resultados são enviados ao componente CGA através do sinal Comp_uwn. A regra utilizada nesta simulação, transferida para a arquitetura através do sinal Regra_uwn, foi a regra 124 de *Wolfram* (WOLFRAM, 1994), 01111100_2. A configuração inicial, transferida para a arquitetura através do sinal CI_uwn teve apenas a célula central ativada. A segunda iteração realizada pelas células do componente UWN é apresentada na Figura 32(b).

Todas as transferências de dados executadas pelo componente UWN ocorrem nas transições positivas do *clock*, até que seja alcançado o número programado de iterações e a unidade CCA seja desativada.



(a) Primeira iteração

(b) Segunda iteração

Figura 32: Simulação da operação do componente UWN

A leitura da regra utilizada permite a compreensão de que qualquer combinação de entrada diferente das combinações $000, 001, 111$ terá como resultado a ativação da célula. Dado o conjunto de regras e a configuração inicial utilizados, analisando as 6 células próximas a célula central, 3 a direita e 3 a esquerda, obtém-se o resumo das primeiras evoluções deste CA, conforme apresentado na Tabela 25. Utilizando a célula central desta tabela como referência, seu estado atual e os de sua *vizinhança* formam a entrada 010, que pela definição da regra faz com que a célula mantenha-se ativa até a próxima iteração. Na iteração seguinte esta mesma célula central e sua *vizinhança* formam a entrada 011 que mantém a célula ativa. Este resumo permite comprovar a importância da configuração inicial para a evolução de qualquer CA uma vez que, como no exemplo simulado apenas a célula central foi ativada toda evolução do CA concentrou-se no entorno desta célula.

Tabela 25: Resumo das primeiras iterações realizadas pelo componente UWN

Configuração Inicial	0	0	0	1	0	0	0
Primeira iteração	0	0	0	1	1	0	0
Segunda iteração	0	0	0	1	1	1	0

A arquitetura e o funcionamento do componente UWI são idênticos aos descritos para o componente UWN.

4.3.2.2 Autômato Celular Bidimensional de Neuman

As regras propostas por *Neumam*, conforme explicado no Capítulo 1, retratam as características de um autômato celular bidimensional. Neste trabalho, que utiliza $r = 1$, o conjunto *vizinhança* de uma célula qualquer será composto por todas as células lateralmente adjacentes a este em um ambiente bidimensional. Portanto, no caso do *grid* bidimensional de *Neuman*, utilizado nesta dissertação, as triplas (i, j, k) e (M, N, P) podem ser substituídas respectivamente pelas triplas $(i, j, 1)$ e $(M, N, 1)$.

Considerando que o modelo de conjunto *vizinhança* utilizado contém quatro elementos, a correta localização dos elementos pertencentes ao conjunto *vizinhança* de cada célula no modelo de *Neuman* dependerá exclusivamente dos valores de (i, j) e (M, N). Neste caso, o formato de localização de cada um dos elementos que compõem o conjunto *vizinhança* de *Neuman* é o apresentado na Equação 10.

$$V_{i,j} = \{V_{f_{(i,M)},j}, V_{i,g_{(j,N)}}, V_{g_{(i,M)},j}, V_{i,f_{(j,N)}}\} \tag{10}$$

De forma equivalente a função V_j, apresentada na Equação 9, a função $V_{i,j}$ apresenta a modelagem aplicada nesta dissertação para resolução do problema de borda e eliminação possíveis variações de resultados causadas por assimilação incorreta de elementos do conjunto *vizinhança*. Um exemplo do modelo de interligação utilizado, através da aplicação da função $V_{i,j}$ é apresentado na Figura 33. Neste exemplo as ligações de *vizinhança* são representadas na cor preta.

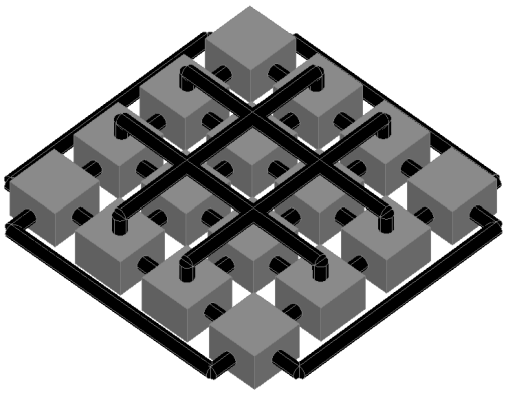


Figura 33: Organização estrutural do componente `BNN`

Na Figura 33, que representa um CA com 16 células, 4×4, e $r = 1$, é possível visualizar a sequência completa de interligação de vizinhança de *Neuman* e a resolução do problema de

borda. A aplicação da função $V_{i,j}$ neste exemplo permite a correta localização dos elementos que compõem o conjunto *vizinhança* de uma célula, como por exemplo:

1. $V_{1,1} = \{V_{4,1}, V_{1,2}, V_{2,1}, V_{1,4}\}$;
2. $V_{2,3} = \{V_{1,3}, V_{2,4}, V_{3,3}, V_{2,2}\}$;
3. $V_{3,4} = \{V_{2,4}, V_{3,1}, V_{4,4}, V_{3,3}\}$;

A arquitetura do componente `BNN` é apresentada na Figura 34, na qual as células são representadas pela denominação $C_{i,j}$. Quando este módulo é habilitado, ele recebe os sinais Clock_bnn e Reset_bnn. Durante o período de inicialização, os valores contidos nos sinais Regra_bnn e CI_bnn são armazenados para que as células contidas neste componente possam efetuar as operações necessárias a realização das iterações programadas. Ao término de cada iteração, o estado atual de cada célula é liberado para o componente `GCA` através do sinal `Comp_bnn`.

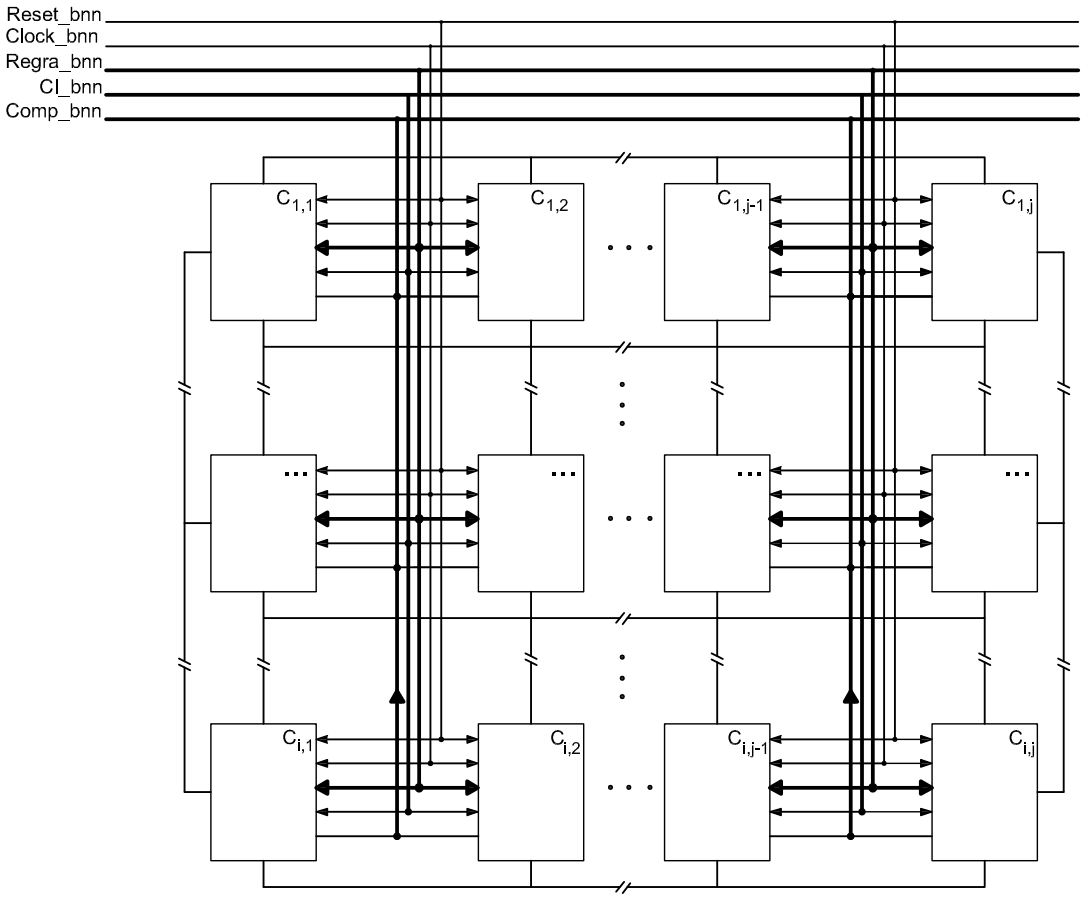
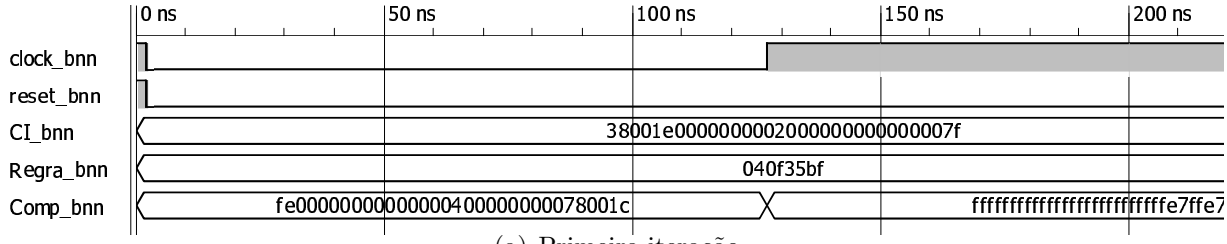
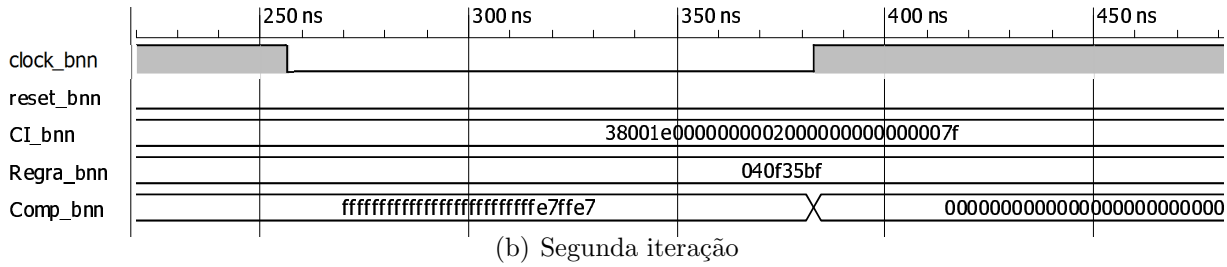


Figura 34: Arquitetura do componente `BNN`

Os dois primeiros ciclos de operação do componente BNN são apresentados na Figura 35. Quando habilitado, durante o período de inicialização, o componente BNN recebe os sinais de Clock_bnn e Reset_bnn enviados pelo componente CGA e interpreta os sinais Regra_bnn e CI_bnn configurados pelo usuário, conforme apresentado na Figura 35(a). Durante o período em que o valor do *clock* é igual a 0, as células deste componente realizam as operações de armazenamento necessárias a iteração, como por exemplo, o registro da configuração inicial e do conjunto de regras. Na primeira transição positiva do sinal Clock_bnn as células liberam os resultados da primeira iteração realizada. Estes resultados são enviados ao componente CGA através do sinal Comp_bnn. A regra utilizada nesta simulação, transferida para a arquitetura através do sinal Regra_bnn, foi a regra ótima para *grids* pares, publicada por *Fynn et al* em seu trabalho de análise de densidade (MARTINS; FYNN; OLIVEIRA, 2011), $040F35BF_{16}$. A configuração inicial, transferida para a arquitetura através do sinal CI_bnn foi realizada a partir da ativação aleatória de algumas células do *grid*. A segunda iteração realizada pelas células do componente BNN é apresentada na Figura 35(b). Todas as transferências de dados executadas pelo componente BNN ocorrem nas transições positivas do *clock*, até que seja alcançado o número programado de iterações e a unidade CCA seja desativada.



(a) Primeira iteração



(b) Segunda iteração

Figura 35: Simulação da operação do componente BNN

A leitura da regra utilizada permite a compressão de que a ocorrência de, por exemplo, uma das entradas $00000, 00001, 00010$, 00011 faz com que a célula seja ativada. Dada a configuração inicial e o conjunto de regras utilizado, analisando a evolução das células contidas na primeira linha do *grid*, obtém-se o resumo das primeiras evoluções deste CA, conforme

apresentado na Tabela 26. Utilizando a primeira célula do *grid* como referência, durante a inicialização, o seu estado atual e os de sua vizinhança geram a combinação 00000 fazendo com que a célula torne-se ativa. Na segunda iteração o estado atual da célula e o de sua vizinhança geram a entrada 00110 que mantém a célula ativa. Este resumo permite comprovar a diferença de evolução em função da configuração inicial do *grid*, uma vez que se forem analisadas outras iterações será possível evidenciar o início do *looping* logo após a segunda iteração.

Tabela 26: Resumo das primeiras iterações realizadas pelo componente `BNN`

Configuração Inicial	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
Primeira iteração	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
Segunda iteração	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Da forma similar aos componentes `UWN` e `UWI` o funcionamento e a arquitetura do componente `BNI` são idênticos aos descritos para o componente `BNN`.

4.3.2.3 Autômato Celular Bidimensional de Moore

O modelo do conjunto *vizinhança* proposto por *Moore*, conforme descrito no Capítulo 1, é uma expansão do modelo de conjunto *vizinhança* proposto por *Neuman*, uma vez que o modelo de *Moore* é formado pela *vizinhança* proposta por *Neuman* acrescida das quatro células diagonalmente adjacentes a célula em análise dentro de um ambiente bidimensional, transformando todas as células imediatamente no entorno de outra célula elementos de seu conjunto *vizinhança*. Portanto, no caso do *grid* bidimensional de *Moore* utilizado nesta dissertação, onde foi considerado $r = 1$, as triplas (i, j, k) e (M, N, P) podem ser substituídas respectivamente pelas triplas $(i, j, 1)$ e $(M, N, 1)$. A correta localização dos elementos pertencentes ao conjunto *vizinhança* de cada célula no modelo de *Moore* dependerá, de forma equivalente ao modelo de *Neuman*, exclusivamente dos valores de (i, j) e (M, N). A diferença entre os modelos de *vizinhança* de *Neuman* e de *Moore* é o número de elementos que constituem o conjunto *vizinhança*, que aumenta de quatro, no modelo de *Neuman*, para oito, no modelo de *Moore*. A Equação 11 apresenta a função para localização dos oito elementos que compõem o conjunto *vizinhança* de *Moore*:

$$V_{i,j} = \{V_{f_{(i,M)},j}, V_{f_{(i,M)},g_{(j,N)}}, V_{i,g_{(j,N)}}, V_{g_{(i,M)},g_{(j,N)}}, V_{g_{(i,M)},j}, V_{g_{(i,M)},f_{(j,N)}}, V_{i,f_{(j,N)}}, V_{f_{(i,M)},f_{(j,N)}}\} \tag{11}$$

Esta versão estendida da função $V_{i,j}$ também abrange a resolução do problema de borda através da ligação entre as células da extremidade, eliminando possíveis valores incorretos

resultantes da localização errada dos elementos do conjunto *vizinhança*. Um exemplo do modelo de interligação utilizado através da aplicação da função $V_{i,j}$ é apresentado na Figura 36, que também utiliza a cor preta para indicar a ligação entre células.

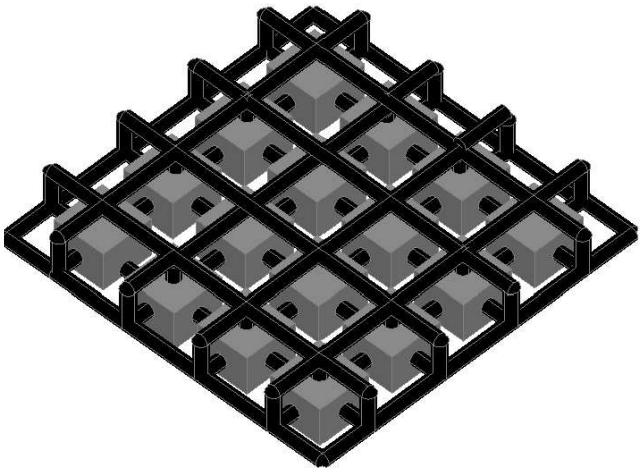


Figura 36: Organização estrutural do componente `BMN`

Na Figura 36, que representa um CA com 16 células, 4×4, com $r = 1$, é possível visualizar a sequência completa de interligação da *vizinhança* de *Moore* e a resolução do problema de borda. Neste exemplo, a aplicação da versão estendida da função $V_{i,j}$, apresentada na Equação 11, torna possível a localização correta dos oito elementos que compõem o conjunto *vizinhança* de cada uma das células do *grid*, como por exemplo:

1. $V_{1,1} = \{V_{4,1}, V_{4,2}, V_{1,2}, V_{2,2}, V_{2,1}, V_{2,4}, V_{1,4}, V_{4,4}\}$;
2. $V_{1,4} = \{V_{4,4}, V_{1,4}, V_{1,1}, V_{2,1}, V_{1,4}, V_{2,3}, V_{1,3}, V_{4,3}\}$;
3. $V_{2,3} = \{V_{1,3}, V_{1,4}, V_{2,4}, V_{3,4}, V_{3,3}, V_{3,2}, V_{2,2}, V_{1,2}\}$;
4. $V_{3,2} = \{V_{2,2}, V_{2,3}, V_{3,3}, V_{4,3}, V_{4,2}, V_{1,4}, V_{3,1}, V_{2,1}\}$;
5. $V_{4,1} = \{V_{3,1}, V_{3,2}, V_{4,2}, V_{1,2}, V_{1,1}, V_{1,4}, V_{3,4}, V_{3,1}\}$;
6. $V_{4,3} = \{V_{3,3}, V_{4,4}, V_{3,1}, V_{1,4}, V_{1,3}, V_{1,2}, V_{4,2}, V_{3,2}\}$.

A arquitetura do componente `BMN` é apresentada na Figura 37, na qual as células são representadas pela denominação $C_{i,j}$. Quando este módulo é habilitado, ele recebe os sinais Clock_bmn e Reset_bmn. Durante o período de inicialização, os valores contidos nos sinais Regra_bmn e CI_bmn são armazenados para que as células contidas neste componente possam efetuar as operações necessárias a realização das iterações programadas. Ao término de cada

iteração, o estado atual de cada célula é liberado para o componente GCA através do sinal Comp_bmn. As ligações entre as células, para melhor visualização da arquitetura, foram omitidas na Figura 37, sendo realizada apenas a indicação do fluxo de ligações entre células e destacados os sinais de interface do componente BMN.

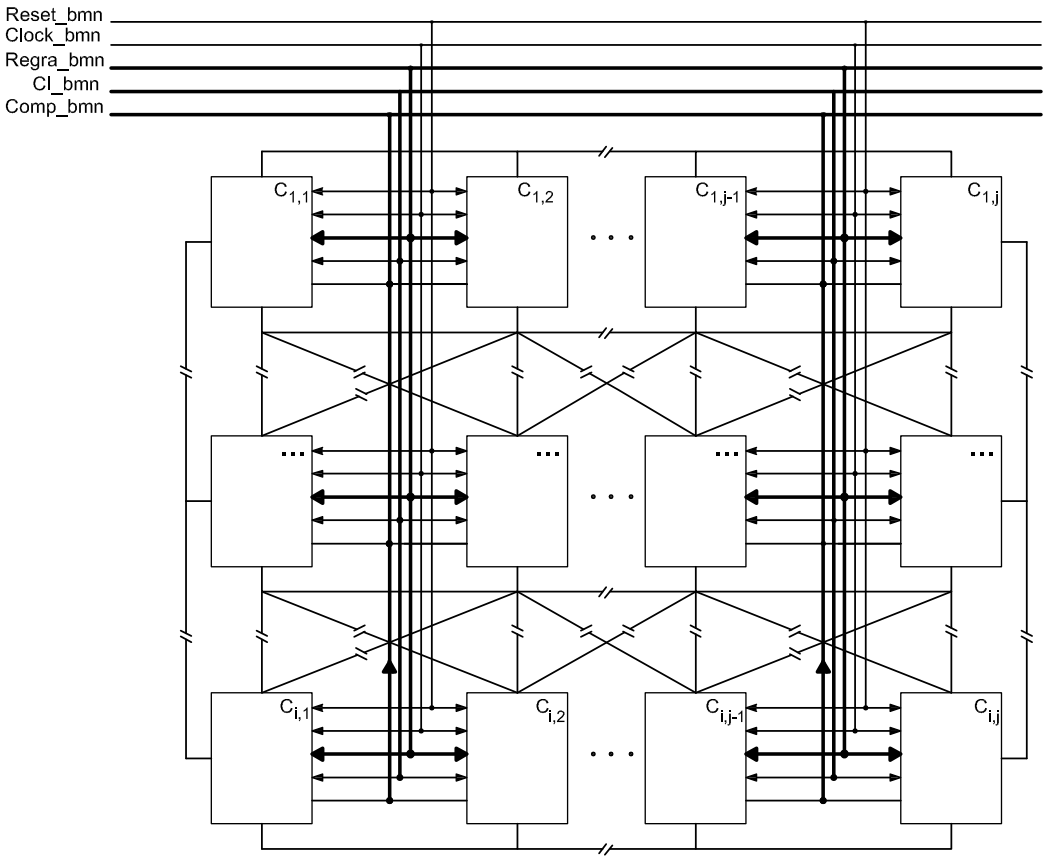
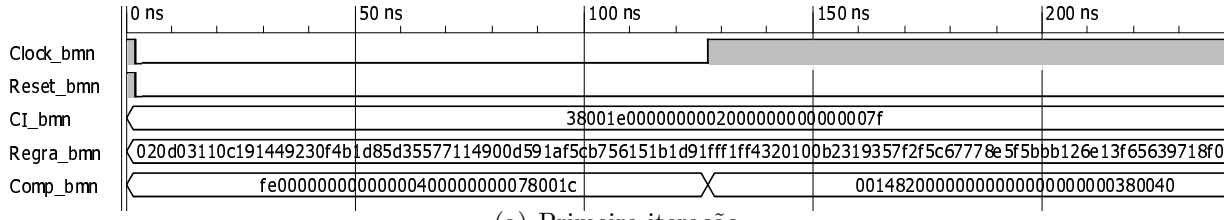


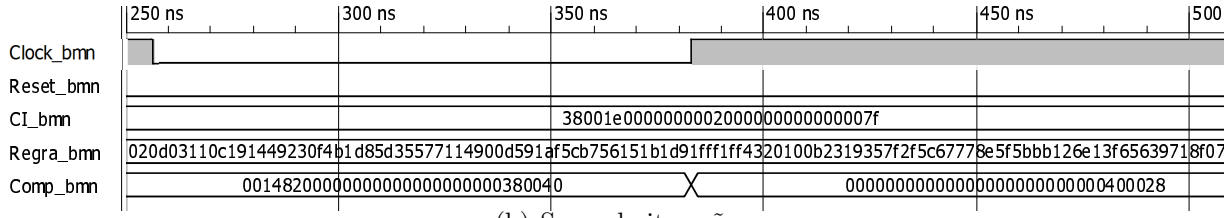
Figura 37: Arquitetura do componente BMN

Os dois primeiros ciclos de operação do componente BMN são apresentados na Figura 38. Quando habilitado, durante o período de inicialização o componente BMN recebe os sinais de Clock_bmn e Reset_bmn, enviados pelo componente CGA , interpretando os sinais Regra_bmn e CI_bmn, configurados pelo usuário, conforme apresentado na Figura 38(a). Durante o período em que o valor do *clock* é igual a 0, as células deste componente realizam as operações de armazenamento necessárias a iteração. Na primeira transição positiva do sinal Clock_bmn as células liberam os resultados da primeira iteração realizada. Estes resultados são enviados ao componente CGA através do sinal Comp_bmn. A regra utilizada nesta simulação, transferida para a arquitetura através do sinal Regra_bmn, foi a regra ótima para *grids* pares, $020D03110C191449230F4B1D85D35577114900D591AF5CB756151B1D91FFF1FF4$ $320100B2319357F2F5C67778E5F5BBB126E13F65639718F076B7CF7CED75777_{16}$, publi-

cada por *Fynn et al* no trabalho de análise de densidade, (MARTINS; FYNN; OLIVEIRA, 2011). A configuração inicial, transferida para a arquitetura através do sinal CI_bmn foi a mesma configuração aleatória utilizada para composição da Figura 35. A segunda iteração realizada pelas células do componente BMN é apresentada na Figura 38(b).



(a) Primeira iteração



(b) Segunda iteração

Figura 38: Simulação da operação do componente BMN

Todas as transferências de dados do componente BNN ocorrem nas transições positivas do *clock*, até que seja alcançado o número programado de iterações e a unidade CCA seja desativada. A leitura da regra utilizada na simulação permite a compressão de que qualquer uma das entradas 000000000, 000000001 , 000000010 fazem com que a célula seja ativada. Dada a configuração inicial e o conjunto de regras utilizado, analisando a evolução das células contidas na primeira linha do *grid*, obtém-se o resumo das primeiras evoluções deste CA, conforme apresentado na Tabela 27. Utilizando a primeira célula do *grid* como referência, durante a inicialização, o seu estado atual e os de sua vizinhança geram a combinação 100000000 fazendo com que a célula, que estava inativa, torne-se ativa. Contudo na segunda iteração o estado atual da célula e o de sua vizinhança geram a entrada 000010100 e a célula passa do estado ativo para o estado inativo. Este resumo permite comprovar que, o modelo do conjunto *vizinhança* e conjunto de regras interferem diretamente na geração de resultados.

Tabela 27: Resumo das primeiras iterações realizadas pelo componente BMN

Configuração Inicial	0	0	1	1	1	0	0	0	0	0	0	0	0	0	0	0
Primeira iteração	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	0
Segunda iteração	0	0	0	0	0	0	0	0	0	0	0	1	0	1	0	0

O funcionamento e a arquitetura do componente `BMN` são idênticos aos descritos para o componente `BMI`.

4.3.2.4 Autômato Celular Tridimensional de Neuman

Conforme detalhado ao longo do Capítulo 1, ao criar um autômato celular um projetista pode elaborar o tipo de modelo de modelo estrutural que desejar para seu projeto, desde que sejam definidas corretamente o modelo de ligação entre células e as condições de fronteira. Esta ideia, que permite a confecção de autômatos celulares *n dimensionais*, foi aplicada para criação do autômato tridimensional utilizado nesta dissertação. Autômatos tridimensionais podem ser criados a partir da idealização de novos modelos de conjunto *vizinhança* ou da através da expansão dos modelos de conjunto *vizinhança* existentes. Um exemplo de expansão de um modelo de conjunto *vizinhança* existente foi realizado através do aumento do número de elementos pertencentes ao conjunto *vizinhança*, de forma a efetuar a transição da estrutura de *grid* do ambiente bidimensional para o tridimensional. Este modelo de *vizinhança* expandido é a *vizinhança* tridimensional de *Neuman*. Esta expansão aumenta o número de elementos do conjunto *vizinhança* de quatro, utilizado no ambiente bidimensional, para seis elementos, aplicado no ambiente tridimensional. Esta expansão é realizada através da inclusão das células imediatamente inferior e superior a célula em análise, conforme exemplificado na estrutura apresentada na Figura 39, que também apresenta a ordem em que os elementos do conjunto vizinhança serão analisados.

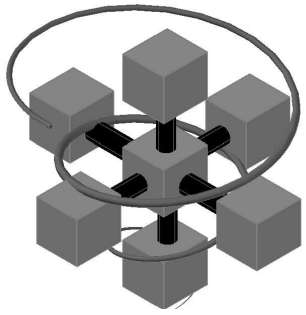


Figura 39: Organização estrutural do modelo de vizinhança utilizado no componente `TNN`

De forma idêntica aos demais *grids* apresentados no decorrer deste capítulo, o *grid* tridimensional de *Neuman* também pode ser definido de forma matricial. Portanto, no caso do *grid* tridimensional utilizado neste trabalho, dado que o conjunto *vizinhança* possui seis elementos, a correta localização dos elementos horizontais e verticais contidos no conjunto *vizinhança* dependerá do valores de (i, j, z) e (M, N, P). A Equação 12 apresenta a função

de localização dos elementos que compõem o conjunto *vizinhança* tridimensional de *Neuman* utilizado neste trabalho.

$$V_{i,j,k} = \{V_{i,j,f_{(k,P)}}, V_{f_{(i,M)},j,k}, V_{i,g_{(j,N)},k}, V_{g_{(i,M)},j,k}, V_{i,f_{(j,N)},k}, V_{i,j,g_{(k,P)}}\} \quad (12)$$

Um exemplo do modelo de interligação utilizado através da aplicação da função $V_{i,j,k}$ é apresentado na Figura 40. Neste exemplo a ligação entre células é apresentada na cor preta.

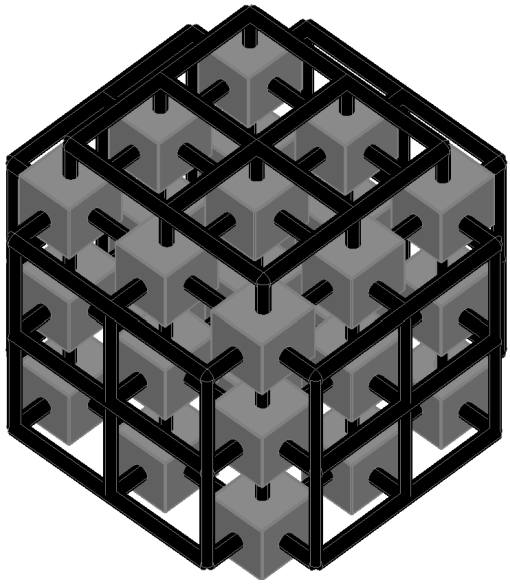


Figura 40: Organização estrutural do componente `TNN`

Na Figura 40, que representa um CA com 27 células, $3 \times 3 \times 3$ e com $r = 1$, é possível visualizar a sequência completa de interligação da *vizinhança* tridimensional de *Neuman* e a resolução do problema de borda. A aplicação da função $V_{i,j,k}$, apresentada na Equação 12, torna possível a correta localização dos elementos que compõem o conjunto *vizinhança* de cada uma das células do *grid*, como por exemplo:

1. $V_{1,1,1} = \{V_{1,1,3}, V_{4,1,1}, V_{1,2,1}, V_{2,1,1}, V_{1,4,1}, V_{1,1,2}\}$;
2. $V_{2,3,3} = \{V_{2,3,2}, V_{1,3,3}, V_{2,4,3}, V_{3,3,3}, V_{2,2,3}, V_{2,3,1}\}$;
3. $V_{4,3,2} = \{V_{4,3,1}, V_{3,3,2}, V_{4,4,2}, V_{1,3,2}, V_{4,2,2}, V_{4,3,3}\}$.

A arquitetura do componente `TNN` é apresentada na Figura 41, na qual as células são representadas pela denominação $C_{i,j,k}$. Para melhor visualização, as ligações entre células foram omitidas, sendo destacados apenas os sinais de interface do componente `TNN`. Quando

este módulo é habilitado, ele recebe os sinais Clock_tnn e Reset_tnn. Durante o período de inicialização, os valores contidos no sinais Regra_tnn e CI_tnn são armazenados para que as células contidas neste componente possam efetuar as operações necessárias a realização das iterações programadas. Ao término de cada iteração, o estado atual de cada célula é liberado para o componente GCA através do sinal Comp_tnn.

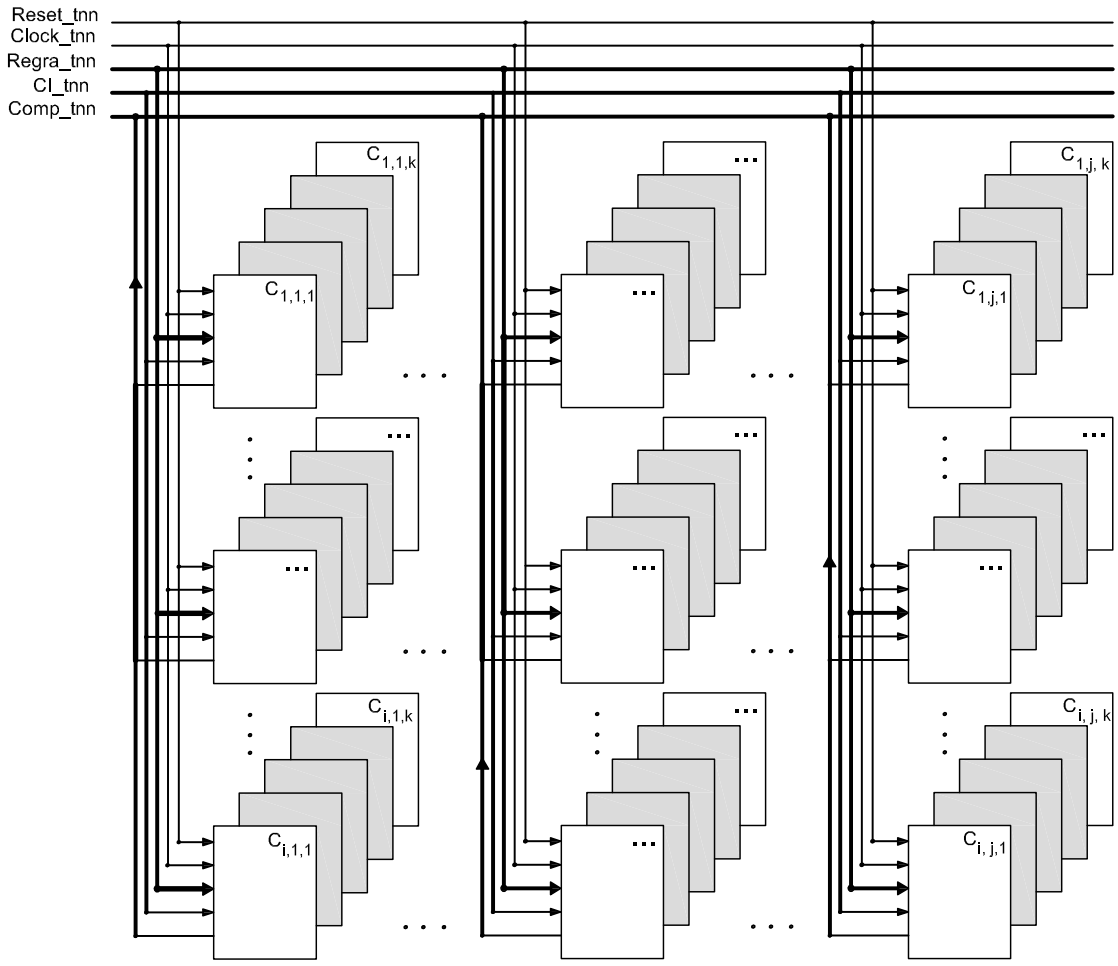
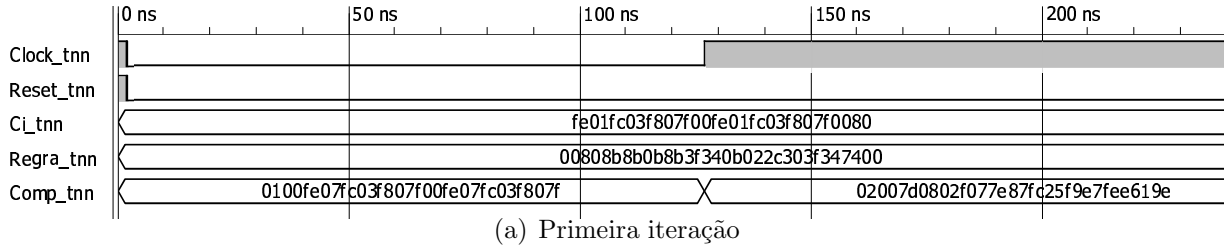


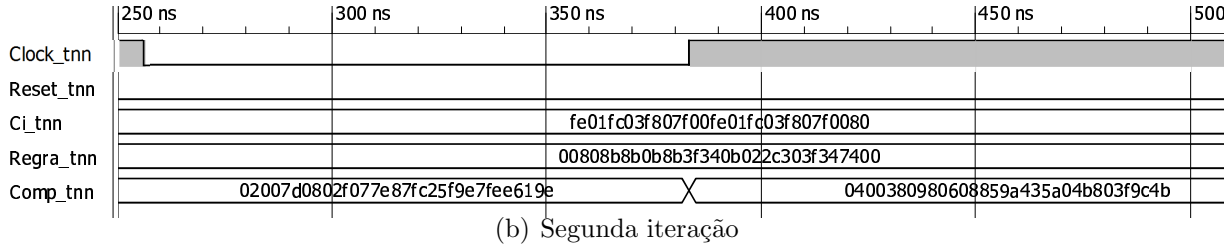
Figura 41: Arquitetura do componente TNN

Os dois primeiros ciclos de operação do componente TNN são apresentados na Figura 42. Quando habilitado, durante o período de inicialização, o componente TNN recebe os sinais de Clock_tnn e Reset_tnn, enviados pelo componente CGA, interpretando os sinais Regra_tnn e CI_tnn, configurados pelo usuário, conforme apresentado na Figura 42(a). Durante o período em que o valor do clock é igual a 0, as células deste componente realizam as operações de armazenamento necessárias a iteração. Na primeira transição positiva do sinal Clock_tnn as células deste CA liberam os resultados da primeira iteração realizada. Estes resultados são enviados ao componente CGA através do sinal Comp_tnn. O conjunto de regras utilizado nesta simulação,

transferido para a arquitetura através do sinal Regra_tnn, foi um adaptação realizada da versão tridimensional do *Game of Life* de *Conway* (BAYS, 1987, 1991). Esta adaptação, em função do menor número de vizinhos, consistiu na utilização do conjunto de proposições utilizadas no *Game of Life 2D*, (CONWAY, 1970), $00808B8B0B8B3F340B022C303F347400_{16}$, reduzindo-se assim o número necessário de vizinhos para que a célula viva ou morra. A configuração inicial utilizada, transferida através do sinal Ci_tnn, foi realizada a partir da ativação aleatória de algumas células do *grid*, de forma a testar a validade dos resultados emitidos por este componente.



(a) Primeira iteração



(b) Segunda iteração

Figura 42: Simulação da operação do componente `TNN`

De forma simular aos demais CAs utilizados nesta dissertação, o funcionamento e a arquitetura do componente `TNN` são idênticos aos descritos para o componente `TNI`. A leitura da regra utilizada permite a compressão de que as entradas $0000000, 0000001, 0000010, 0000011$ fazem com que a célula seja desativada. Dada a configuração inicial e o conjunto de regras utilizado, analisando a evolução das células contidas na primeira linha do *grid*, de $C_{1,1,1}$ a $C_{1,8,1}$, obtém-se o resumo das primeiras evoluções deste CA, conforme apresentado na Tabela 28. Utilizando a primeira célula do *grid* como referência, durante a inicialização, o seu estado atual e os de sua vizinhança geram a combinação 0000000 fazendo com que a célula continue inativa. Na segunda iteração, a combinação de estados repete a entrada 00000 que mantém a célula inativa.

Tabela 28: Resumo das primeiras iterações realizadas pelo componente `TNN`

Configuração Inicial	0	0	1	1	1	0	0
Primeira iteração	1	1	1	1	1	1	1
Segunda iteração	0	0	0	0	0	0	0

4.4 Célula do CA

Conforme descrito ao longo do Capítulo 1 deste trabalho, uma célula de um autômato celular precisa possuir pelo menos três características, guardar todos os estados possíveis do sistema, manter e compartilhar o seu estado atual até a próxima iteração e receber e analisar os estados de seu conjunto *vizinhança*, de forma a obter as informações necessárias para o cálculo de seu próximo comportamento, que poderá ser migrar para outro estado ou manter o estado atual. A metodologia utilizada para formulação destes requisitos, bem como os parâmetros de análise da correta realização dos mesmos, fica a cargo do projetista, variando em função dos objetivos do sistema.

A liberdade para a elaboração de um projeto que envolva utilização de CAs é tanta que em alguns casos, o projetista pode apenas emular os dados que seriam resultantes de um determinado número de evoluções de um CA, sem ter de se preocupar em analisar individualmente os resultados de cada célula do *grid* ao longo da evolução, preservando a compreensão de que o sistema só funcionará corretamente se as características emuladas corresponderem ao que seria realizado por cada célula individualmente. Este tipo de estratégia é possível porque todas as células são idênticas em constituição e funcionamento e em algumas aplicações, cujo objetivo não é o resultado individual de cada iteração corrente, mas, sim o resultado final após um determinado número de iteração, não se faz necessária a implementação individual de todas as células integrantes do *grid*.

A consecução dos objetivos propostos nesta dissertação demanda os resultados de cada uma das células durante, todas as iterações programadas, uma vez que estes serão utilizados diretamente para composição musical. Neste caso, tornou-se necessária a criação de uma estrutura fundamentada em quatro parâmetros. O primeiro parâmetro é que a célula proposta teria de ser facilmente replicável, uma vez que a linguagem de descrição de *hardware* é menos flexível que qualquer linguagem de programação. O segundo parâmetro é que ao longo do tempo, a célula proposta teria de manter as características básicas de uma célula de autômato celular. O terceiro parâmetro é que a célula proposta deveria permitir a realização de novas configurações, uma vez que um dos objetivos deste trabalho é criar uma estrutura para estudos futuros em aplicações de áreas diversas. E, o quarto parâmetro é o de que a célula proposta

necessita ter uma arquitetura simples, de forma a reduzir ao máximo o custo computacional, uma vez que esta célula precisará ser replicada tantas vezes quanto for necessário para alcançar as estruturas de *grid* pretendidas. Deste modo, a célula projetada teria de obedecer a um ciclo rígido de funcionamento, com a finalidade de evitar qualquer tipo de atraso que comprometesse a integridade dos dados recebidos ou enviados pelos componentes da arquitetura.

De forma a atender estas características, após vários testes com estruturas similares, optou-se pela construção da arquitetura apresentada na Figura 43. Nesta arquitetura o único sinal que não é de tamanho 1 *bit* é o sinal Regra, que contém os dados do conjunto de regras que será utilizado pela célula como base para análise dos parâmetros de sua evolução. Os três componentes que formam a célula projetada são o `DECODER`, o `ROMC` e o `FFD`.

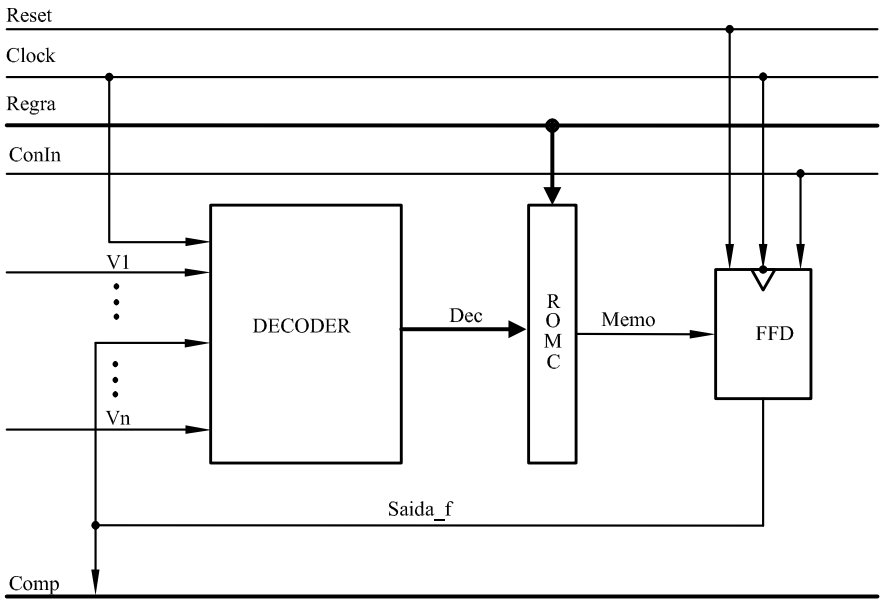


Figura 43: Arquitetura da Célula

O componente Decodificador, `DECODER`, é responsável por receber os valores de *vizinhança*, analisá-los e ativar a liberação do valor verdade para o cruzamento *vizinhança* × conjunto de regras. Este cruzamento é feito com base nos estados atuais da vizinhança, recebidos nos sinais V, e pelo estado atual da célula, recebido através do sinal Saida_f. O componente Memória Rom, `ROMC`, é uma memória do tipo ROM, $n \times 1$, responsável por armazenar o conjunto verdade, ou seja os estados, previstos pelo conjunto de regras. O tamanho do endereçamento da memória interna do componente `ROMC` foi identificado como $n \times 1$ pelo fato de que, como cada célula possui no máximo dois estados possíveis, $\{0, 1\}$, e que o estado atual da célula influência no resultado da iteração, o número de linhas de memória irá variar em função do

modelo de vizinhança, sendo 8 linhas para o modelo de vizinhança de *Wolfram*, 32 linhas para o modelo de vizinhança de *Neuman*, 512 linhas para o modelo de vizinhança de *Moore* e 128 linhas para o modelo de vizinhança tridimensional de *Neuman*. A ativação do componente ROMC é realizada através do sinal Dec, oriundo do componente DECODER. Este sinal, entregue a cada iteração, contém o endereço da linha de memória que deverá ser lida. O componente Flip Flop, FFD, é o *flip-flop D* responsável por armazenar e distribuir o estado atual de célula, recebido através do sinal Memo. A cada ciclo de operação, os estados atuais das células são transmitidos as suas respectivas vizinhança e o componente CGA através do barramento Comp, que armazena o valor contido no sinal Saida_f.

4.5 Máquina MIDI

A arquitetura da Máquina MIDI, MMIDI, apresentada na Figura 44, é formada por nove componentes diferentes, sendo dois destes memórias dedicadas. O componente Gestor MIDI, GMIDI, é o componente responsável pelo controle dos demais componentes da unidade MMIDI, coordenando as operações entre estes e a gestão do processamento dos dados oriundos unidade CCA. O componente Controle Notas, CNO é uma memória do tipo ROM, 128×8, que armazena uma tabela com os códigos de todas as notas aceitas pelo protocolo MIDI (STANDARD, 1999). O componente Controle Instrumentos, CNI é uma memória do tipo ROM, 128×40, que armazena a tabela com os códigos de todos os instrumentos aceitos pelo protocolo MIDI (ASSOCIATION, 2013). O componente Temporizador, TEM, é responsável por entregar ao componente CNO os dados de volume e duração de execução de notas, que serão utilizados na composição do produto musical. O componente Metrônomo, MET, armazena os valores de fórmula de compasso e metrônomo, que ditarão o ritmo da composição musical. O componente Acumulador, ACC, é responsável pela contabilização do número total de instrumentos e notas, escolhidos durante o processo de composição musical. O componente Divisor, DIV é responsável pelos cálculos de equilíbrio, entre o total de notas e instrumentos gerados, para que seja possível o fechamento do arquivo. O componente Armazém, AZM, é formado por duas memórias do tipo RAM, cujos tamanhos serão abordado no detalhamento deste componente. Estas memórias são utilizadas para armazenar as notas, instrumentos musicais e os códigos de ativação e desativação que serão utilizados na composição musical. O componente Aglutinador, AGT é uma memória do tipo RAM, cujo tamanho será abordados no detalhamento deste componente, que possui a finalidade de organizar os dados oriundos do componente AZM efetuando a criação do produto musical que deverá ser convertido para a extensão *.mid*.

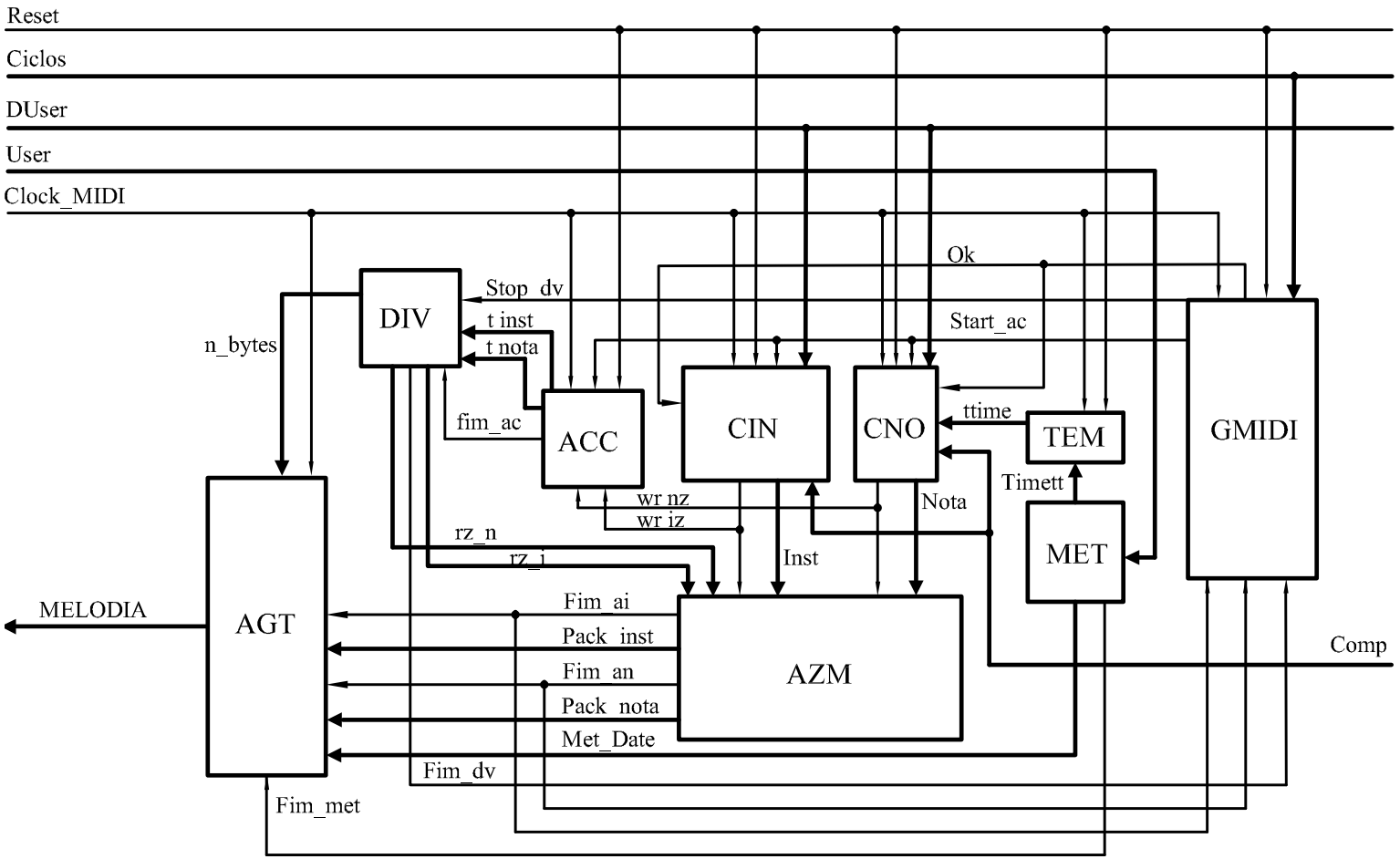


Figura 44: Arquitetura da unidade Máquina MIDI

As especificidades de cada componente da unidade MMIDI serão descritas ao longo desta seção. O fluxo de sinais de interface da unidade MMIDI é composto pelo sinal Clock_midi e por outros cinco sinais externos. O sinal Reset permite a inicialização da unidade MMIDI. O sinal Ciclos, cujo valor é configurado pelo usuário, contém o número total de iterações que a unidade CCA deverá executar, informando a unidade MMIDI os dados necessários ao cálculo do total de ciclos de operação que deverão ser realizados. O sinal DUser contém a identificação dos CAs que foram habilitados pelo usuário, permitindo a unidade MMIDI a triagem dos dados recebidos através do sinal Comp, separando os trechos referentes aos CAs habilitados. O sinal User, configurado pelo usuário, contém a informação de qual opção do componente MET deverá ser habilitada. E, o sinal Melodia exporta o produto musical para o meio externo, para que este possa ser convertido na extensão *.mid*. Dada a diversidade e a quantidade de sinais presentes na unidade MMIDI, o detalhamento do fluxo dos sinais internos desta unidade será realizado em conjunto com o detalhamento dos componentes.

4.5.1 Gestor MIDI

O componente Gestor MIDI, GMIDI, é a unidade responsável por controlar, através da análise do sinal Ciclos, os ciclos que deverão ser executados pelos componentes da unidade MMIDI. Conforme citado na Seção 4.2, para que a arquitetura atue de forma síncrona, a frequência de operação dos componentes da unidade máquina MIDI precisa ser 128 vezes mais rápida que a dos componentes da unidade CCA. Esta variação de frequência é informada para a arquitetura através da multiplicação apresentada na Equação 13.

$$Ciclos_{MMIDI} = total\,de\,ciclos \times 128 \tag{13}$$

O componente GMIDI também é responsável por autorizar o início do funcionamento dos componentes CNI e CNO, através do sinal OK, por autorizar o início do processo de escrita do arquivo *.txt*, através do sinal Start_ac, e por confirmar o início do processo de encerramento do arquivo *.txt* através do sinal Encerra. A arquitetura do componente GMIDI é apresentada na Figura 45. Internamente este componente possui uma unidade de controle, ParM, que atua autorizando o funcionamento de alguns dos componentes chave da unidade MMIDI, um registrador, RCiclos, que armazena o número total de ciclos, e um contador, Count, que controla o tempo de espera para o início da atuação dos componentes CNI e CNO e também o número total de ciclos de operação da unidade MMIDI. Todas as autorizações de funcionamento oriundas da unidade MMIDI são efetuadas pelo componente ParM, que também recebe os avisos de encerramento dos componentes chave da unidade MMIDI.

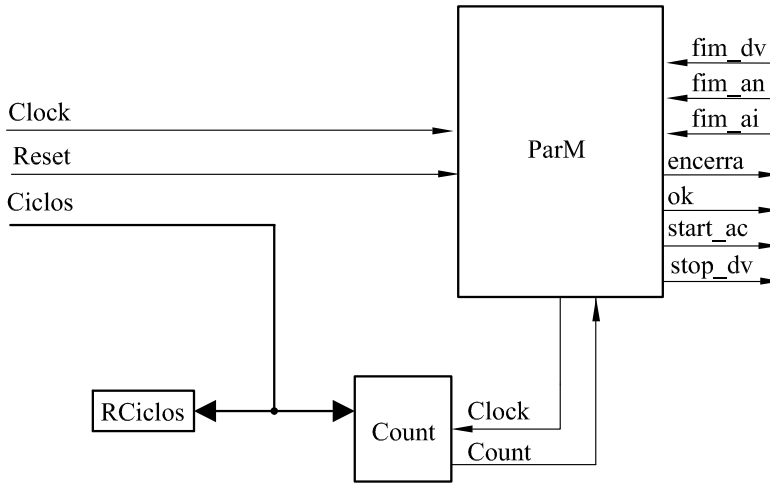


Figura 45: Arquitetura do componente GMIDI

As Figuras 46, 47 e 48 apresentam o funcionamento do componente GMIDI através do seu diagrama de transição. Durante o período de inicialização da arquitetura, Figura 46, o componente GMIDI recebe o valor do sinal Ciclos e o armazena até que a arquitetura seja autorizada a iniciar o processamento. Após a autorização de início de operação, que acontece quando sinal Reset retorna a 0, o componente Count controla o tempo de espera, até que a primeira transição positiva de *clock* seja realizada pelo componente CCA. Quando esta transição positiva ocorre, o sinal Count é ativado, informando ao componente ParM que os componentes CNI e CNO deverão ser habilitados.

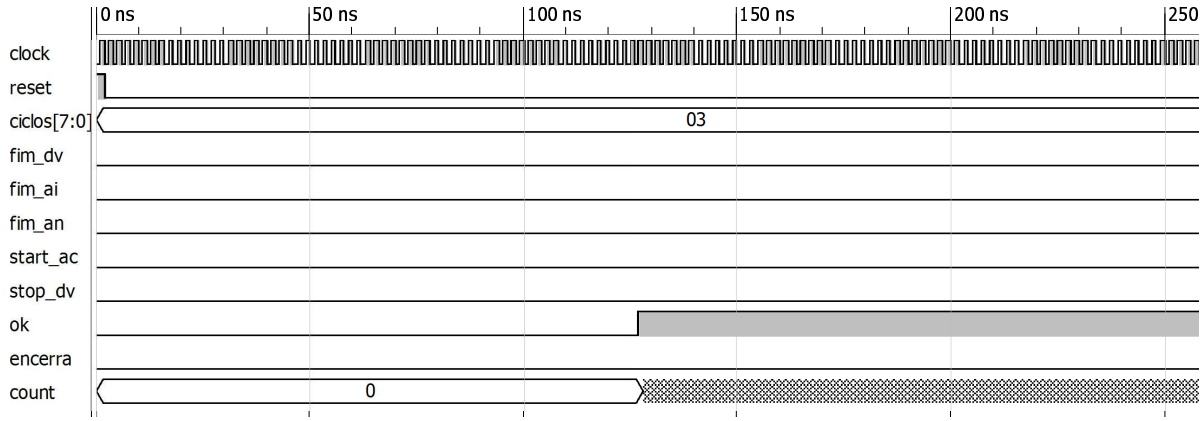


Figura 46: Processo de inicialização do componente GMIDI

Conforme apresentado na Figura 47, ao atingir o total definido de ciclos, que é apurado conforme definido na Equação 13, o componente ParM, através do sinal Start_ac, solicita ao componente ACC que libere os totais apurados para o componente DIV.

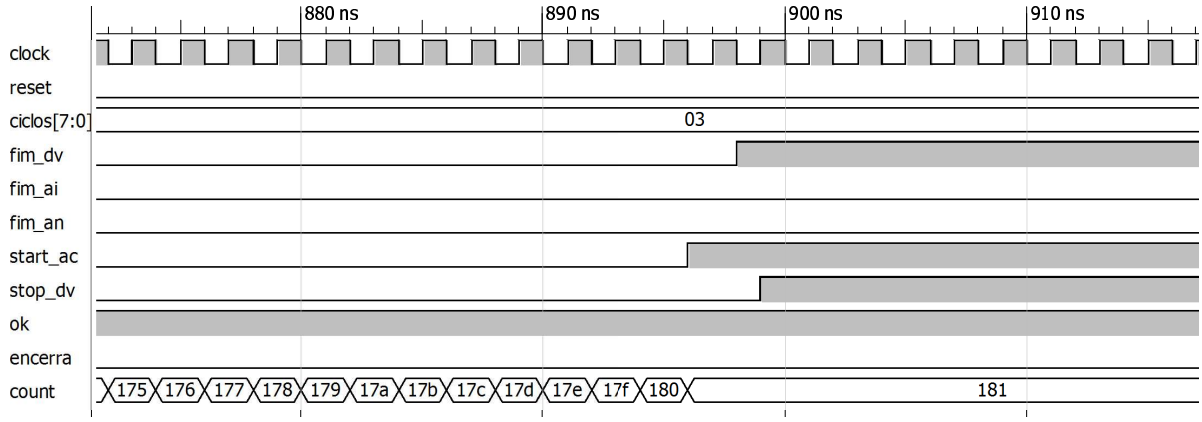
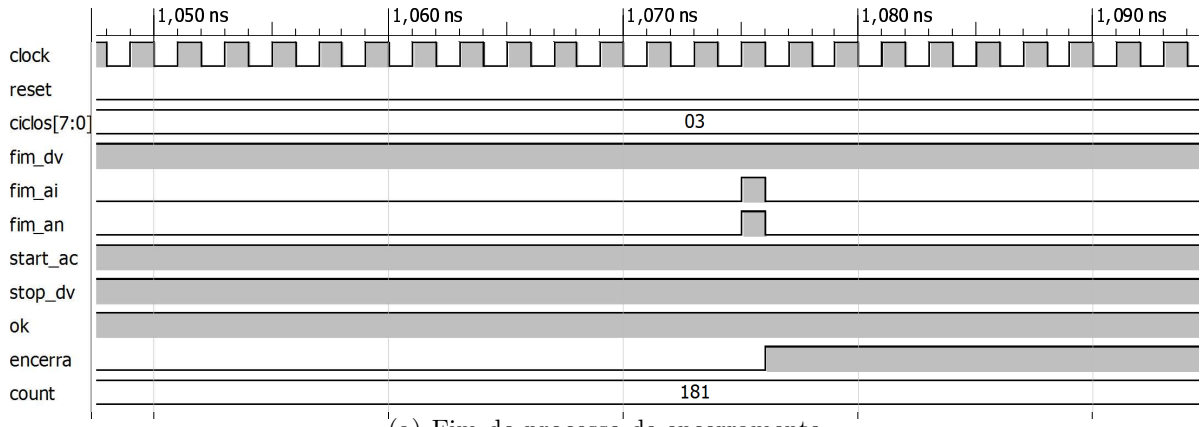


Figura 47: Início do processo de encerramento do componente GMIDI

Ao término do processamento dos dados recebidos, o componente DIV emite um alerta para o componente ParM, através do sinal Fim_dv, recebendo através do sinal Stop_dv autorização para encerrar seu funcionamento. Quando o componente AZM encerra o processo de leitura das notas e instrumentos armazenados ao longo de sua operação, os sinais Fim_ai e Fim_an são enviados ao componente ParM para que este autorize o fechamento do arquivo e o encerramento das operações da unidade MMIDI, conforme apresentado na Figura 48.



(a) Fim do processo de encerramento

Figura 48: Operação simulada do componente GMIDI, processo de encerramento

4.5.2 Metrônomo

Ativado pelo sinal User, de tamanho 5 *bits*, o componente metrônomo, MTN, é o decodificador responsável por enviar o código hexadecimal que conterá os dados inerentes a configuração do andamento musical, fornecendo de acordo com a escolha do usuário os dados que indicam o ritmo do produto musical que será gerado. A informação emitida pelo componente MTN é

repassada ao componente AGT, através do sinal Met_date, de tamanho 11 *bytes*, e ao componente TEM, através do sinal Timett, de tamanho 13 *bytes*. A definição das informações contidas no sinal Met_date e no sinal Timett, estruturada por *bits*, são respectivamente.

1. 1 *byte* para *delta time*, com valor 00;
2. 1 *byte* para o número de PPQ, com a configuração 96;
3. 1 *byte* para indicar o tamanho da informação de compasso e 2 *bytes* para a formula do compasso, que possui como opções de configuração 1\2, 1\4, 4\4, 5\4 e 3\8;
4. 2 *bytes* para indicação de fusas por unidade de tempo, em semínimas, com a opção de 8 fusas por semínima para os compassos base 2 e 4 ou 12 fusas por semínima para os compassos base 8;
5. 4 *bytes* para representar a quantidade de semínimas por minuto configurado em 96 *bpm*;
6. 2 *bytes* informando os valores de volume e altura que serão utilizados.

Durante a configuração da arquitetura, o usuário opta por uma das 5 configurações disponíveis, descritas na Tabela 29. A diferença entre cada uma das opções de configuração deste componente é a mudança rítmica que elas provocam, sendo a configuração 00000 a que gera melodias mais lentas e a configuração 10000 a que gera melodias mais rápidas.

Tabela 29: Opções de configuração para o componente MET

Opção	**informações enviadas a arquitetura**
00000	Compasso 1 \ 2, 96 *bpm*, volume 60 com altura variando entre 40 e 80
00010	Compasso 1 \ 4, 96 *bpm*, volume 60 com altura variando entre 40 e 80
00100	Compasso 4 \ 4, 96 *bpm*, volume 60 com altura variando entre 60 e 100
01000	Compasso 5 \ 4, 96 *bpm*, volume 60 com altura variando entre 20 e 60
10000	Compasso 3 \ 8, 96 *bpm*, volume 60 com altura variando entre 50 e 90

Durante a inicialização da arquitetura, a configuração escolhida pelo usuário é transferida ao componente MET através do sinal User, sendo de forma imediata assimilada por este componente. Após a assimilação, o valor acionado pelo sinal User é transferido quase que de forma imediata aos componentes TEM e AGT. O componente TEM recebe a informação de duração das notas através dos sinal Timett. Para o componente AGT são transmitidas, através do sinal Met_date, as demais informações de ritmo, tais como PPQ e fórmula do compasso.

4.5.3 Temporizador

O componente Temporizador, TEM, é o responsável pela definição das alturas e durações das notas executadas. Para tal, este componente, que é uma máquina de estados, durante o período de inicialização, recebe os dados selecionados pelo usuário, através do sinal Timett. No decorrer de toda a sua operação os valores da máquina de estados, que podem variar de 1 a 4, são alterados em função da continuidade ou não da sequência de notas, liberando-os para o componente CNO o par volume × altura que deverá ser utilizado para composição da melodia, conforme o conjunto de regras apresentado na Tabela 30. A variável x apresentada nesta tabela indica o estado no qual a máquina se encontra. Caso a próxima nota a ser emitida seja subsequente a anterior, a máquina muda para o estado seguinte. Caso contrário, a máquina retorna ao estado inicial. Este processo é realizado até que seja atingido o número total de iterações previstas para a unidade CCA. Esta estratégia permite a variação das alturas e volumes das notas executadas, inserindo marcações rítmicas na melodia.

Tabela 30: Regras do componente TEM

Se	Então
$x = 1$	será liberado o primeiro par volume e altura
$x = 2$	será liberado o segundo par volume e altura
$x = 3$	será liberado o terceiro par volume e altura
$x = 4$	será liberado o quarto par volume e altura

4.5.4 Central Notas e Central Instrumentos

Os componentes Central Notas e Central Instrumentos, respectivamente CNO e CNI, são responsáveis pela conversão dos dados contidos no sinal Comp para a codificação definida no protocolo MIDI de notação musical e instrumental (STANDARD, 1999; ASSOCIATION, 2013). O funcionamento destes componentes, que são constituídos por memórias do tipo ROM, de tamanho respectivamente, 128 × 8 e 128 × 40, é idêntico e em paralelo, cabendo a cada um a leitura e análise de um segmento do sinal Comp.

As linhas de memória destes componentes armazenam os códigos do protocolo MIDI para notas e instrumentos, vinculando cada linha de memória a uma célula dos autômatos celulares habilitados, perfazendo 128 células para o endereçamento da notação musical e 128 células para o endereçamento da notação instrumental.

A cada transição positiva do sinal Clock_aut, os resultados de iteração corrente dos CAs habilitados são liberados aos componentes CNO e CNI, através do sinal Comp. Os componentes

CNO e CNI efetuam, *bit* a *bit*, a leitura dos segmentos válidos do sinal Comp. Quando o *bit* lido tem valor 1, significa que o resultado da iteração corrente do CA foi a ativação da célula. Neste caso, a linha de memória referente a esta célula é lida e os dados armazenados nesta linha são enviados ao componente AZM, através dos sinais Nota e Inst. Em paralelo a esta ação os sinais Wr_iz, no caso da leitura de instrumentos e Wr_nz, no caso da leitura de notas, são enviados simultaneamente ao componente AZM , para o informe de que o dado enviado deverá ser escrito, e ao componente ACC, para que este contabilize o número de notas e instrumentos que deverão ser emulados. Quando o *bit* lido tem valor 0, o resultado da iteração corrente foi a desativação da célula e os componentes CNO e CNI deverão efetuar a leitura do próximo *bit*, enviando através dos sinais Wr_nz e Wr_iz o valor 0, que é indicativo de que os dados contido nos sinais Nota e Inst deverão ser ignorados, não havendo contabilização pelo componente ACC.

A diferença na operação dos componentes CNO e CNI é em função do tamanho e finalidade dos sinais Nota e Inst. O sinal Nota é composto pelo código referente a nota que deverá ser executada e pelos valores de altura e volume para esta nota, oriundos do sinal Timett, como por exemplo, a codificação 006060, que significa a execução da nota 00, *Dó* $_{-1}$ com altura 60 e volume 60. Já o sinal Inst é composto pela codificação MIDI referente ao instrumento a ser executado, pela codificação do canal no qual deverá ser executado o instrumento e pelos comandos de ativação e desativação, como por exemplo, o código $C018009080$, que solicita que o instrumento 18, violão, seja executado no canal 0.

A inserção dos comandos de ativação e desativação no componente CNI deve-se ao fato de que, as solicitações de execução no protocolo MIDI, conforme escrito no Capítulo 2, são realizadas a partir do instrumento, constituindo-se sempre a seguinte instrução para o interpretador: Toque a nota X no instrumento Y durante um tempo total T.

4.5.5 Acumulador

O componente Acumulador, ACC, é o contador responsável por calcular a quantidade total de notas e instrumentos resultantes das iterações dos CAs presentes na arquitetura. Este cálculo, que ocorre enquanto o sinal Start_ac encontra-se com valor 0, é realizado durante a conversão dos resultados de iteração corrente em notas e instrumentos musicais, que ocorre nos componentes CNO e CNI. Durante a leitura do sinal Comp, a cada localização de célula ativa os componentes CNO e CNI efetuam o informe para o componente ACC através dos sinais Wr_nz e Wr_iz, sendo contabilizado +1 quando um destes dois sinais encontra-se com valor 1.

Ao término das iterações programadas, o componente GMIDI ativa o sinal Start_ac,

informando ao componente ACC que o número total de iterações foi realizado e que os dados acumulados no componente ACC ao longo das iterações podem ser liberados. A liberação destes dados para a arquitetura ocorre através dos sinais T_nota, que contém o valor total de notas, e T_inst, que contém o valor total de instrumentos. Estes sinais, de tamanho 4 *bytes*, são destinados ao componente DIV para que este efetue os cálculos necessários ao fechamento do arquivo musical.

4.5.6 Divisor de Instruções

Já o componente Divisor de Instruções, DIV, é o subtrator responsável por efetuar duas tarefas, essenciais para o fechamento do arquivo musical:

1. Calcular a diferença entre o número total de notas e o número total de instrumentos a serem emulados;

2. Efetuar a apuração do total geral de notas e instrumentos que serão emulados, para que seja possível efetuar o fechamento do arquivo.

A diferença apurada é enviada ao componente AZM, para que este efetue o correto endereçamento de leitura dos dados armazenados em sua memória RAM. Para tal o, caso o número de notas seja maior que o número de instrumentos o valor da diferença é enviado através do sinal Rz_iz, de tamanho 14 *bits*. Caso o número de instrumentos seja maior que o número de notas o valor da diferença é enviado através do sinal Rz_nz, de tamanho 14 *bits*. Já o total geral apurado é enviado ao componente AGT, através do sinal N_bytes, de tamanho 4 *bytes*. A confirmação de que estes três sinais podem ser utilizados é efetuada através do sinal Fim_dv, que também é enviado aos componentes AGT e AZM.

4.5.7 Armazém

O componente armazém, AZM, é o responsável por receber todos os dados gerados pelos demais componentes da unidade máquina MIDI e armazená-los temporariamente para posteriormente transmití-los ao componente AGT. Sua constituição é formada por duas memórias do tipo RAM, 16384×40. A primeira memória será utilizada para armazenar os códigos de notas enviadas pelo componente CNO, recebendo dados de escrita através do sinal Nota. A segunda memória será utilizada para armazenar os códigos de instrumentos enviados pelo componente CNI, recebendo seus dados de escrita através do sinal Inst.

A escrita nas memórias contidas no componente AZM é sequencial. O número de linhas de cada memória RAM foi estimado a partir do pior caso de evolução, que seria o das 128 células

possuírem o estado ativo no decorrer de 128 iterações, como o que acontece na regra 255 de *Wolfram*. O processo de escrita do componente AZM é realizado em função do recebimento dos sinais Wr_nz e Wr_iz. Durante este processo o componente AZM armazena os dados oriundos do componentes CNO e CNI, transferidos pelos sinais Nota e Inst.

O processo de leitura também é realizado de forma sequencial, do primeiro ao último endereço. Quando um dos sinais Rz_nz ou Rz_iz é diferente de zero, o componente AZM efetua a releitura dos dados armazenados na memória quantas vezes forem necessárias até que o número total de instrumentos seja igual ao número total de notas.

4.5.8 Aglutinador

O componente aglutinador, AGT, é uma memória do tipo RAM, de tamanho variável, responsável pelo armazenamento e organização das informações liberadas pelos demais componentes da unidade MMIDI. O tamanho deste componente varia em função da quantidade de notas e instrumentos que deverão ser emulados através do arquivo *.mid*. Quanto mais notas e instrumentos houver, maior será o número de linhas desta memória. Cada linha, de tamanho 1 *byte* teve seu tamanho determinado de forma a eliminar possíveis alocações incorretas de dados. Esta estratégia foi adotada em função do fato de que, em arquivos *.mid* informações incorretas nos campos de notas e instrumentos são ignoradas, mas informações incorretas nos campos de contagem de *bits*, ativação e desativação corrompem o arquivo, inutilizando todo o processamento realizado pela arquitetura.

Quando a arquitetura é iniciada, o componente AGT recebe os valores liberados pelo componente MET, enviados através do sinal met_date. Estes valores são armazenados no componente AGT até o momento de escrita do arquivo *.txt*, realizada através do sinal Melodia.

Durante o encerramento do componente DIV o valor total de *bits* a serem escritos no arquivo *.txt* é destinado ao AGT através do sinal N_bytes. Este valor é armazenado nas linhas de endereço referentes ao segmento *Length* do *Track* que contém as informações de notação musical e instrumental.

Quando o componente AZM inicia seu processo de leitura, os dados contidos nas linhas de memória deste componente recebidos pelo componente AGT através dos sinais pack_inst e pack_nota. Quando o componente AZM termina seu processo de leitura os sinais de encerramento Fim_ai e Fim_an são recebidos pelo AGT que inicia o processo de encerramento do arquivo. Este processo de encerramento ocorre através da escrita dos três últimos *bytes* do arquivo, que deverão ser, conforme explicitado ao longo do Capítulo 2, $FF2F00_{16}$.

4.6 Considerações Finais do Capítulo

Neste capítulo foi apresentada a arquitetura proposta através do detalhamento da microarquitetura de cada um de seus componentes fundamentais. A descrição da arquitetura proposta foi complementada através da apresentação de simulações de seu funcionamento, realizadas a partir da utilização dos conjuntos de regras propostos por *Wolfram*, *Conway* e *Fyyn et al.* e das configurações iniciais descritas ao longo deste capítulo. No capítulo seguinte são apresentados alguns dos resultados obtidos durante a confecção desta dissertação.

Capítulo 5

ANÁLISE DOS RESULTADOS

NESTE capítulo são apresentados os resultados obtidos através de simulações da arquitetura descrita no Capítulo 4. Estas simulações, realizadas a partir de algumas das opções de configuração, objetivaram a comprovação do desempenho da arquitetura proposta e a análise da qualidade e da validade do produto musical gerado. Para tal, este capítulo foi estruturado em cinco seções. A Seção 5.1 apresenta a descrição dos parâmetros que foram utilizados para realização dos testes. Na Seção 5.2 é apresentada a metodologia que será utilizada para aferição, apresentação e avaliação dos resultados obtidos. Na Seção 5.3 são detalhados e avaliados os resultados obtidos na síntese da arquitetura. A Seção 5.4 apresenta alguns dos produtos sonoros gerados, pontuando e comparando as peculiaridades de cada um destes produtos. A Seção 5.5 apresenta o comparativo entre os resultados obtidos nesta dissertação e nos trabalhos relacionados.

5.1 Descrição dos Parâmetros Utilizados

Conforme descrito ao longo dos capítulos anteriores, a geração de produtos musicais através da utilização de autômatos celulares depende de alguns parâmetros de configuração da arquitetura proposta. Nesta dissertação, um produto musical é considerado válido quando, após o processamento realizado pela arquitetura obtêm-se uma melodia que possa ser classificada como música aleatória/experimental. Para tal, os testes realizados tem o objetivo de validar algumas características da música obtida, que são a *distribuição tonal*, o *formato de transposição entre as notas*, o *contorno melódico* e a *chave sonora*, esta última especificamente apresentada por trecho da melodia.

No Capítulo 4, foram definidas as opções de configuração presentes na arquitetura, e, para obtenção das características citadas anteriormente, seis destas opções de configuração, *configuração inicial das células*, *conjunto de regras*, *gama de notas disponíveis*, *valor do metrô-*

nomo, *volume e duração das notas* e *armadura de clave*, serão exploradas na etapa de testes. Cada uma destas opções e as motivações para tais escolhas serão comentadas ao longo desta Seção.

5.1.1 Definição das Configurações Iniciais

Conforme descrito ao longo do Capítulo 1, a configuração inicial de cada uma das células de um CA, q_0, interfere diretamente no resultado da evolução deste. A consequência deste fato, para o trabalho proposto, é a variação do produto musical obtido. A escolha de diferentes configurações iniciais para a etapa de testes será utilizada para espelhar o potencial de interferência da configuração inicial no produto musical obtido. Para tal, foram escolhidos dois modelos de configuração inicial para cada um dos CAs utilizados integrantes da arquitetura.

Para os CAs unidimensionais foram escolhidas a configuração inicial proposta por *Wolfram* (WOLFRAM, 2006), CW1, e uma configuração inicial pseudo aleatória, CW2, na qual foram ativadas as células cujo índice é um número primo, como por exemplo, a célula C_2 e a célula C_7. A escolha da configuração proposta por *Wolfram* foi em função do connhecimento prévio do comportamento do CA no decorrer de sua evolução.

Também foram definidos dois tipos de configuração inicial para os CAs bidimensionais. A primeira configuração inicial, CB1, foi gerada através do arranjo aleatório de *gliders*, *frogs* e *blinkers*. *Gliders*, *frogs* e *blinkers* são padrões gráficos que comumente surgem no decorrer da evolução do CA *Game of Life* (CONWAY, 1970). A segunda configuração inicial, CB2, foi gerada através da utilização apenas de *gliders* espalhados ao longo de CA.

Já para os CAs tridimensionais de *Neuman*, as duas configurações iniciais foram criadas com base na utilização de *gliders*, *frogs* e *blinkers* em ambiente 3D. A primeira configuração, CT1, utiliza *gliders* e *frogs* distribuídos longo do CA, já a segunda configuração, CT2, utiliza *frogs* e *blinkers* distribuídos pelo CA.

5.1.2 Definição dos Conjuntos de Regras

Conforme descrito no decorrer do Capítulo 4, os módulos CA utilizados neste trabalho possuem raio de vizinhança igual a 1. Portanto para a etapa de testes foram escolhidos conjuntos de regras com esta característica, publicados em artigos diversos (CONWAY, 1970; WOLFRAM, 2006; MARTINS; FYNN; OLIVEIRA, 2011). A escolha dos conjuntos de regras que serão utilizados baseou-se no fato de que, nos testes preliminares realizados durante a confecção desta dissertação padrões gráficos densos ou imutáveis gerados por determinados conjuntos

de regras, como por exemplo, as regras 1 e 7 de *Wolfram*, demonstraram-se auditivamente cansativos ao longo de sua execução, pois padrões gráficos densos geram produtos musicais extremamente longos e padrões gráficos imutáveis geram um produto musical extremamente repetitivo.

Para os CAs unidimensionais foram escolhidas a regra 026 e a regra 124 dentre os 256 conjuntos propostos por *Wolfram*. Estas regras, que nesta etapa serão definidas como RU1 e RU2, possuem como característica a formação de padrões gráficos distintos (WOLFRAM, 2006). Para os CAs bidimensionais de *Neuman* foram escolhidos os 2 conjuntos de regras apresentados como ótimos em pesquisas acerca de classificação de densidade (MARTINS; FYNN; OLIVEIRA, 2011). O conjunto ótimo para *grids* pares, denominado RN1, e o conjunto ótimo para *grids* impares, denominado RN2. Para os CAs bidimensionais de *Moore*, foi escolhido o melhor conjunto de regras para *grids* pares, obtido através de pesquisas acerca de classificação de densidade (MARTINS; FYNN; OLIVEIRA, 2011), denominado RM1 e o conjunto de regras de *Game of Life* (CONWAY, 1970), denominado RM2. Para os CAs tridimensionais de *Neuman* foi utilizado o conjunto de regras do *Game of Life*, denominado TN1 e o conjunto de regras com base em números primos, denominado TN2.

5.1.3 Demais Configurações

Conforme explicitado ao longo do Capítulo 4, além da determinação da configuração inicial e do conjunto de regras utilizado por cada CA, também é necessária a definição da *gama de notas disponíveis*, da *gama de instrumentos disponíveis*, do *metrônomo* e da *armadura de clave*, que são responsáveis pela parcial rítmica da complexidade audível do produto sonoro. Serão utilizadas todas as notas disponíveis no padrão MIDI. Com relação aos instrumentos serão utilizados apenas os instrumentos de corda disponíveis no padrão MIDI. A armadura de clave foi mantida em *C*, e para configuração do componente `MET` escolheu-se a opção 00. Para cada CA foram definidas 10 iterações, de forma a permitir a melhor visualização dos resultados.

5.2 Metodologia da Avaliação

Conforme explicitado no decorrer do Capítulo 4, o produto resultante da arquitetura implementada são informações musicais. Estas informações, que são registradas conforme a codificação MIDI-1, podem ser gravadas com a extensão *.mid*, através de um editor hexadecimal, para posteriormente serem executadas em qualquer leitor MIDI. De arquivos *.mid* é possível realizar

a extração de algumas características que permitirão a realização de uma análise de qualidade sonora.

De forma a comprovar a validade do produto sonoro gerado, o processo de análise de resultados foi realizado com o objetivo de evidenciar a qualidade e os detalhes do produto musical resultante do processamento efetuado pela arquitetura. A estratégia utilizada para realização dos testes foi elaborada a partir do uso de técnicas para extração de informação musical conhecidas como *Musical Information Retrieval - MIR* (LOUREIRO, 2006). Para tal foi escolhida a biblioteca MatLab MIDI ToolBox (EEROLA; TOIVIAINEN, 2004; SMIT, 2011), que contém funções compatíveis com o *MatLab*, permitindo a extração direta de dados que viabilizam a análise da informação musical contida em um arquivo *.mid*. As próximas seções apresentam as funções que foram utilizadas para análise da qualidade sonora.

5.2.1 Distribuição Tonal

A primeira função utilizada para apresentação de resultados foi a de amostragem do percentual de distribuição tonal. Esta função permite a identificação da proporção de distribuição de notas ao longo de uma melodia. O gráfico resultante apresenta no eixo das ordenadas o valor percentual da ocorrência de um determinado grupo de notas ao longo da melodia e no eixo das abscissas os grupos de notas contidos no arquivo. Grupo de notas é a denominação para a ocorrência de uma determinada nota ao longo da melodia, independente de qual *oitava* ela pertença. Um exemplo de distribuição tonal é apresentado na Figura 49.

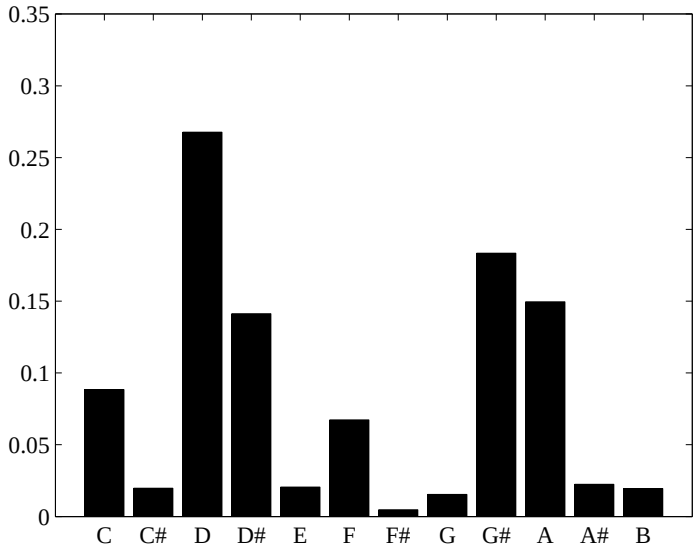


Figura 49: Histograma normalizado de distribuição tonal da música *The Beautiful People*

Neste exemplo, no qual é apresentada a distribuição tonal de uma música conhecida, *The Beautiful People* (MANSON, 1997), é possível evidenciar que o grupo de notas *D* foi o mais utilizado e o grupo de notas *F#* o menos utilizado.

5.2.2 Distribuição de Intervalos

A segunda função utilizada permite que seja realizada a identificação da distribuição de *intervalos musicais* durante a melodia. O gráfico resultante desta função apresenta no eixo das abscissas os *intervalos musicais* que ocorreram no decorrer da melodia, como por exemplo, *oitava perfeita* e *terça menor*. O eixo das ordenadas desta função indica a distribuição percentual de ocorrência dos *intervalos musicais* na melodia. A Figura 50 apresenta a proporção de distribuição de *intervalos* em um dos trechos da música *The Beautiful People* (MANSON, 1997). Nesta distribuição é possível evidenciar que no trecho em questão há maior ocorrência do intervalo denominado *quinta justa*, *P5*, e nenhuma ocorrência do *intervalo* denominado *oitava perfeita*, *P8*.

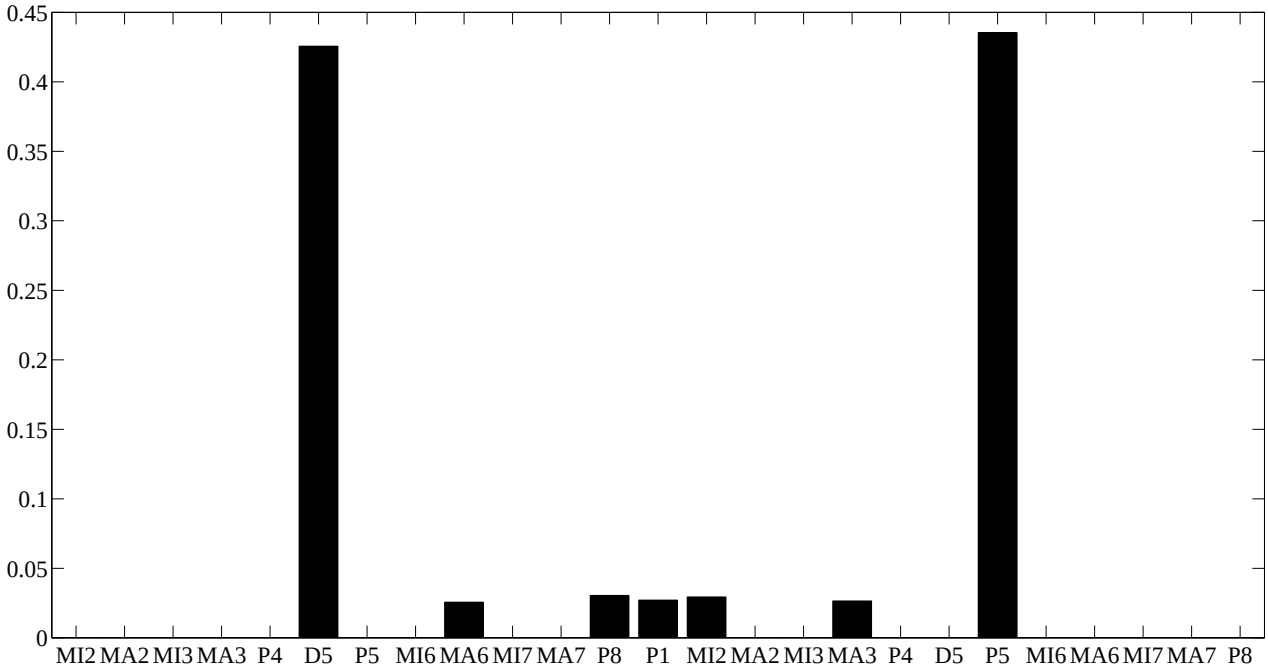


Figura 50: Histograma de *intervalos* em um trecho da música *The Beautiful People*

5.2.3 Transposição Tonal

A terceira função utilizada, de forma a complementar a informação acerca da distribuição tonal, foi a de proporção de transposição entre grupos de notas. Este função apresenta a proporcio-

nalidade de ocorrências de transposição entre dois grupos de notas. O gráfico resultante desta função apresenta no eixo das ordenadas o primeiro grupo executado, e no eixo das abscissas o segundo grupo executado. Quanto mais vezes ocorrer a transposição entre um grupo e outro, mais escura será marcada a interseção entre estes grupos. A Figura 51 apresenta a proporção de variação entre grupos de notas no decorrer de um dos trechos da música *The Beautiful People* (MANSON, 1997). Neste gráfico é possível notar que a maior incidência de transposição ocorre entre as notas do grupo D para as notas do grupo A. Também é possível evidenciar a menor incidência de transposição, que ocorre das notas do grupo B para as notas do grupo D.

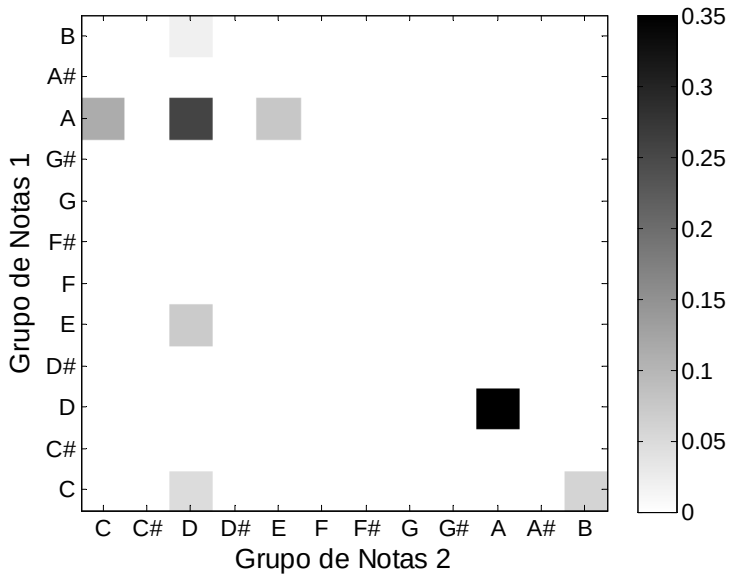


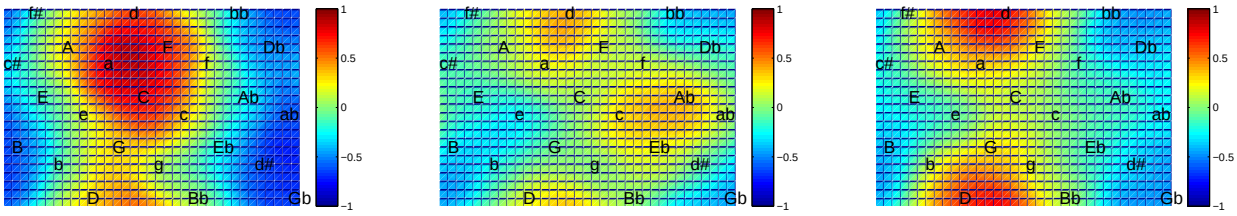
Figura 51: Histograma de tranposição de notas em um trecho da música *The Beautiful People*

5.2.4 Força Tonal

A quarta função utilizada foi a de força tonal por tempo musical (FARIA, 1991; COSTA; FINE; BITTI, 2004; TOIVIAINEN, 2005; PASCOAL; PASCOAL, 2012). Nesta função, é possível visualizar a movimentação do *centro tonal* ao longo da música, auxiliando na compreensão da variação das notas ao longo do tempo. *Centro tonal* é a nota dois tons e meio acima da tônica do acorde dominante, servindo, em música tonal, como referência para a *harmonia* e consequentemente para a compreensão musical do ouvinte (FARIA, 1991; DUDEQUE, 2008; SOLIMANDO, 2013).

O gráfico desta função é um toroide aberto, no qual os acordes menores são representados por letras minúsculas e os acordes maiores são representados por letras maiúsculas, como por exemplo, *Am* é representado no gráfico pela letra *a*. A proximidade do *centro tonal* é

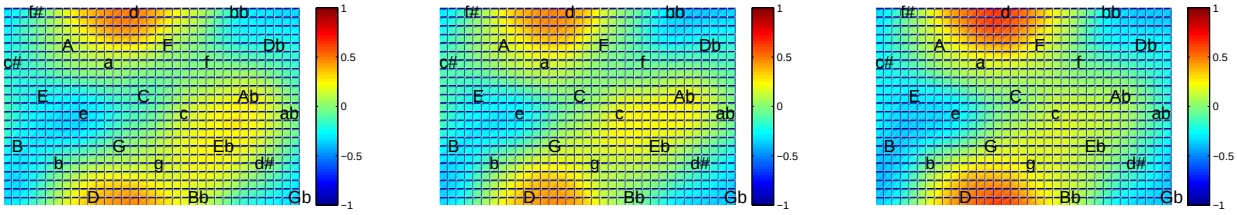
marcada pela cor vermelha, enquanto a distância do *centro tonal* é denotada pela cor azul. As Figuras 52 e 53 apresentam a distribuição tonal do segundo e do sexto segmentos da música *The Beautiful People* (MANSON, 1997), a cada 40 batidas. Na Figura 52, que apresenta o segundo trecho da música analisada, é possível evidenciar que o *centro tonal* começa próximo a nota *Am* e termina entre as notas *D* entre *Dm*.



(a) Batida 40 (b) Batida 80 (c) Batida 120

Figura 52: Variações de *centro tonal* no segundo segmento da música *The Beautiful People*

Já na Figura 53, que apresenta o sexto segmento da música The Beautiful People, é possível evidenciar que o *centro tonal* se estabilizou entre as notas *D* e *Dm*.



(a) Batida 240 (b) Batida 280 (c) Batida 320

Figura 53: Variações de *centro tonal* no sexto trecho da música *The Beautiful People*

5.2.5 Contorno Melódico

A última função utilizada foi a de contorno melódico (ANDRADE, 2004). Nesta função, é possível visualizar a evolução da melodia através da indicação das notas que ocorrem a cada batida da música, permitindo que seja evidenciado com exatidão comportamento musical ao longo do tempo, desde existam dados suficientes para esta análise. O gráfico resultante desta função apresenta no eixo das abscissas o *tempo musical*, em batidas, e no eixo das ordenadas apresenta as notas, de acordo com a codificação MIDI, como por exemplo, o número 44, que representa a nota *G#*. A Figura 54 apresenta o contorno melódico da introdução da música *The Beautiful People*. Nesta distribuição é possível evidenciar que a nota *E*, no padrão MIDI nota

40, é executada repetidas vezes em *intervalos* regulares, caracterizando-se como base rítmica desta parte da melodia. Também é possível evidenciar que as notas A e C, respectivamente notas 33 e 36 no padrão MIDI, aparecem ritmicamente durante este trecho da melodia.

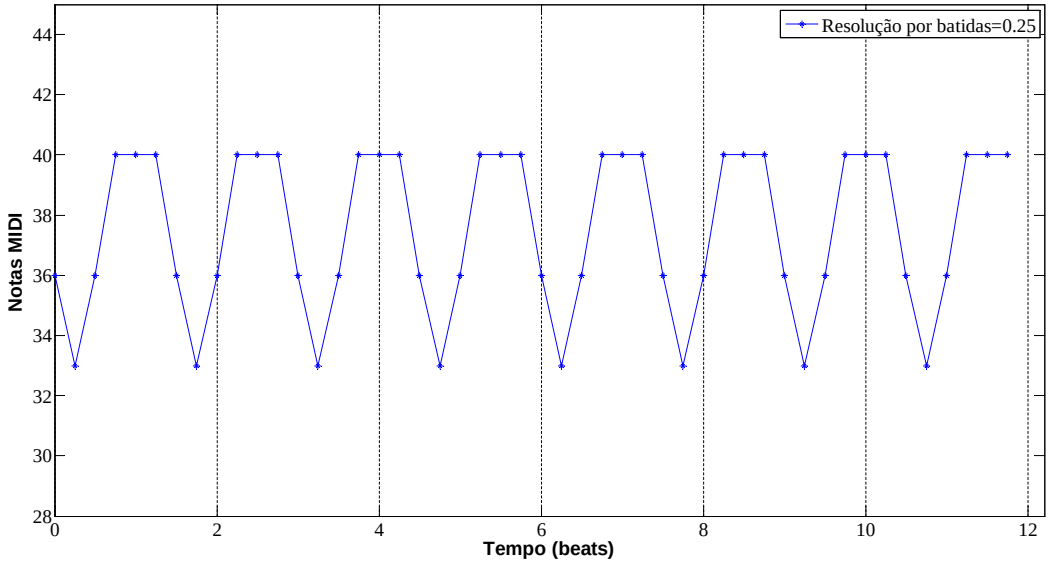


Figura 54: Contorno melódico de um dos trechos da música *The Beautiful People*

5.3 Resultados de Síntese

O *hardware* proposto nesta dissertação foi especificado utilizando a linguagem de descrição de *hardware* VHDL (ASHENDEN, 1990; NAVABI, 1997). Para obtenção dos resultados de síntese foi utilizada a ferramenta *Project Navigator*, que pertence ambiente integrado de desenvolvimento *Xilinx Design Suite* (XILINX, 2010, 2013b). Esta etapa permitiu a obtenção das demandas de área necessárias a implementação em FPGA da arquitetura. Para tal considerou-se, para fins de comparação, a placa Virtex-5 XC5VLX110T, (XILINX, 2013a; UG349, 2010), que contém um total de 69.120 *LookUp-Tables - LUTs*.

Os resultados de síntese da arquitetura proposta foram efetuados a partir da análise das combinações possíveis para escolha dos CAs existentes na arquitetura. Conforme explicitado no Capítulo 4, durante a etapa de configuração da arquitetura, o usuário escolhe dois CAs para efetuar a composição musical, um para geração dos dados referentes a notação instrumental e outro para geração dos dados referentes a notação musical. Durante esta etapa de configuração o usuário define, dentre outros itens, três tipos de dados que interferem diretamente no tamanho, em *bits*, do arquivo que deverá ser convertido para a extensão *.mid* e consequente-

mente no tamanho dos componentes `AZM` e `AGT`. Estes dados, que são o conjunto de regras, os estados iniciais das células e número de iterações a serem realizadas pelos CAs impossibilitam a estimativa de consumo de área para os componentes `AZM` e `AGT`.

A rotina de testes realizada para avaliação dos valores de consumo de área e de *clock* necessários a implementação da arquitetura proposta é apresentada na Tabela 31. Esta rotina foi estruturada em função do fato de que independentemente dos módulos escolhidos para geração musical, os componentes projetados, com exceção dos componentes `AZM` e `AGT`, não têm seus tamanhos alterados.

Tabela 31: Rotina de testes de síntese realizados

Teste	Primeiro autômato celular	Segundo autômato celular
TS1	UW	UW
TS2	UW	BN
TS3	UW	BM
TS4	UW	TN
TS5	BN	BN
TS6	BN	BM
TS7	BN	TN
TS8	BM	BM
TS9	BM	TN
TS10	TN	TN

O detalhamento dos resultados de síntese, que indicam a demanda de área necessária a implementação do projeto em FPGA, são apresentados na Tabela 32. Os valores obtidos nesta etapa confirmam a perspectiva de que os CAs com maior número de proposições lógicas no conjunto de regras demandariam o maior consumo de área para implementação da arquitetura, em LUTs e Flip Flop Pairs.

Tabela 32: Resultados obtidos na etapa de síntese

Teste	**LUTs**	**Slice Registers**	**Flip Flop Pairs**
TS1	3066	1121	3259
TS2	11338	1121	11385
TS3	131848	1121	132036
TS4	32454	1110	32501
TS5	19080	1118	19263
TS6	139779	1121	139962
TS7	40037	1107	40080
TS8	263368	1121	263551
TS9	161111	1110	161293
TS10	61554	1096	61597

A comparação entre os resultados obtidos na etapa de síntese é apresentada na Figura 32. Através deste comparativo é possível evidenciar que nas implementações efetuadas a demanda, em cada teste, por LUTs e Flip Flop Pairs foi similar. Já a demanda de Slice Registers foi praticamente a mesma em todos os testes. Neste comparativo também é possível evidenciar que 40% das combinações propostas não poderiam ser implementadas na placa de referência em função área necessária, como por exemplo, o teste TS8 que demanda mais de 200% da área disponível na placa, utilizando os dois CAs com vizinhança bidimensional de *Moore*. Com relação a frequência de *clock* necessária para implementação, em todos os testes realizados o valor evidenciado foi o de 4.494 *MHz*.

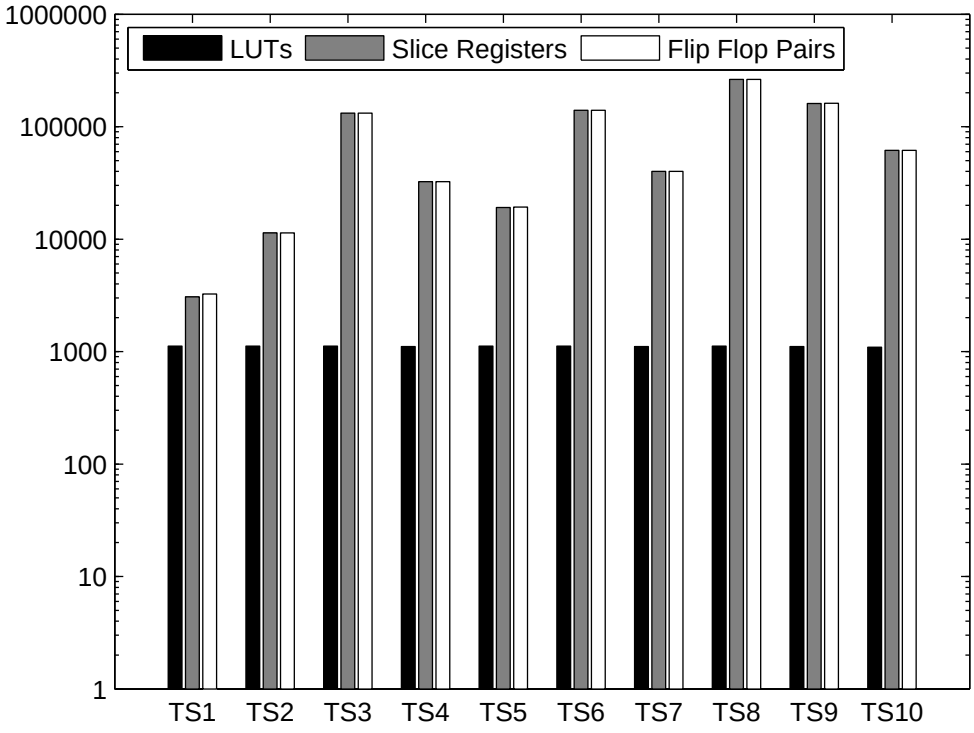


Figura 55: Comparativo dos resultados de consumo de área

5.4 Análise dos Produtos Musicais Obtidos

A apresentação dos resultados foi dividida por função, conforme explicitado no decorrer da Seção 5.2. A decisão de efetuar apresentação dos resultados por função objetivou fornecer uma visão global acerca dos resultados obtidos. Para tal, os testes realizados foram estruturados de forma a permitir a observação da diferença nos resultados que é gerada pela variação do modelo de CA utilizado, para geração de notas e instrumentos e dos parâmetros de configuração. As

tabelas 33 e 34 apresentam a estrutura de testes utilizada para coleta dos dados necessários a avaliação dos resultados.

A Tabela 33 apresenta a estruturação dos testes realizados, através da indicação do módulo CA utilizado para geração de notas musicais. A primeira coluna da Tabela 33 indica o intervalo de testes, a segunda coluna indica o CA utilizado, a terceira coluna indica a regra utilizada, e a quarta coluna indica a configuração inicial utilizada.

Tabela 33: Descrição dos parâmetros utilizados para geração de notação musical

Teste	Módulo CA	Regra	CI
T01 – T04	UWN	RU1	CW1
T05 – T08	BNN	RN1	CB1
T06 – T12	BMN	RM1	CB1
T13 – T16	TNN	RT1	CT1
T17 – T20	UWN	RN2	CW2
T21 – T24	BNN	RN2	CB2
T25 – T28	BMN	RM2	CB2
T28 – T32	TNN	RN2	CT2

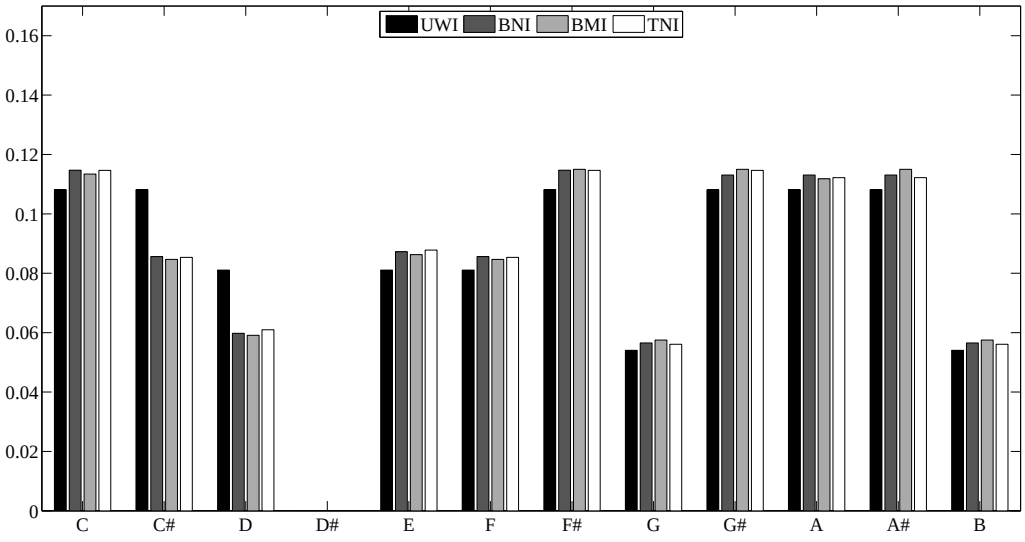
A Tabela 34 apresenta a rotina de testes, organizada por CA para geração de instrumentos. A primeira coluna da Tabela 34 indica o nome dos testes realizados, a segunda coluna indica o CA utilizado geração de instrumentos, a terceira coluna indica a regra utilizada, e a quarta coluna indica a configuração inicial utilizada. A junção das Tabelas 33 e 34 fornece as especificações de todos os 32 testes realizados para análise dos produtos musicais.

Tabela 34: Descrição dos parâmetros utilizados para geração de notação instrumental

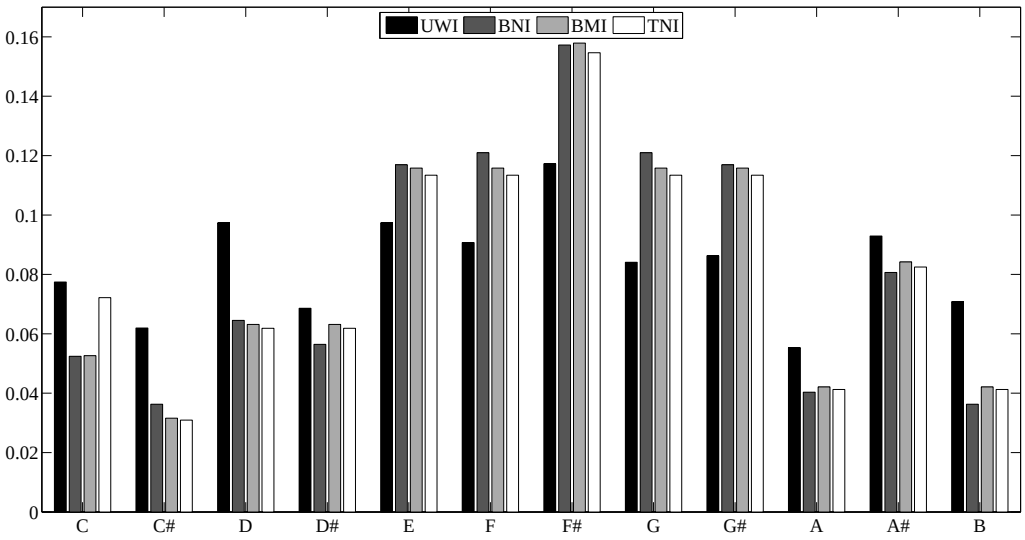
Teste	Módulo CA	Regra	CI
T01, T05, T09, T13	UWI	RU1	CW1
T02, T06, T10, T14	BNI	RN1	CB1
T03, T07, T11, T15	BMI	RM1	CB1
T04, T08, T12, T16	TNI	RT1	CT1
T17, T21, T25, T29	UNI	RU2	CB2
T18, T22, T26, T30	BNI	RN2	CB2
T19, T23, T27, T31	BMI	RM2	CB2
T20, T24, T28, T32	TNI	RN2	CB2

5.4.1 Resultados de Distribuição Tonal

A Figura 56 apresenta os comparativos acerca da distribuição percentual tonal obtida com a utilização do componente `UWN`. Para tal, são apresentados os resultados dos testes T01 a T04 e T17 a T20. O primeiro comparativo que se refere aos testes T01 a T04, é apresentado na Figura 56(a).



(a) RU1 e CW1

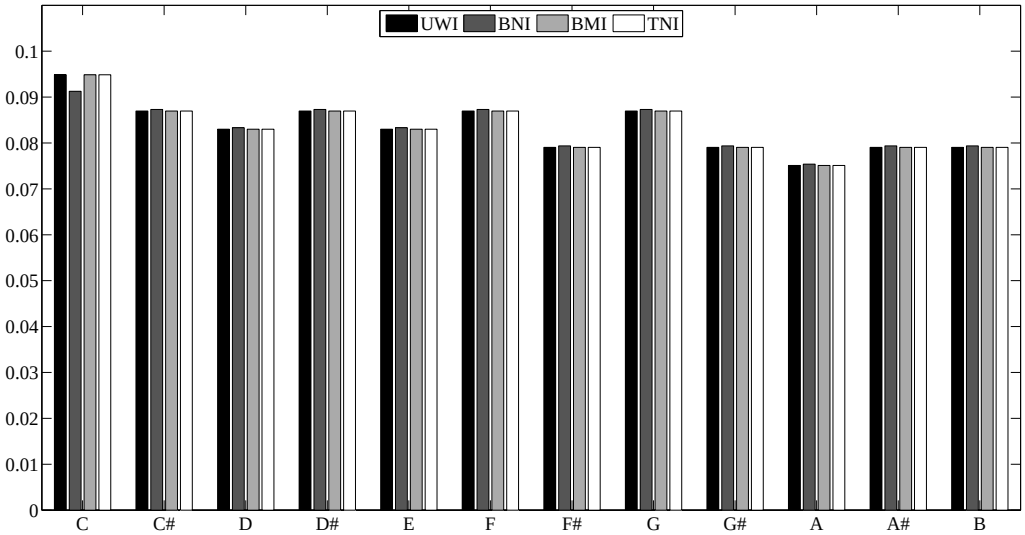


(b) RU2 e CW2

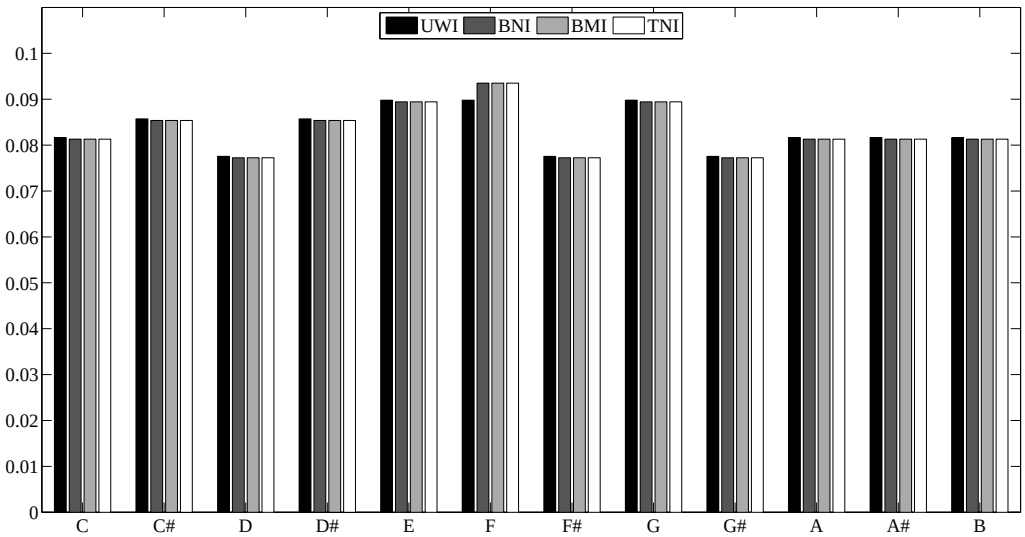
Figura 56: Comparativo das distribuições de nota obtidas com o componente `UWN`

Neste comparativo é possível evidenciar que todas as combinações utilizadas conseguiram alcançar a variedade tonal esperada, gerando melodias estruturalmente equilibradas. Contudo, nenhuma das combinações apresentadas neste comparativo gerou a nota D#. É possível evidenciar também, que, em todos os testes apresentados as tonalidades *C*, *F#*, *G#*, *A* e *A#* foram as mais executadas. Nos testes T02, T03 e T04 o valor do acorde *F#* foi idêntico. A maior variação percentual de distribuição tonal ocorreu no teste T01 para as tonalidades *C#* e *D*. O segundo comparativo refere aos testes de T17 a T20. Neste comparativo, que é apresentada na Figura 56(b), é possível evidenciar que os testes analisados alcançaram toda a variedade tonal possível. Os únicos testes que apresentaram percentuais idênticos foram os testes T18 e T19, para as notas *C* e *C#*. De forma idêntica a distribuição apresentada na

Figura 56(a), a nota F# obteve maior valor percentual. Contudo, comparando as Figuras 56(a) e 56(b), é possível evidenciar que o testes de T17 a T20 apresentaram maior equilíbrio na distribuição dos valores percentuais. Os resultados percentuais de distribuição tonal dos testes T05 a T08 e T21 a T24, nos quais o componente `BNN` foi utilizado para geração da notação musical, são apresentados na Figura 57.



(a) RN1 e CB1

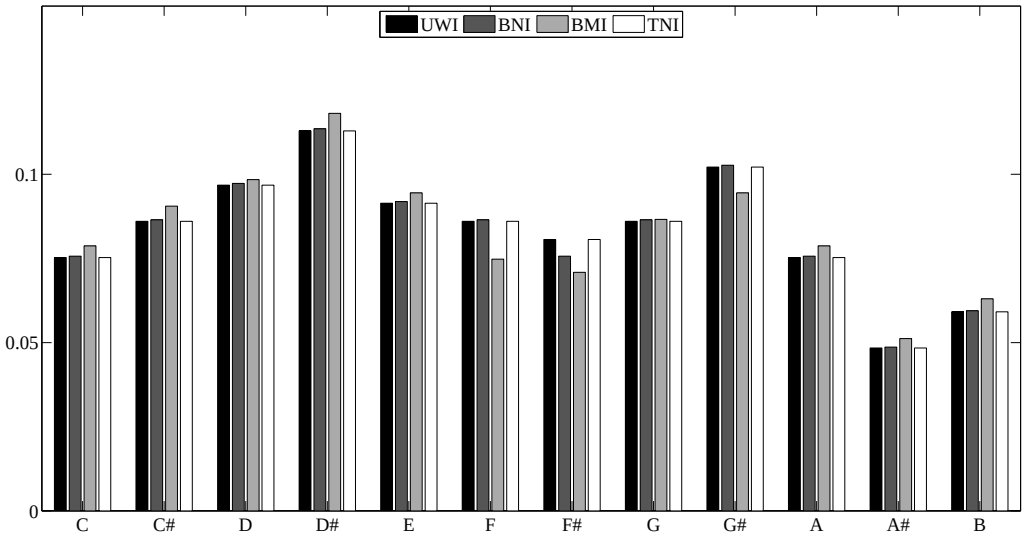


(b) RN2 e CB2

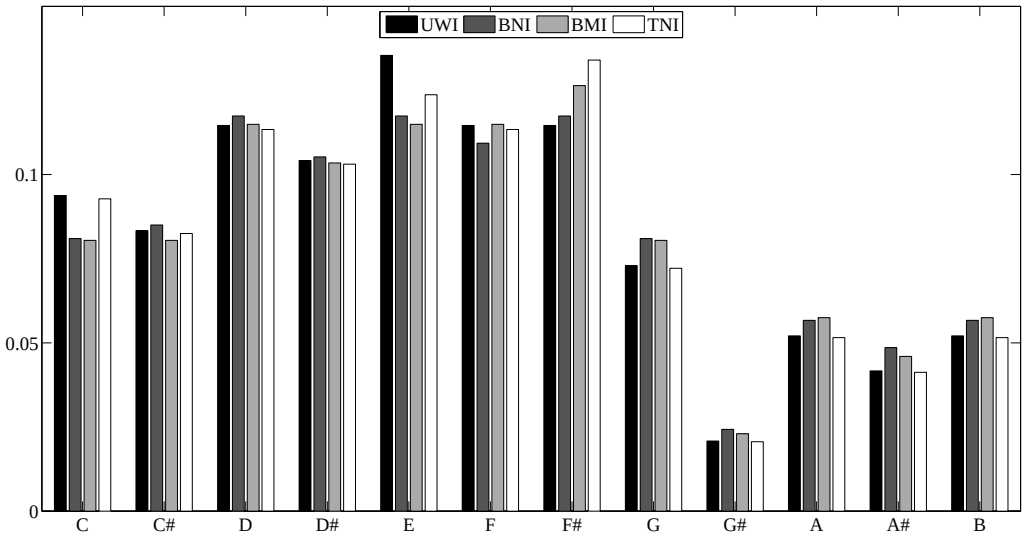
Figura 57: Comparativo das distribuições de nota obtidas com o componente `BNN`

O primeiro comparativo, referente aos testes T05 a T08, é apresentado na Figura 57(a). Neste comparativo é possível evidenciar que os testes realizados forneceram toda a gama de notas musicais existente no padrão MIDI. Em relação aos percentuais alcançados, com exceção da nota *C*, os valores foram equivalentes em todos os testes. Os testes T06 e T07, em função

dos conjuntos de regras e as configurações iniciais, produziram resultados percentuais similares. O comparativo realizado entre os testes T21 a T24, é apresentado na Figura 57(b). De forma equivalente ao primeiro comparativo, os testes apresentados na Figura 57(b) também alcançaram toda a gama de notas disponíveis no padrão MIDI. Os percentuais de distribuição se mantiveram estáveis para quase todos os grupos de notas, com exceção do grupo *F*, que obteve um valor percentual menor no teste T05. A Figura 58 apresenta os comparativos dos testes T09 a T12 e T25 a T28, que utilizaram o componente `BMN` para geração dos dados de notação musical.



(a) RM1 e CB1

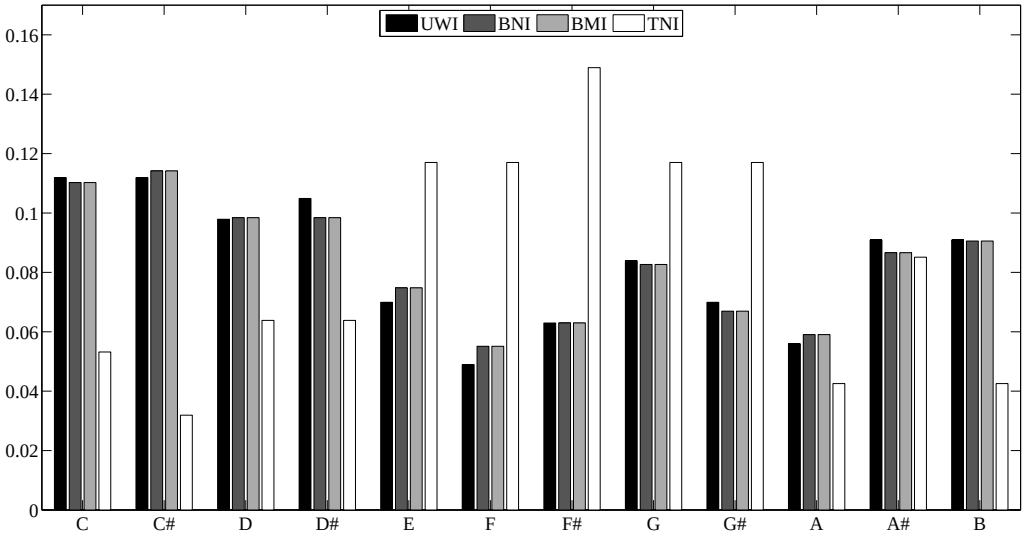


(b) RM2 e CB2

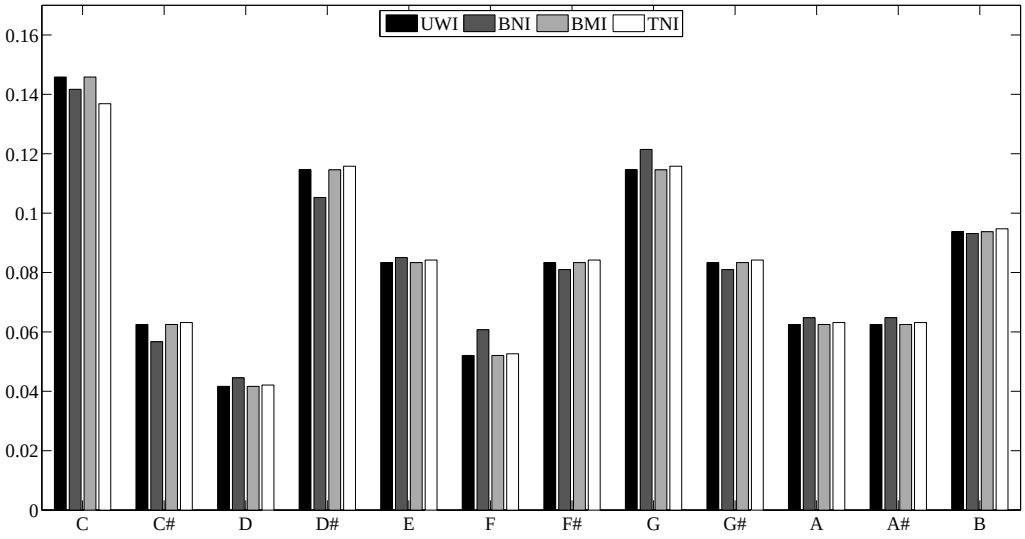
Figura 58: Comparativo das distribuições de nota obtidas com o componente `BMN`

O primeiro comparativo que se refere aos testes T09 a T12, são apresentadas na Figura

58(a). Nesta distribuição, que utilizou a mesma configuração inicial dos testes apresentados na Figura 58(a), é possível evidenciar a influência que os conjuntos de regra e de *vizinhança* exercem sobre a evolução de um CA. Já a Figura 58(b) apresenta o quadro comparativo referente aos testes T25 a T28. Para ambos os casos, os únicos percentuais de distribuição que se mostraram equivalentes foram os referentes as notas $A\#$ e B. Os comparativos referentes aos testes T13 a T16 e T29 a T32 são apresentados na Figura 59.



(a) RT1 e CT1



(b) RT2 e CT2

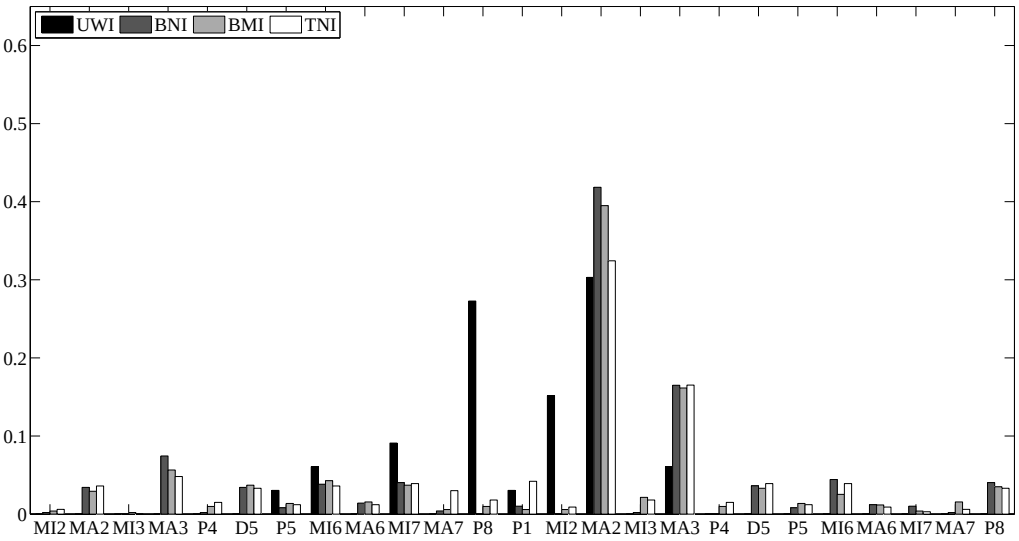
Figura 59: Comparativo das distribuições de nota obtidas com o componente `TNN`

O primeiro comparativo, que se refere aos testes T13 a T16, é apresentado na Figura 59(a). De todas as distribuições neste comparativo, o teste T13 teve a maior variação percentual, como maior valor percentual o obtido para a nota $F\#$. Os demais testes, principalmente

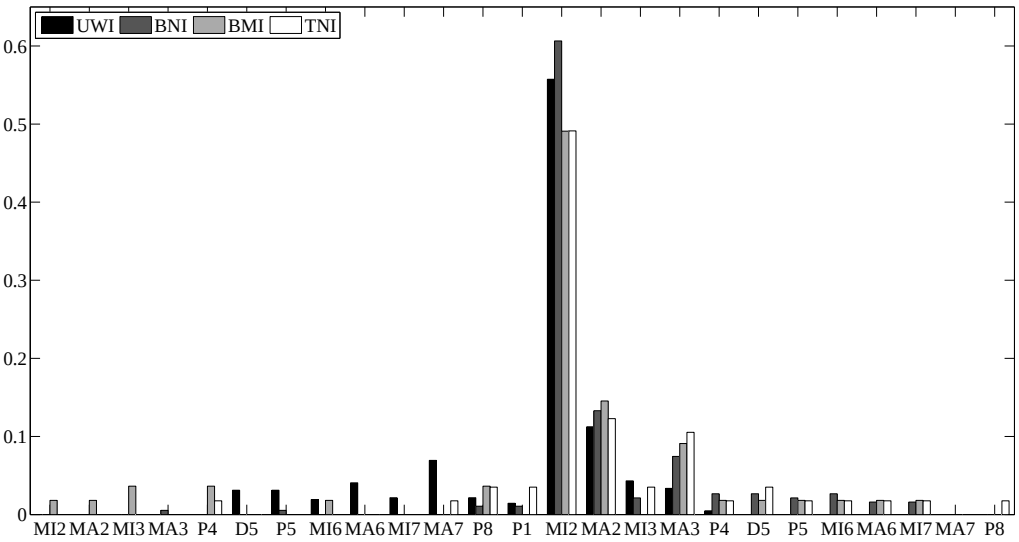
o T14 e o T15, apresentaram percentuais equivalentes de distribuição. Na Figura 59(b) é apresentado o comparativo referente aos testes T29 a T32. Ao contrário do comparativo anterior, Figura 59(a), com exceção das notas do grupo *C*, os valores percentuais obtidos nos testes T29 a T32 são equivalentes. Os resultados apresentados indicam que as melodias, obtidas através do processamento realizado pela arquitetura, produziram melodias com uma boa distribuição de valores percentuais.

5.4.2 Resultados de Distribuição de Intervalos

Os quadros comparativos dos valores percentuais de distribuição de intervalo obtidas nos testes T01 a T04 e T17 a T20 são apresentados na Figura 60.



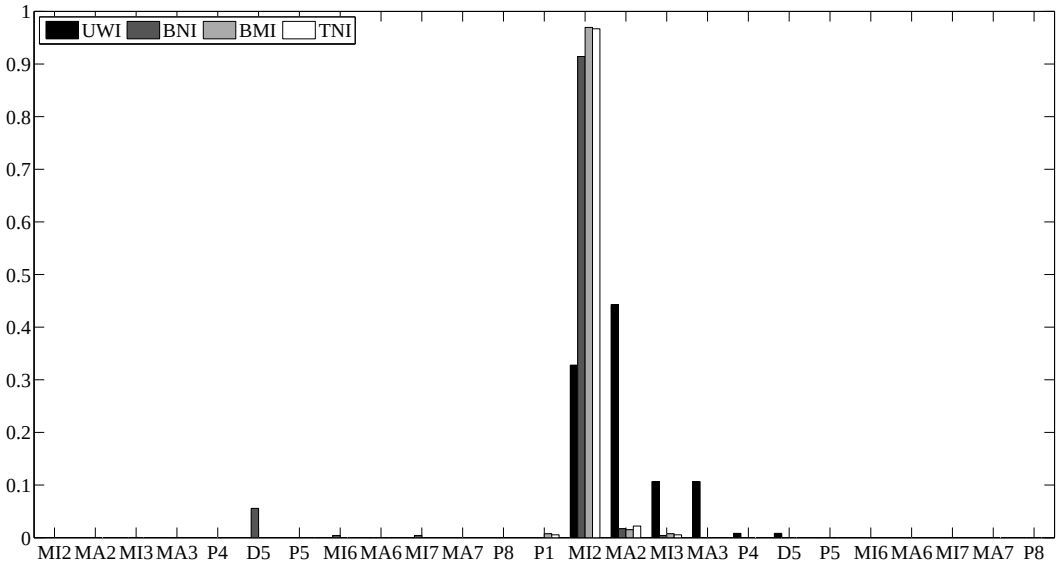
(a) RW1 e CW1



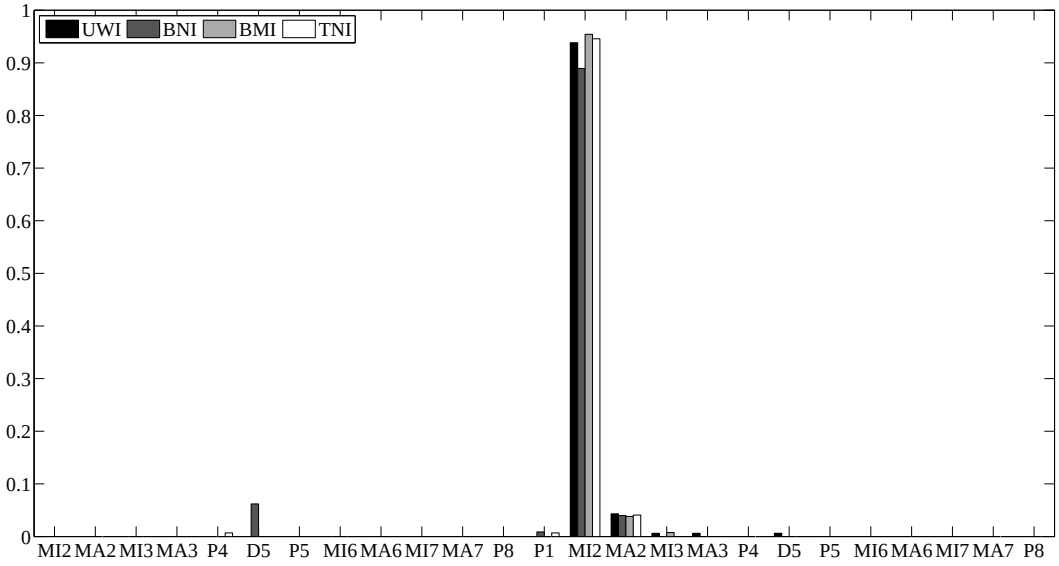
(b) RW2 e CW2

Figura 60: Comparativo das distribuições de intervalo obtidas com o componente `UWN`

O primeiro comparativo, Figura 60(a), apresenta os percentuais de distribuição obtidos nos testes T01 a T05. Neste comparativo é possível evidenciar que estes testes alcançaram um número variado de intervalos, sendo o maior valor percentual registrado para o intervalo *segunda maior*. O segundo comparativo, Figura 60(b), apresenta os valores percentuais para testes T17 a T20, que demonstram que, ao contrário do comparativo anterior, Figura 60(a), os valores percentuais dos testes T17 a T20 tiveram maior concentração em apenas um intervalo. A Figura 61 apresenta os resultados de distribuição de intervalos obtidos nos testes T05 a T08 e T21 a T24.



(a) RN1 e CB1

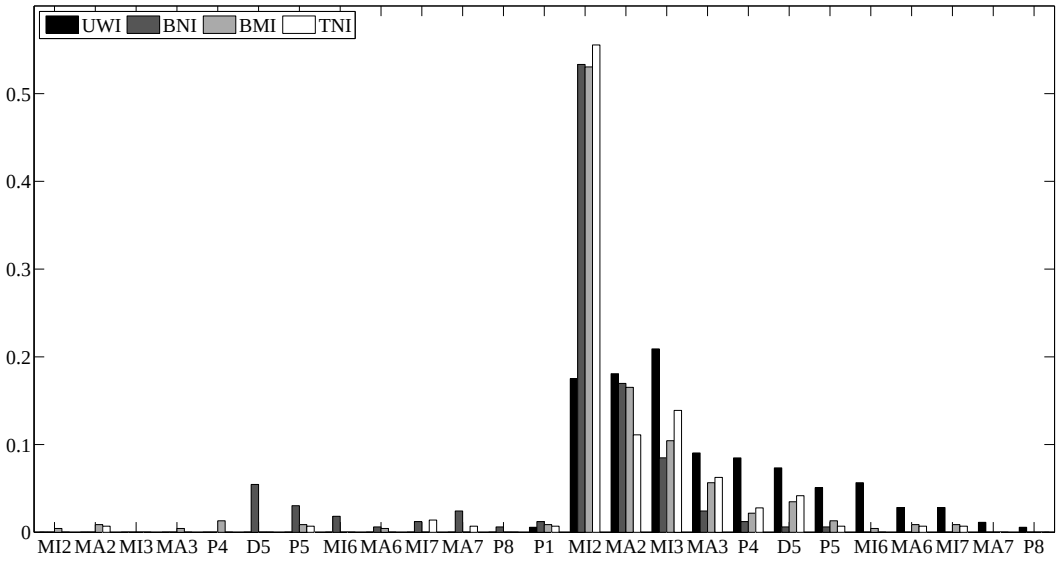


(b) RN2 e CB2

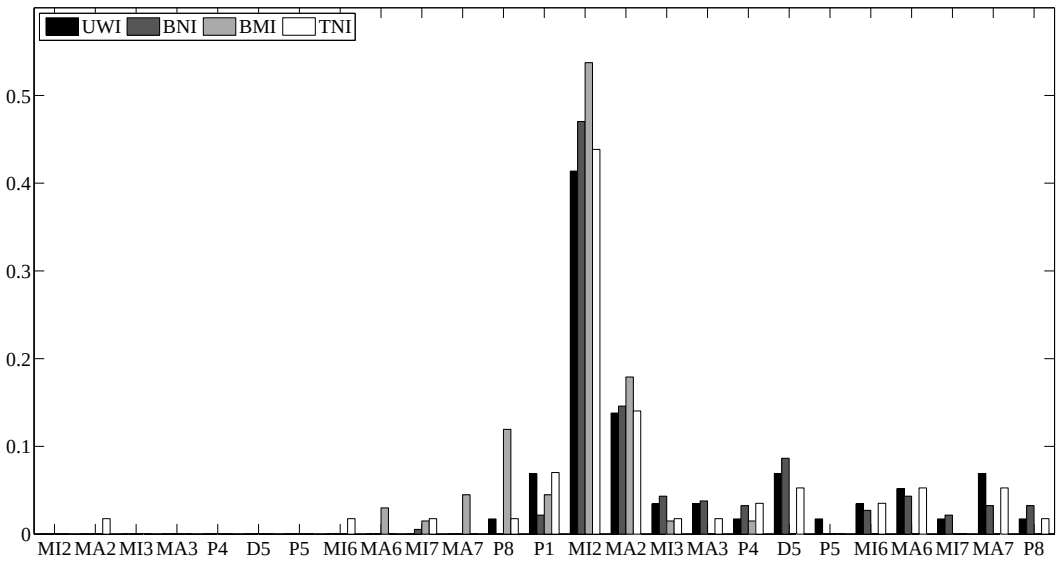
Figura 61: Comparativo das distribuições de intervalo obtidas com o componente `BNN`

Na Figura 61(a) é apresentado o comparativo dos resultados obtidos nos testes T05 a

T08. O principal destaque desta distribuição é o fato de que nos testes T06 e T07 a melodia foi confeccionada quase que em totalidade utilizando apenas um tipo de intervalo. A Figura 61(b) apresenta o comparativo das distribuições de intervalos referentes aos testes T21 a T24. De forma simular ao resultado apresentado na Figura 61(a) a distribuição de intervalos concentrou-se também em apenas um intervalo. A Figura 62 apresenta os resultados percentuais da distribuição de intervalos nos testes T09 a T12 e T25 a T28.



(a) RM1 e CB1

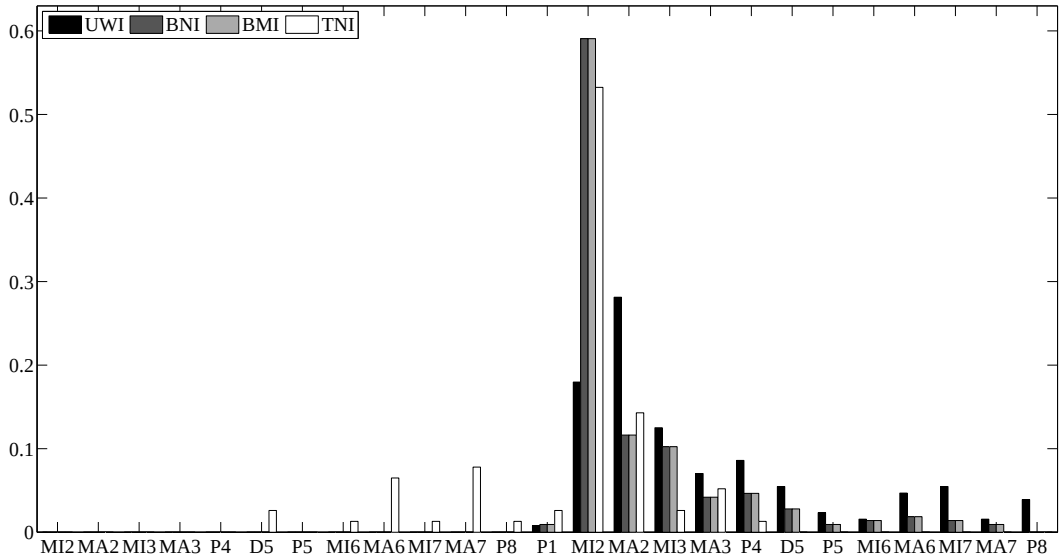


(b) RM2 e CB2

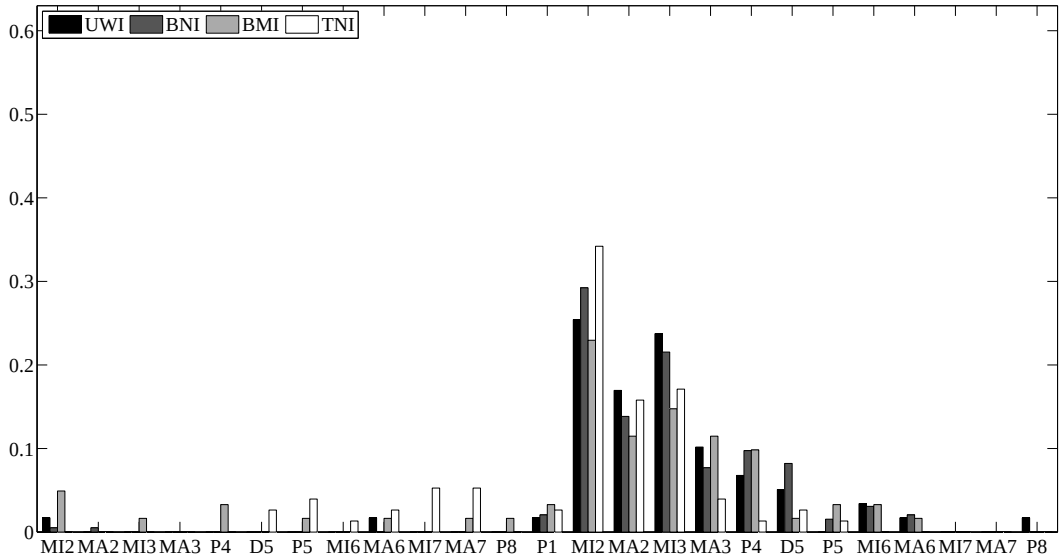
Figura 62: Comparativo das distribuições de intervalo obtidas com o componente `BMN`

A Figura 62(a) apresenta o comparativo de distribuição de intervalos entre os testes T09 a T12. Neste, os testes T10 e T11 geraram melodias que se distribuíram ao longo dos intervalos maiores, apresentando poucos índices de *oitavas perfeitas* e *quartas perfeitas*. A Figura 62(b),

que apresenta as distribuições de intervalo para os testes T25 a T28, demonstra que estes testes geraram uma distribuição de intervalos similar a apresentada nos testes T09 a T12, apesar de ter ocorrido uma alta concentração percentual, em ambos os casos, em apenas um intervalo. Na Figura 63 são apresentados os resultados de distribuição referentes aos testes T13 a T16 e T29 a T32.



(a) RT1 e CT1



(b) RT2 e CT2

Figura 63: Comparativo das distribuições de intervalo obtidas com o componente `TNN`

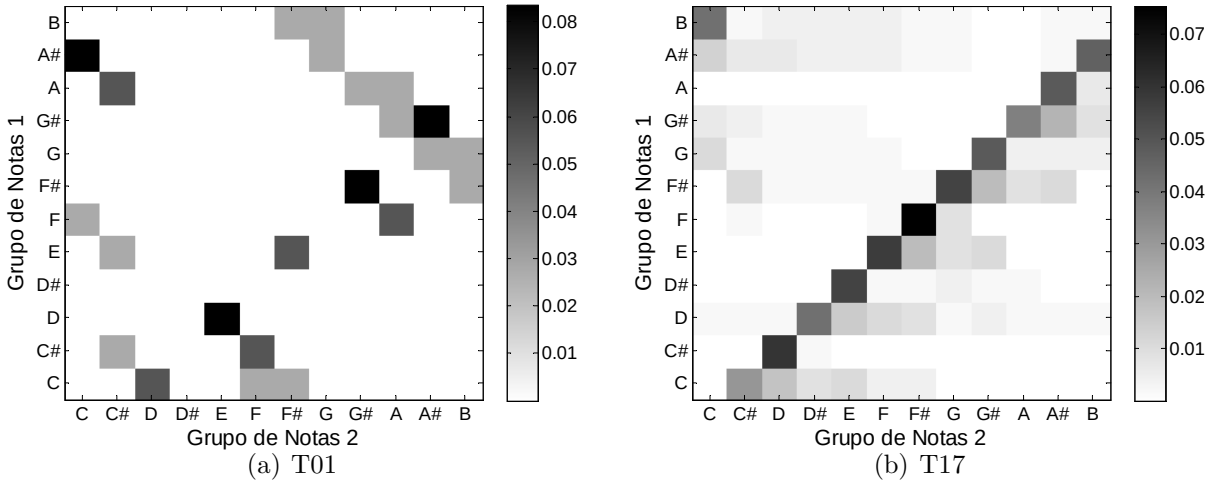
Os valores referentes aos testes T13 a T16 são apresentados na Figura 63(a). Neste quadro comparativo, é possível evidenciar que os testes T14 e T15 apresentaram a mesma proporção de distribuição de intervalos ao longo da melodia. Já a Figura 63(b), que apresenta o comparativo dos resultados dos testes T29 a T32, permite a compreensão a distribuição de

intervalos ocorreu de maneira mais uniforme. Contudo, a concentração da distribuição em ambos os casos permaneceu nas notas maiores. As distribuições de intervalos obtidas nos testes realizados evidenciaram as diferenças entre as melodias resultantes do processamento da arquitetura, corroborando, para a compreensão das aplicações da arquitetura proposta nesta dissertação.

5.4.3 Resultados de Transposição Tonal

Para aferição dos resultados de transposição tonal foram escolhidos os 4 testes propostos que utilizaram o mesmo tipo de CA, tanto para geração de notação musical quanto para geração de notação instrumental. Esta escolha baseou-se no fato de que, o CA utilizado para geração de notação instrumental tem a capacidade de alterar apenas a quantidade de notas que serão executas, exercendo pouquíssima influência na transposição tonal. Estes resultados, que têm como objetivo confirmar a validade da classificação dos produtos musicais obtidos como música aleatória, servirão também para ratificar a influência dos conjuntos de regras e de *vizinhança* no produto final resultante do processamento da arquitetura.

A Figura 64 apresenta os resultados de transposição tonal para os testes T01 e T17. Na Figura 64(a), que apresenta os resultados do teste T01, é possível evidenciar que a transposição ocorre de maneira esparsa, sendo maior o número de ocorrências nas transposições de $A\#$ para C, de D para E, de $G\#$ para $A\#$ e de $F\#$ para $G\#$. Na Figura 64(b), que apresenta os resultados do teste T17, é possível evidenciar que a transposição ocorreu de forma linear, com maior número de ocorrências para as transposições de $C\#$ para D e F para $F\#$.

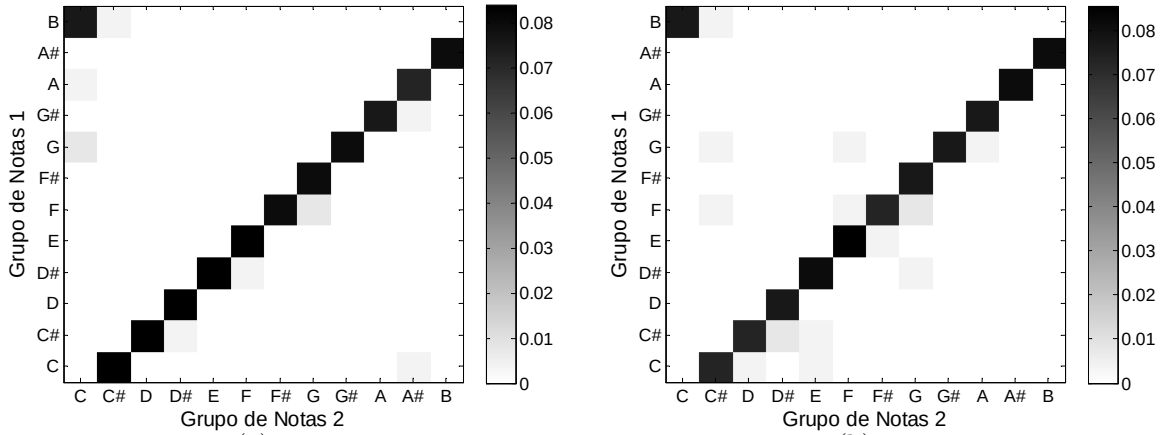


(a) T01 (b) T17

Figura 64: Comparativo das transposições tonais obtidas com o componente `UWN`

Os resultados de transposição dos testes T06 e T22 são apresentados ma Figura 65. Es-

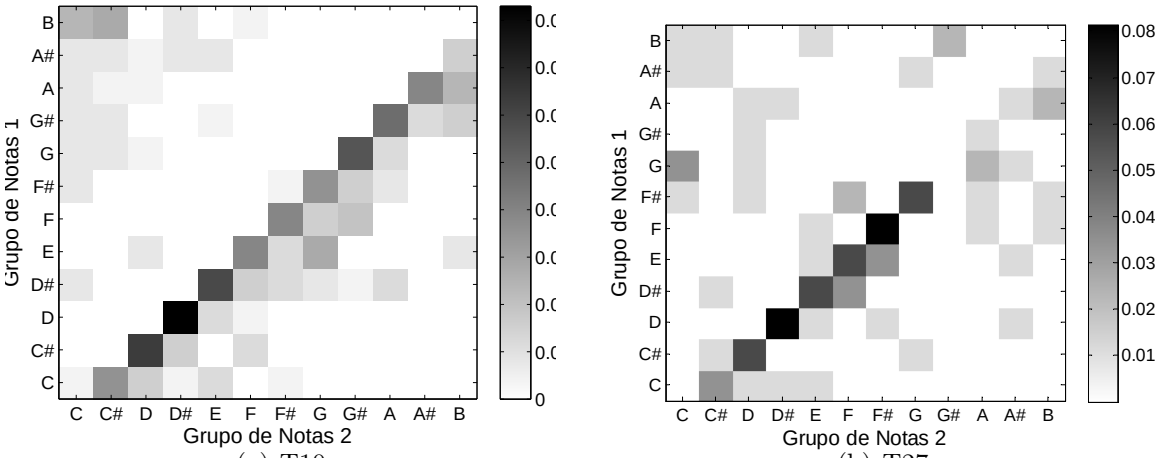
tas transposições, respectivamente Figura 65(a) e na Figura 65(b), apontam as similaridades entre os dois produtos musicais, no que tange a transposição de notas. Ambos os gráficos apresentaram as mesmas concentrações de maior transposição, sendo o diferencial as transposições de menor ocorrência, como por exemplo, no teste T06 que foi de A para C enquanto no teste T22 foi de G para $G\#$.



(a) T06 (b) T22

Figura 65: Comparativo das transposições tonais obtidas com o componente `BNN`

A Figura 66 apresenta os resultados de transposição obtidos nos testes T11 e T27.

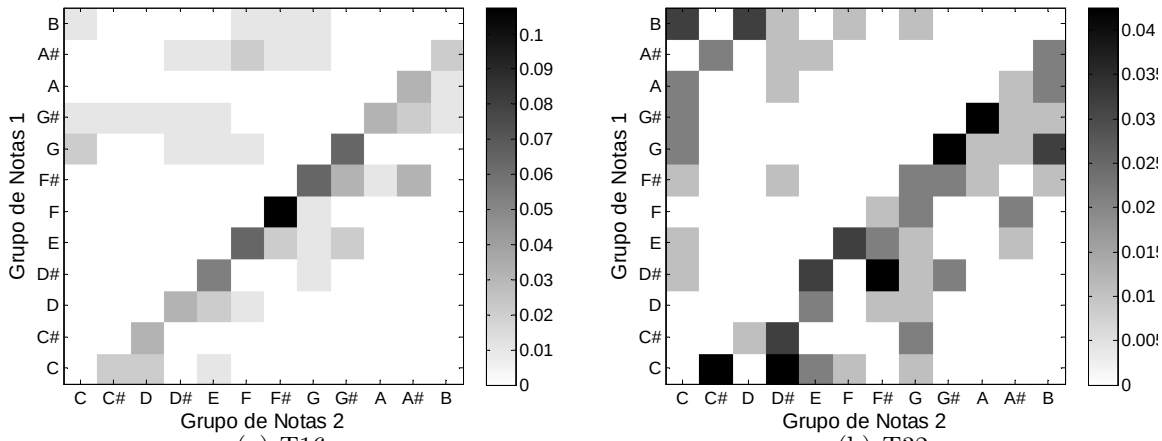


(a) T10 (b) T27

Figura 66: Comparativo das transposições tonais obtidas com o componente `BMN`

Diferentemente dos resultados evidenciados nos testes T06 e T22, estas transposições, respectivamente Figuras 66(a) e 66(b), possuem poucas similaridades. O teste T11 apresentou um comportamento linear, sendo seu maior índice de transposição de D para $D\#$. Já o teste T27 apresentou um comportamento misto, marcado pela maior ocorrência das transposições

de D para $D\#$ e de F para $F\#$ e por diversos trechos esparsos. A Figura 67 apresenta as transposições tonais obtidas nos testes T16 e T32. A transposição do teste T16, apresentada na Figura 67(a), denota um comportamento linear, de forma similar a alguns dos testes analisados anteriormente. Já na transposição obtida no teste T32, apresentada na Figura 67(b), os altos índices de concentração que ocorreram em pontos diversos, como por exemplo, de $G\#$ para $C\#$.



(a) T16 (b) T32

Figura 67: Comparativo das transposições tonais obtidas com o componente `TNN`

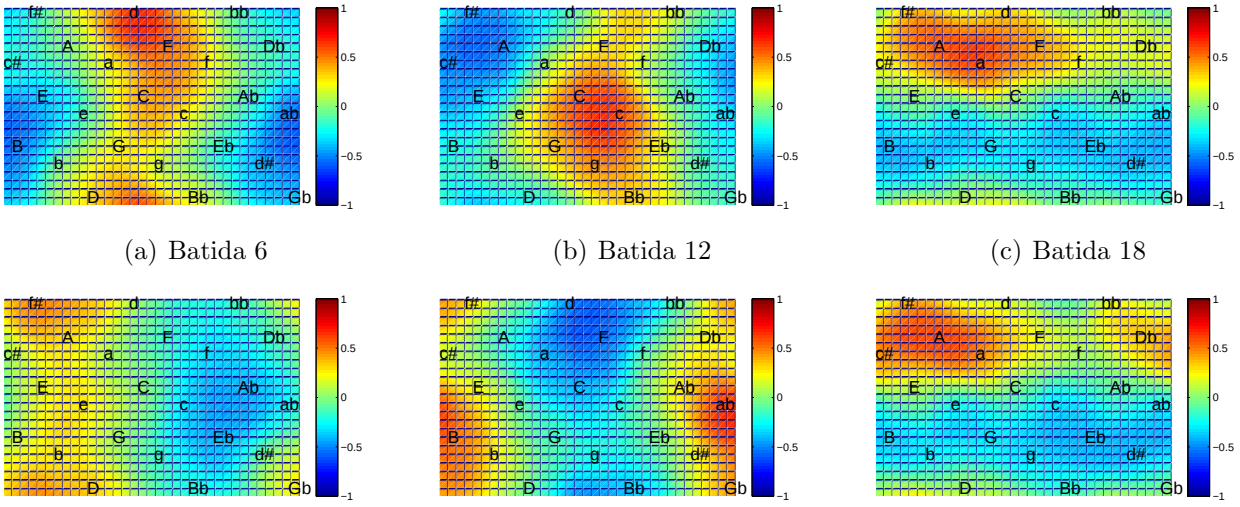
Os resultados de transposição tonal, mostraram-se diversificados para as melodias analisadas, evidenciando que quanto mais complexo o conjunto *vizinhança* mais diversificada será a combinação melódica obtida, e consequentemente, a transposição tonal.

5.4.4 Resultados de Força Tonal e Contorno Melódico

Para aferição da movimentação do *centro tonal* ao longo da melodia e a apresentação do *contorno melódico* também foram utilizados os testes T01, T06, T11, T16, T17, T22, T27 e T32. A escolha destes testes para apresentação da movimentação de *centro tonal* e do *contorno melódico* baseou-se nos resultados obtidos referentes a transposição entre duas notas. Como cada produto musical analisado possui um tempo de execução diferente, para cada uma das apresentações de movimentação de *centro tonal* serão indicados os parâmetros utilizados para obtenção dos gráficos.

A Figura 68 apresenta a movimentação de *centro tonal* ocorrida no teste T01. O arquivo audível utilizado neste teste possui um tempo pequeno de execução, com um total de 36 batidas, portanto, para obtenção dos gráficos de movimentação de centro *tonal* dividiu-se a melodia em 6 segmentos, com intervalo de 6 batidas entre cada segmento. O resultado obtido, apresentado

na Figura 68(a), indica que inicialmente o *centro tonal* era fortemente marcado em *Dm*. Ao longo da melodia o *centro tonal*, Figuras 68(b) a 68(f), se mantém próximo dos acordes menores. No final da melodia, 68(f), o *centro tonal* localiza-se próximo ao acorde *Am*.



(a) Batida 6 (b) Batida 12 (c) Batida 18

(d) Batida 24 (e) Batida 30 (f) Batida 36

Figura 68: Movimentação do *centro tonal* ocorrida no teste T01

O *contorno melódico* do teste T01 é apresentado na Figura 69. Este *contorno melódico*, dado o tamanho diminuto da melodia gerada, indica exatamente quais são as notas executadas a cada batida, apontando o *ritmo* do produto musical referente ao teste T01.

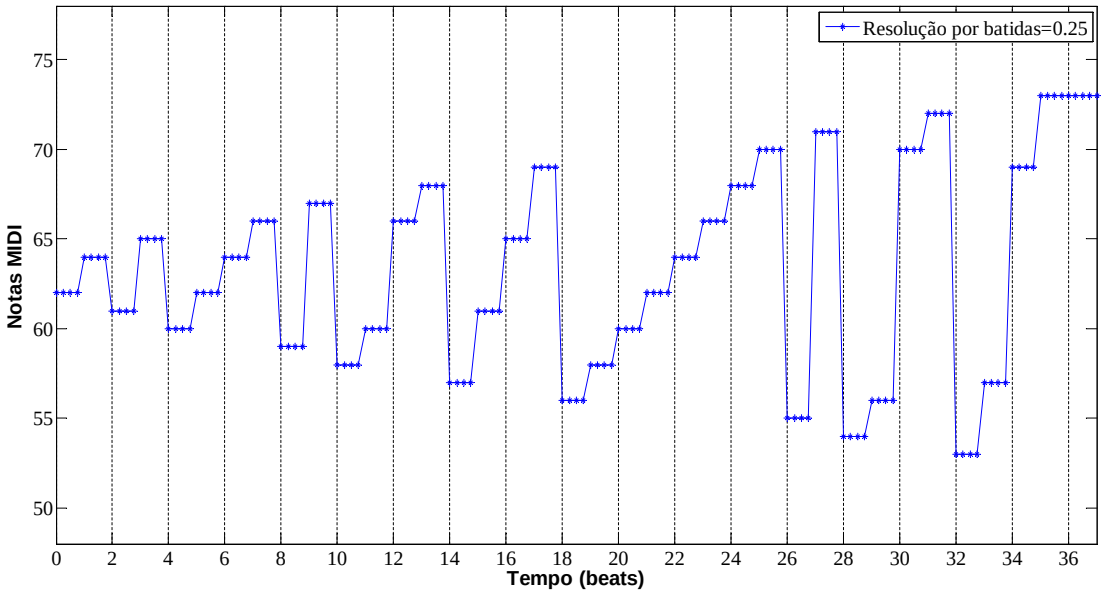
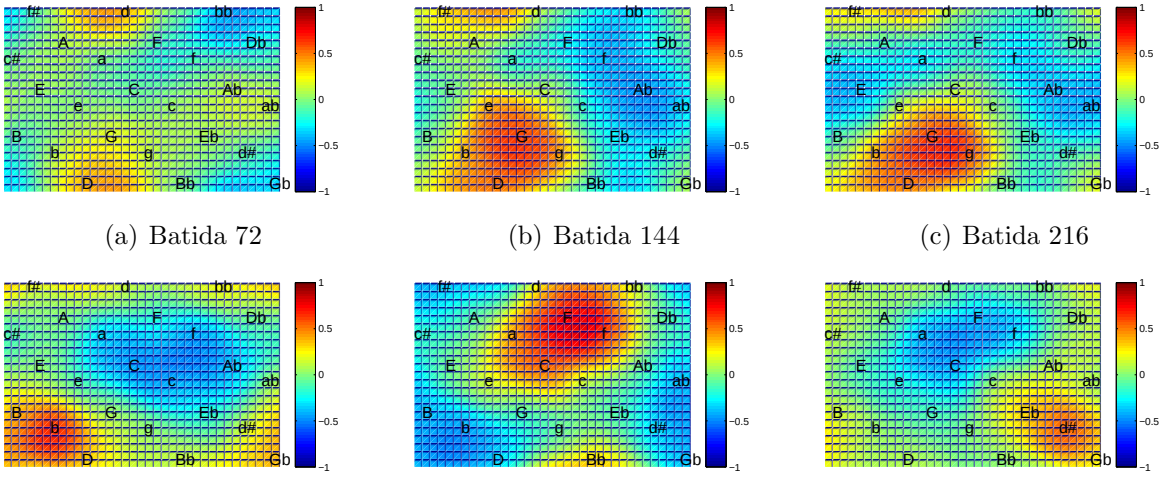


Figura 69: Contorno melódico do produto musical obtido no teste T01

A Figura 70 apresenta a *força tonal* do teste T17. Como este produto musical possui

o tempo de execução 12 vezes maior que o do teste T01, para obtenção dos dados *força tonal* dividiu-se a melodia em 6 segmentos, com 72 batidas cada. O resultado obtido indica que inicialmente a melodia utilizada não possui *centro tonal* fortemente definido, Figura 70(a). Ao longo da melodia, a movimentação do *centro tonal*, Figuras 70(b) a 70(e) varia alternadamente entre os acordes menores e maiores, e no final da melodia o *centro tonal* tende ao acorde *D#m*.



(a) Batida 72 (b) Batida 144 (c) Batida 216

(d) Batida 288 (e) Batida 360 (f) Batida 432

Figura 70: Movimentação do *centro tonal* ocorrida no teste T17

O contorno melódico das primeiras 94 batidas do teste T17 é apresentado na Figura 71. De forma similar ao contorno melódico anteriormente apresentado, este contorno também apresenta variedade *rítmica*.

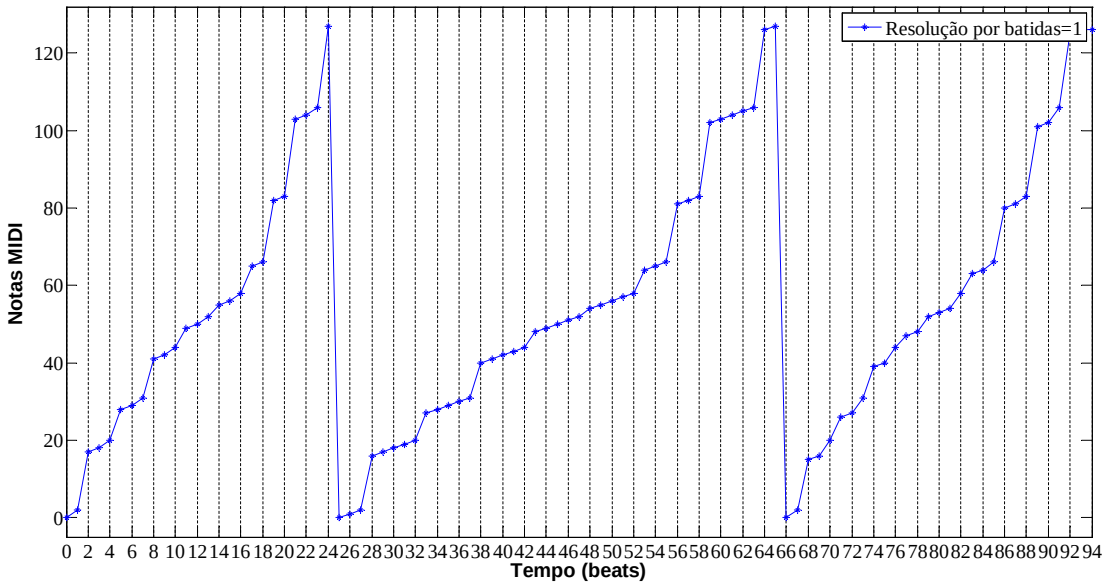
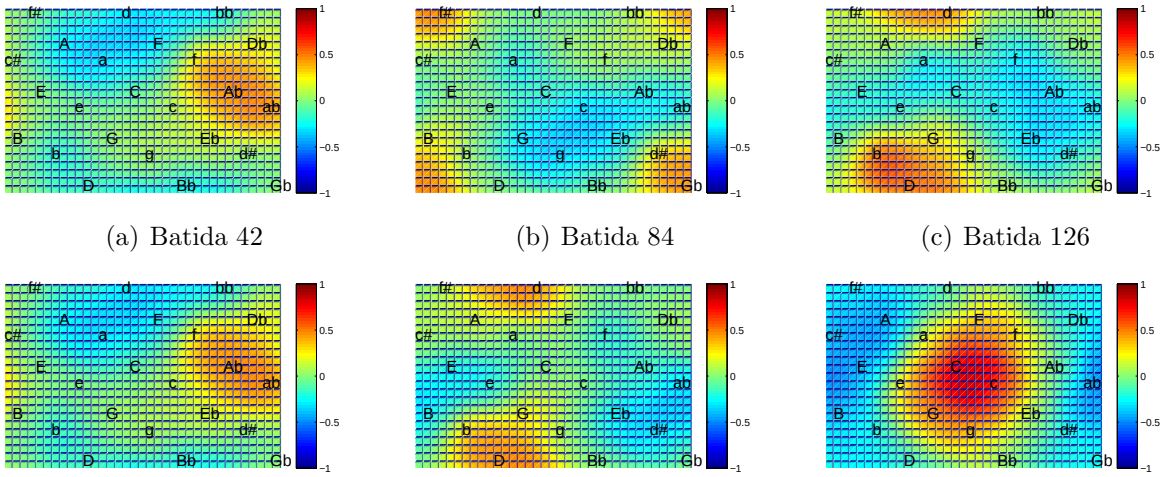


Figura 71: Contorno melódico do produto musical obtido no teste T17

Na Figura 72 é apresentada a variação de força tonal do teste T06, que foi estruturada em segmentos de 42 batidas. Inicialmente, este teste apresenta o *centro tonal* próximo aos acordes bemóis, Figura 72(a). Posteriormente o *centro tonal* se movimenta pelos acordes sustenidos e menores até retornar, Figuras 72(b) e 72(c), retornando a posição inicial, Figura 72(d), para posteriormente se fixar próximo a C e Cm, Figura 72(e).



(a) Batida 42 (b) Batida 84 (c) Batida 126

(d) Batida 168 (e) Batida 210 (f) Batida 252

Figura 72: Movimentação do *centro tonal* ocorrida no teste T06

De forma similar ao teste T17, a Figura 73 apresenta o contorno melódico das primeiras 94 batidas do teste T06.

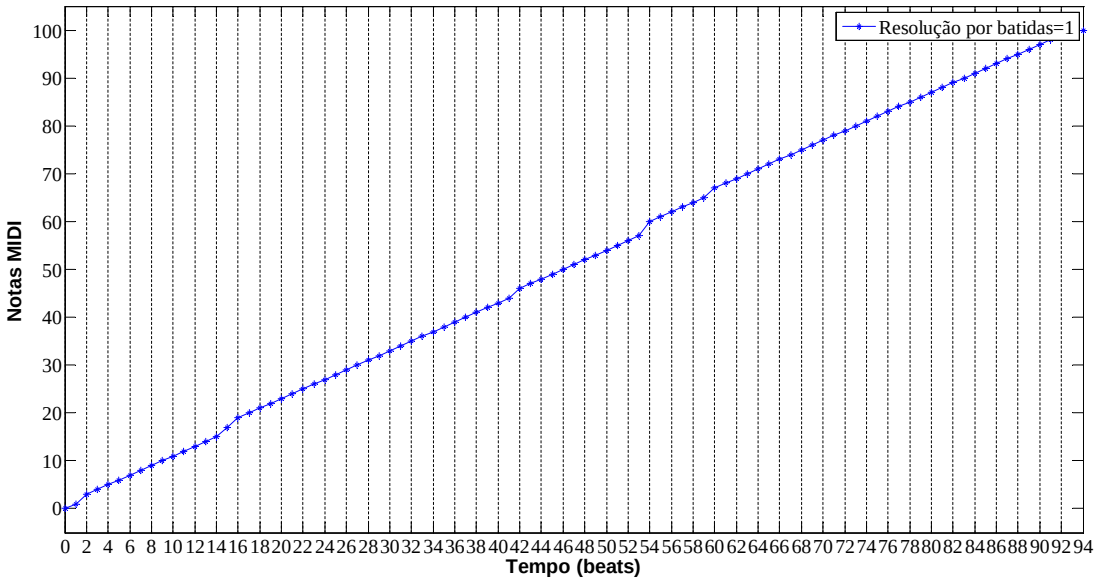
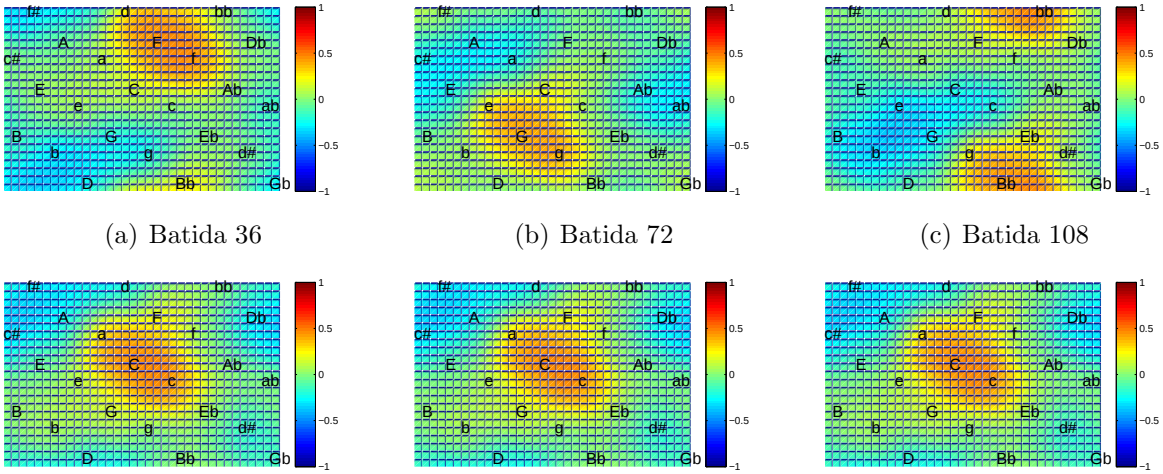


Figura 73: Contorno melódico do produto musical obtido no teste T06

O contorno apresentado na Figura 73 evidencia que o resultado musical obtido foi praticamente

o acionamento contínuo de notas ao longo do tempo, tornando a marcação rítmica auditivamente cansativa. A movimentação do *centro tonal* do teste T22 a cada 36 batidas é apresentada na Figura 74. De forma similar ao teste T06, a *força tonal* no decorrer do teste T22 não esteve fortemente definida, sendo caracterizada pela proximidade aos acordes *Am*, *C* e *Cm*.



(a) Batida 36 (b) Batida 72 (c) Batida 108

(d) Batida 144 (e) Batida 180 (f) Batida 216

Figura 74: Movimentação do *centro tonal* ocorrida no teste T22

De forma curiosa, o contorno melódico do teste T22, cujas primeiras 94 batidas são apresentadas na Figura 75, é praticamente idêntico ao do teste T06.

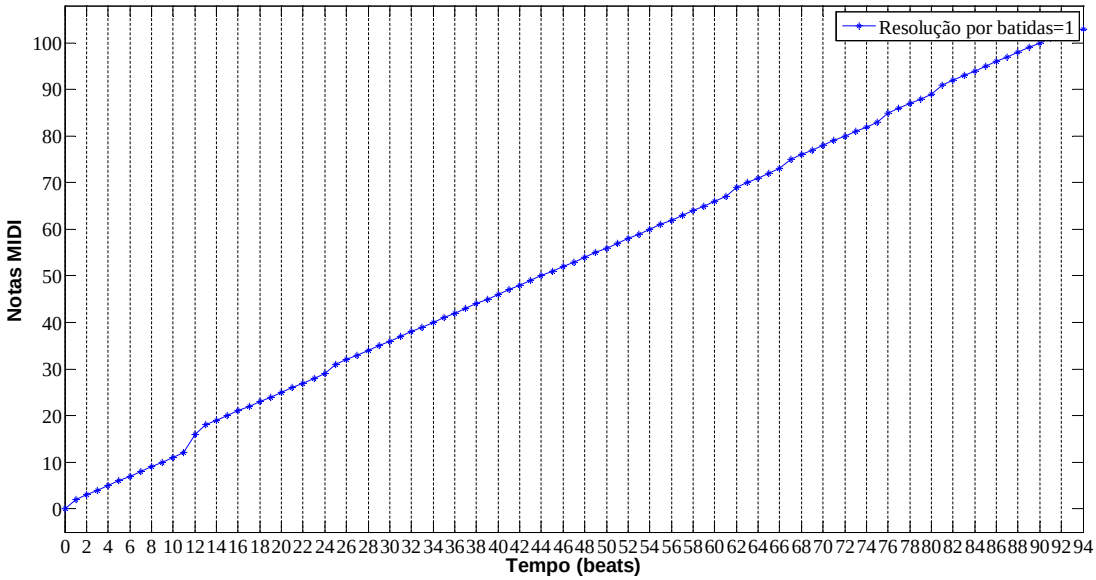
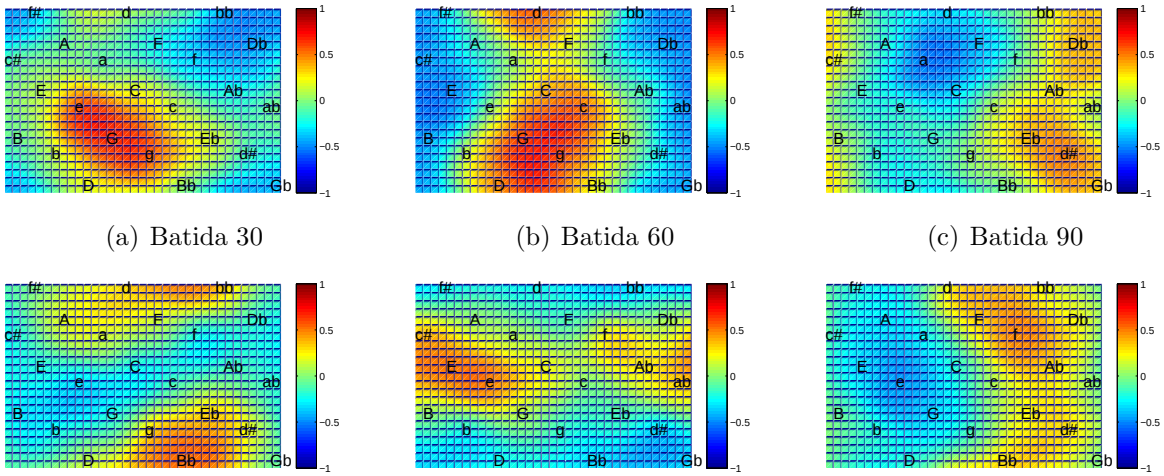


Figura 75: Contorno melódico do produto musical obtido no teste T22

A diferenciação entre estes dois contornos, apresentados nas Figuras 73 e 75, denota pouquíssimas alterações em relação as notas que não são executadas, como por exemplo, a

batida 15 do teste T06 e a batida 12 do teste T22 que marcam a primeira oscilação *rítmica* destas melodias. A cada 30 batidas o teste T11, Figura 76, apresentou uma excelente movimentação do *centro tonal*, que começa fortemente definido próximo a nota G e se espalha pelos demais acordes ao longo da melodia, Figuras 76(b) a 76(e), atingindo fracamente ao término da melodia sete notas, conforme Figura 76(f).



(a) Batida 30 (b) Batida 60 (c) Batida 90

(d) Batida 120 (e) Batida 150 (f) Batida 180

Figura 76: Movimentação do *centro tonal* ocorrida no teste T11

O contorno melódico das primeiras 94 batidas do teste T11 apresenta a marcação *rítmica* contida na Figura 77. Esta marcação, que é pseudo linear, retorna a nota inicial após 57 batidas.

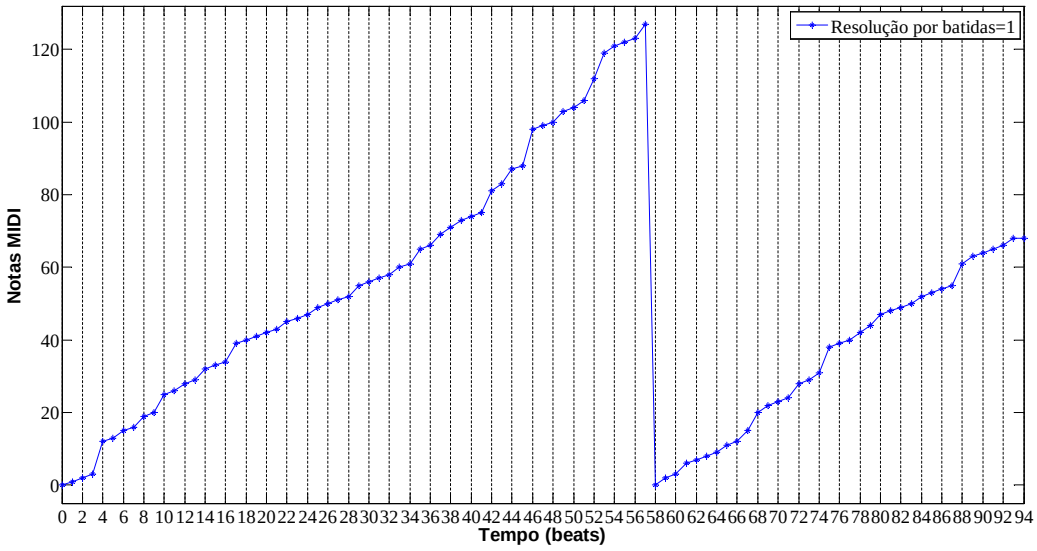
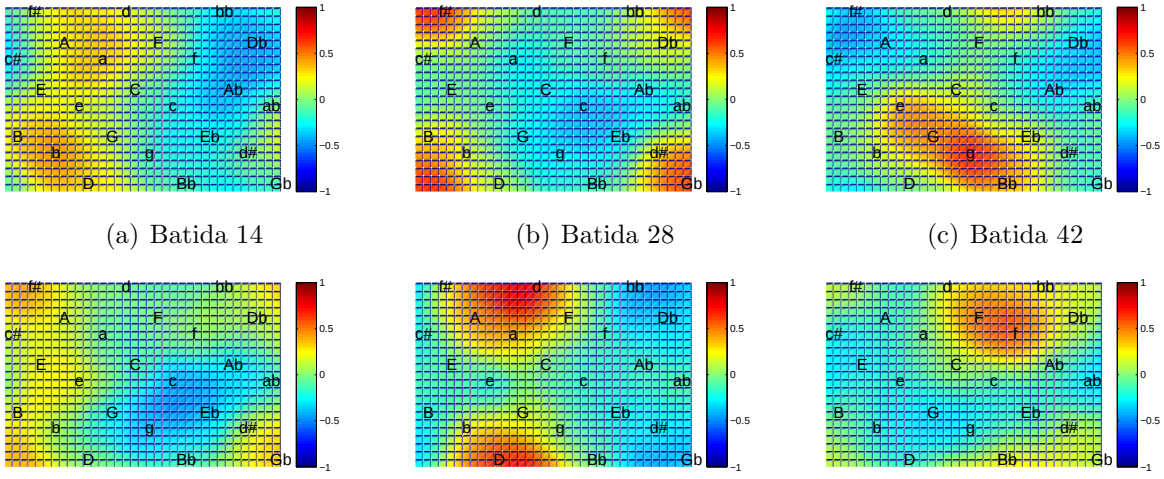


Figura 77: Contorno melódico do produto musical obtido no teste T11

Apresentada a cada 14 batidas, a movimentação do *centro tonal* do teste T27, Figura 78, começa fracamente definida entre acordes menores e maiores, Figura 78(a), tornando-se em

seguida fortemente definida entre os acordes bemóis e sustenidos, Figura 78(b). Posteriormente ocorre a transição da definição da *força tonal* dos acordes bemóis para os acordes menores e maiores, Figura 78(c). Esta oscilação continua até a ocorrência de um espalhamento da *força tonal*, Figura 78(d). Após o espalhamento o *centro tonal* torna-se fortemente definido, Figura 78(e), posicionando-se ao final da melodia próximo aos acordes *F* e *Fm*.



(a) Batida 14 (b) Batida 28 (c) Batida 42

(d) Batida 56 (e) Batida 70 (f) Batida 84

Figura 78: Movimentação do *centro tonal* ocorrida no teste T27

Esta grande movimentação do *centro tonal* reflete-se no contorno melódico do teste T27, apresentado na Figura 79.

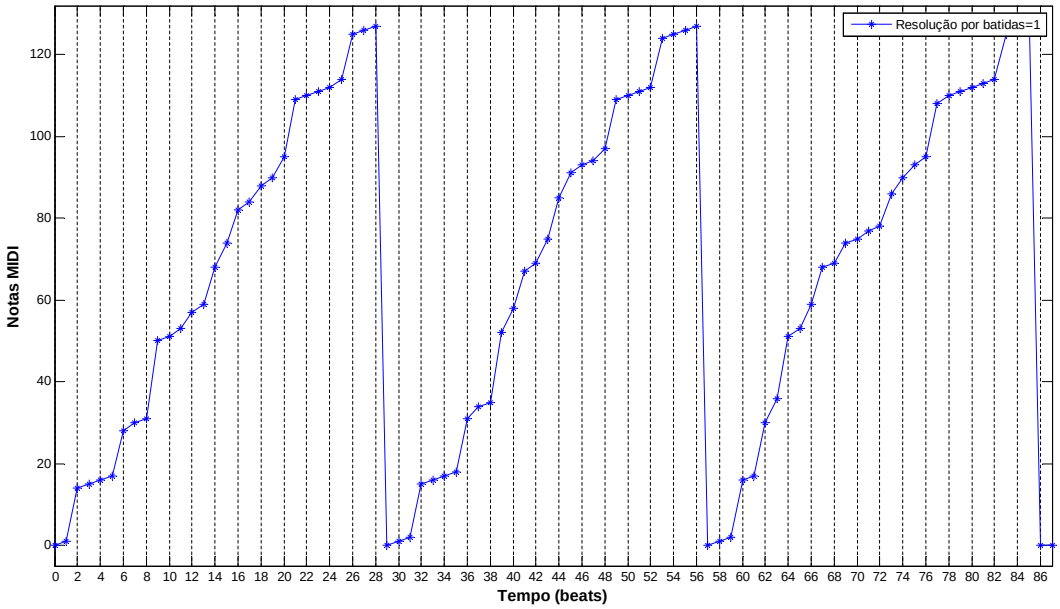
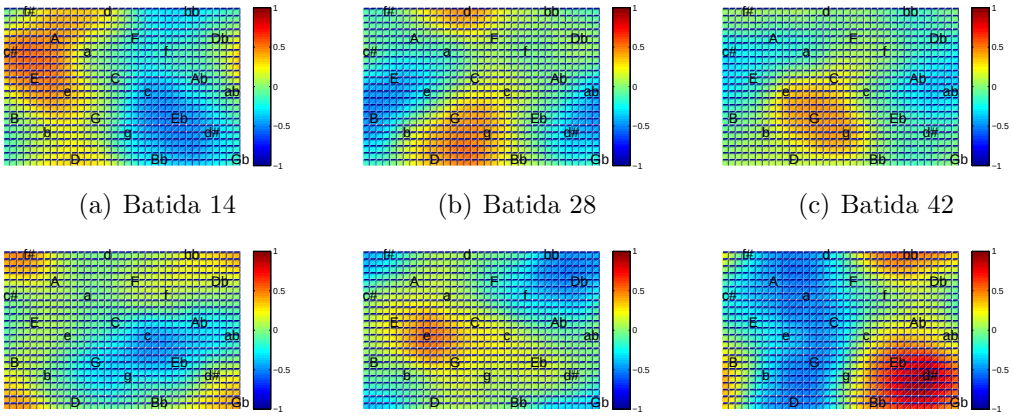


Figura 79: Contorno melódico do produto musical obtido no teste T27

A melodia do teste T27 possui marcações rítmicas bem definidas, como indica seu contorno melódico. Estas marcações oscilam de forma crescente ao longo do tempo, havendo a repetição destas oscilações a cada 30 batidas.

Ao longo das 94 batidas do teste T16, Figura 80, a força tonal começa fracamente definida, espalhando-se por diversos tipos de acordes, Figura 80(a). Este processo, que é apresentado a cada 14 batidas, continua com a definição fraca oscilando no entorno do acorde *G*, conforme Figuras 80(b) a 80(d). Próximo ao final da melodia, Figura 80(e), o *centro tonal* define-se fracamente próximo ao acorde *Em* para posteriormente se definir forte próximo ao acorde *D#*, Figura 80(f).



(a) Batida 14 (b) Batida 28 (c) Batida 42

(d) Batida 56 (e) Batida 70 (f) Batida 84

Figura 80: Movimentação do *centro tonal* ocorrida no teste T16

O contorno melódico do teste T16 é apresentado na Figura 81.

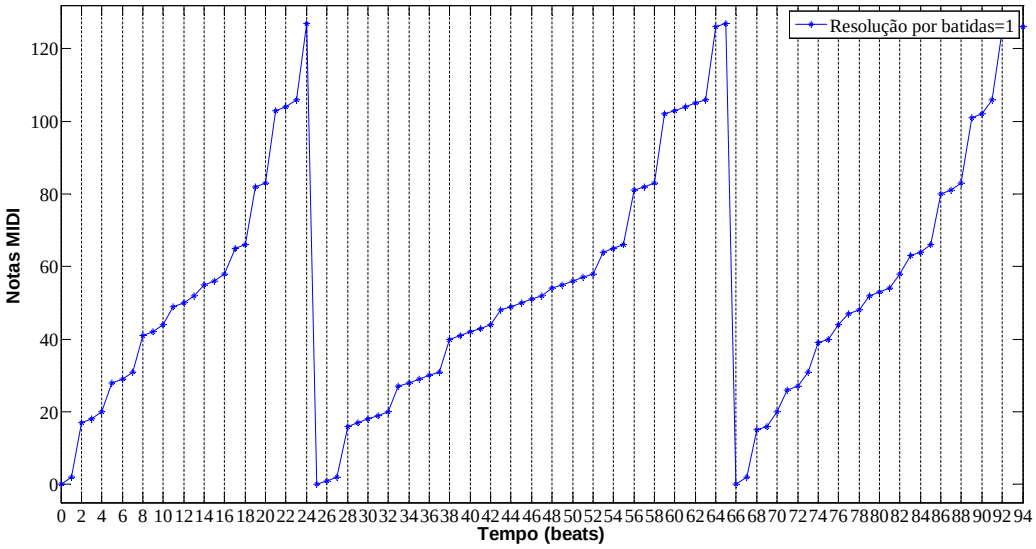
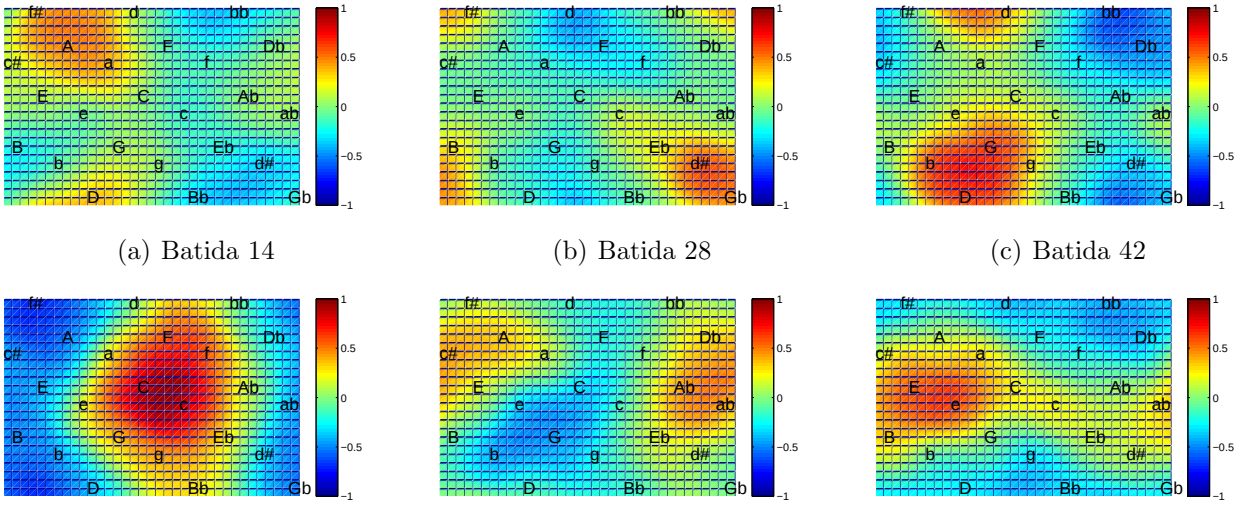


Figura 81: Contorno melódico do produto musical obtido no teste T16

O destaque do contorno melódico do teste T16, que é caracterizado por uma marcação pseudo linear ao longo de toda a melodia, é a diferença na quantidade de notas executadas nas primeiras 25 batidas para a quantidade de notas executadas nas batidas de 26 a 66.

A Figura 82 apresenta a movimentação de *centro tonal* do teste T32. Esta amostragem, realizada em intervalos de cada 14 batidas, apresentou oscilações de *centro tonal* fortemente marcado, Figuras 82(b) a 82(d), para *centro tonal* fracamente marcado, Figuras 82(b), 82(e) e 82(f). A diferença deste teste consiste no fato de que, de forma similar ao teste T27, a maior concentração de força tonal ocorreu após a metade da melodia, mas, antes do seu término. No teste T32, o melhor momento para visualização da *força tonal* ocorre na batida 56.



(a) Batida 14 (b) Batida 28 (c) Batida 42

(d) Batida 56 (e) Batida 70 (f) Batida 84

Figura 82: Movimentação do *centro tonal* ocorrida no teste T32

O *contorno melódico* do teste T32, apresentado na Figura 83, de forma idêntica ao resultado do teste T16, é marcado por oscilações no número de notas executadas ao longo da melodia, variando entre um maior número de notas ao longos as primeiras 28 batidas para um menor numero de notas ao longo das 24 batidas seguintes. As últimas 48 batidas apresentadas também possuem um número maior de notas musicais em relação às demais, esboçando o comportamento do CA ao longo do tempo. Esta característica de aumento no número de notas ao longo do tempo permite evidenciar que neste caso a evolução do CA, em função da configuração inicial e do conjunto de regras, possui como marca o aumento do número de células ativas.

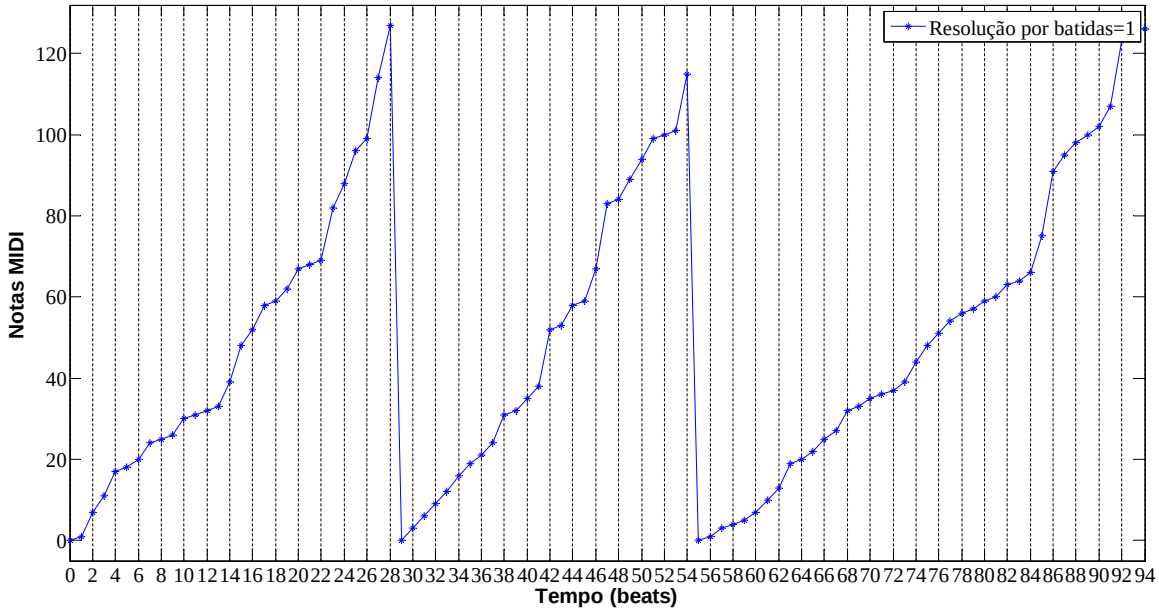


Figura 83: Contorno melódico do produto musical obtido no teste T32

Os resultados de *força tonal* e *contorno melódico* obtidos e apresentados ao longo desta seção, em conjunto com os demais resultados apresentados, permitiram uma melhor compreensão do funcionamento da arquitetura proposta e das possibilidades de implementação e utilização desta. As configurações iniciais escolhidas apresentaram-se satisfatórias para os testes realizados permitindo a visualização das diferenças e similaridades entre os resultados gerados por cada uma destas configurações.

5.5 Comparativo entre Ferramentas

Nesta seção é apresentado um comparativo entre algumas das diferentes ferramentas existentes para composição musical algorítmica e a ferramenta proposta nesta dissertação. O primeiro objetivo deste comparativo é proporcionar um panorama acerca do estado atual de desenvolvimento das ferramentas de composição algorítmica. O segundo objetivo deste comparativo é ratificar a validade da implementação destas ferramentas em *hardware*, de forma a ampliar suas aplicabilidades. Para tal, este comparativo será realizado entre a arquitetura ACH2M, proposta nesta dissertação, e os projetos *AntMusic*, (GUÉRET; MONMARCHÉ; SLIMANE, 2004), *Automatous Monk*, (REINERS, 2004), *CAMus*, (MIRANDA, 1993) e *ChaOSynth*, (MIRANDA, 1995) implementados em *software*, conforme explicitado anteriormente no Capítulo 3. A ferramenta *Wolfram Tones*, (WOLFRAM, 1994), não será utilizada para fins de comparação. A não utilização desta ferramenta foi em função de não haver publicação que descreva o

método utilizado para conversão dos resultados de iteração dos autômatos celulares em produto musical. A Tabela 35 apresenta um resumo das técnicas utilizadas para composição musical, por cada uma das ferramentas analisadas neste comparativo.

Tabela 35: Detalhamento das técnicas de composição musical por ferramenta

Ferramenta	**Técnica utilizada**
ACH2M	combinação de autômatos celulares
AntMusic	metaheurística de colônia de formigas
Automatous Monk	autômatos celulares unidimensionais
CAMus	combinação de autômatos celulares
ChaOSynth	síntese granular e autômatos celulares

De forma a evitar qualquer tipo de interferência nos resultados deste comparativo, a qualidade dos produtos sonoros foi controlada através do processo de obtenção das melodias, que foi realizado da seguinte forma:

1. Localização da melodia de demonstração no site do desenvolvedor da ferramenta, ou endereço referenciado por este;

2. Gravação, em tempo real, da melodia através do *software Audacity*, evitando qualquer tipo de perda de informação musical;

3. Armazenamento do arquivo obtido no formato *.wav*

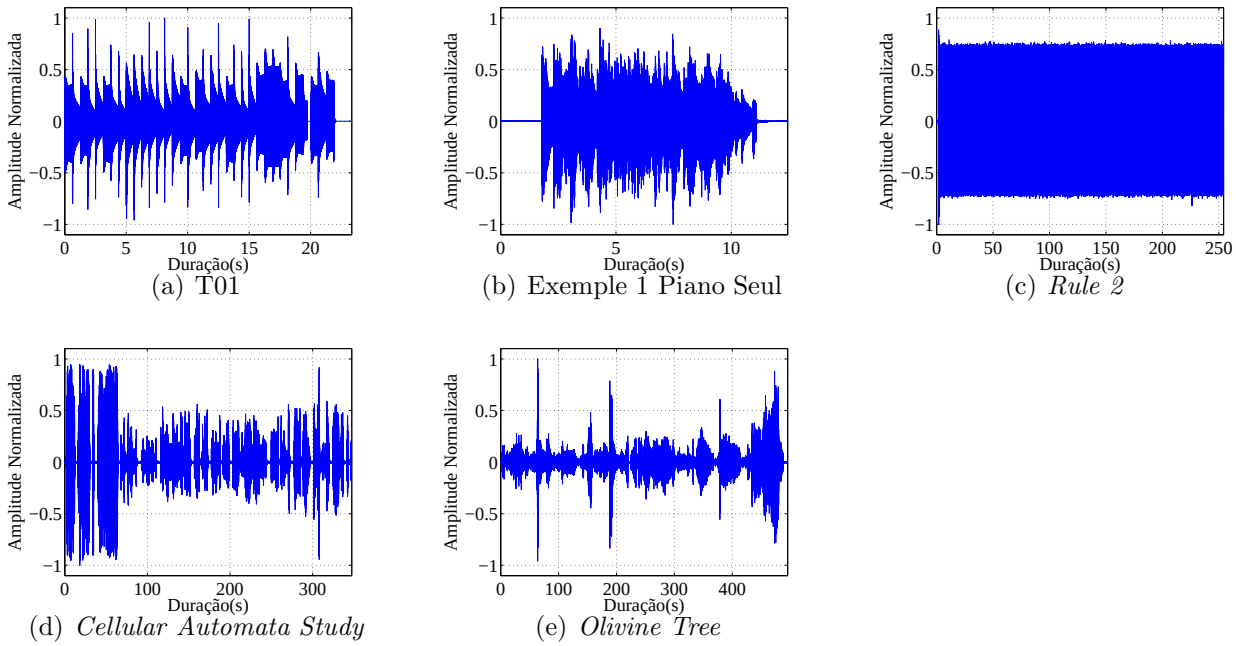
A melodia resultante da arquitetura proposta também foi convertida em formato *.wav* através da utilização de um *software* para conversão *.mid*/*.wav*. A decisão da utilização de arquivos em formato *.wav*, foi embasada nas peculiaridades de cada ferramenta. As melodias que serão comparadas são apresentadas na Tabela 36, através da ferramenta que compôs a melodia. Este detalhamento contém o tamanho dos arquivos já no formato *.wav* e a duração de cada arquivo em segundos.

Tabela 36: Detalhamento das melodias

Ferramenta	**Melodia**	**Tamanho** (*Mb*)	**Duração** (*segundos*)
ACH2M	T01	3,92	23
AntMusic	*Exemple 1 Piano Seul*	2,09	12
Automatous Monk	*Rule 2*	42,7	254
CAMus	*Cellular Automata Study*	58,4	347
ChaOSynth	*Olivine Tree*	83,3	495

A comparação entre os produtos musicais será realizada através da análise da distribuição de amplitude normalizada de cada produto sonoro. Esta comparação, apresentada na

Figura 84, indica que, com exceção da melodia *Rule 2*, Figura 84(c), que é formada por repetições da mesma frequência ao longo do tempo, as composições algorítmicas analisadas são caracterizadas pela grande variação de amplitude ao longo do tempo. Também é possível observar que, a melodia composta pela ferramenta *Ant Music*, Figura 84(b), não teve qualquer ocorrência de pausa, mesmo havendo a previsão da ocorrência deste tipo de *figura musical* no aplicativo. Já a melodia produzida pela ferramenta *CAMus*, Figura 84(d), apresenta um extenso número de *figuras de pausa*. Estas *figuras*, que ditam o ritmo da composição ao longo do tempo, são obtidas em função da redução da amplitude sonora. A *figura de pausa*, presente no instante 20 da melodia T01, Figura 84(a), ocorreu em função de um cruzamento nota × instrumento inválido. Este cruzamento inválido é a ocorrência de uma solicitação de emulação de uma nota aguda ou grave demais para um determinado instrumento. O som produzido pela ferramenta *ChaOSynth*, Figura 84(e), apresenta uma variação de amplitude linearmente uniforme ao longo do tempo. Esta variação linear de amplitude infere ao ouvinte a sensação de continuidade musical, diferentemente da melodia produzida pela ferramenta *CAMus*, que em função da intercalação de *figuras musicais*, provoca no ouvinte a sensação de recomeço da melodia, após cada *figura de pausa*.



(a) T01 (b) Exemple 1 Piano Seul (c) *Rule 2*

(d) *Cellular Automata Study* (e) *Olivine Tree*

Figura 84: Distribuição de Amplitude Normalizada

Em termo gerais, apesar de cada ferramenta possuir um método próprio para composição musical, a apresentação dos espectros de amplitude permite a compreensão da continuidade como uma das principais características das ferramentas de composição musical algorítmica.

As composições apresentadas também são caracterizadas pela repetição de comportamentos musicais, em maior ou menor escala, inferindo ao ouvinte as sensações de métrica e ritmo. Outra característica da composição algorítmica, vinculada diretamente ao conjunto de regras é a variação de amplitude, e consequentemente, de *figuras musicais* ao longo do tempo.

5.6 Considerações Finais do Capítulo

Neste capítulo foram implementadas as diversas combinações de autômatos celulares possíveis para a arquitetura proposta para obtenção do produto musical. Através do uso dos conjuntos de regras e configurações iniciais, detalhados no início deste capítulo, e partindo da exemplificação de uma música conhecida foram exibidos e comentados os resultados análise musical para cada uma das combinações possíveis, configurando um total de 32 testes de implementação. Foram apresentados ainda, os resultados de síntese para as 10 combinações possíveis de autômatos celulares para a arquitetura proposta. Também foi realizado um comparativo entre algumas das ferramentas de composição algorítmica existentes, identificando o panorama atual da composição algorítmica. O capítulo seguinte finaliza este trabalho, abordando as conclusões realizadas e enumerando os pontos considerados importantes para esta dissertação e os trabalhos futuros.

Capítulo 6

CONCLUSÕES

A ARQUITETURA proposta nesta dissertação foi descrita e implementada em *hardware*, e seu funcionamento e demanda de consumo, em área, foram validados através de simulação e síntese, apresentados no Capítulo 5, perfazendo um total de 10 diferentes combinações de autômatos celulares para síntese e 32 aplicações diferentes para avaliação de desempenho e análise de qualidade sonora. Este capítulo apresenta primeiramente as conclusões inerentes ao trabalho realizado e em seguida, as propostas de melhoria para o mesmo.

6.1 Conclusões

Esta dissertação abordou a teoria dos autômatos celulares, as definições do protocolo MIDI e aplicabilidade da junção destes para obtenção de música aleatória a partir da utilização de intervalos melódicos. Este estudo foi motivado pela carência de implementações de autômatos celulares em *hardware* e pela crescente demanda no desenvolvimento de ferramentas, em *hardware*, para produção de música aleatória, que possam auxiliar nas pesquisas relacionadas a som e música.

Seu objetivo foi o desenvolvimento de uma arquitetura em *hardware* reconfigurável, do tipo FPGA, para composição de melodias através do uso de 4 modelos diferentes de autômatos celulares para definição de intervalos e instrumentos musicais, permitindo a composição melódica a partir de um total de 16 combinações diferentes de autômatos celulares. Esta combinação de autômatos celulares, realizada a critério do usuário, é executada de forma que sempre exista um autômato celular responsável pela escolha das notas que deverão ser executadas e outro autômato celular responsável pela escolha dos instrumentos que deverão ser emulados. Estas opções, em conjunto com as demais configurações a cargo do usuário, tornam a arquitetura proposta uma boa alternativa para experimentos auditivos e também uma ferramenta para estudos acerca da evolução de autômatos celulares com *grids* de até três dimensões e raio de

vizinhança igual a 1.

O tempo de processamento necessário a obtenção dos resultados musicais manteve-se, em todos os testes, na escala de alguns microssegundos, tornando os resultados de implementação da arquitetura proposta coerentes com os objetivos determinado, permitindo a aplicabilidade da arquitetura proposta neste trabalho em tarefas que demandem processamento em tempo real. Os resultados obtidos nos testes realizados, apresentados no Capítulo 5, comprovaram a validade da utilização de autômatos celulares para geração de diferentes composições de música do tipo aleatória. Os testes realizados apontaram ainda que a qualidade sonora das melodias obtidas não é vinculada a complexidade do conjunto de regras. Esta desvinculação do resultado audível com a complexidade do conjunto de regras é denotada pelo fato de que os melhores resultados obtidos, ou seja, aqueles em que o contorno melódico apresentou comportamento aparentemente periódico, foram os testes cujos CAs tiveram uma evolução denotada pela esparsidade de células ativas, como por exemplo o teste T01. Contudo, as regras mais complexas em conjunto com configurações iniciais densas, ou seja, com uma quantidade razoável de células ativas, apresentaram uma melhor movimentação de *centro tonal*, alcançando uma grande variedade de notas musicais e melhores distribuições de intervalo, com maior incidência de acordes maiores.

Uma restrição da arquitetura proposta é o fato desta considerar todas as notas e instrumentos disponíveis no protocolo MIDI. Esta abrangência, que possibilita a obtenção das mais variadas melodias, permitindo uma vasta aplicabilidade da arquitetura proposta, desde áreas de entretenimento até aplicações na área médica, gera, em alguns momentos, a ocorrência de solicitações de execução de notas agudas ou graves demais para o instrumento emulado, inferindo ao ouvinte a sensação musical de pausas, que poderão ser demasiadamente longas em função dos resultados da iteração corrente dos autômatos celulares.

Outra restrição da arquitetura proposta é gerada pelo método adotado para o cálculo decisório utilizado no encerramento da melodia. Conforme explicitado no Capítulo 2, para que um arquivo *.mid* possa ser executado por um interpretador não podem haver erros em alguns segmentos do arquivo, denominados segmentos rígidos. Um dos segmentos de arquivo *.mid* onde não pode haver erros é o campo *Length*, que armazena o número total de *bits* do *Track* de notas e instrumentos. Como o cálculo decisório consiste na apuração dos totais de notas e instrumentos, de forma a igualá-los, caso a razão percentual entre o número de células ativas nos dois CAs utilizados, para notação musical e instrumental, seja muito diferente, a melodia tende a se tornar extremamente repetitiva.

Apesar das restrições apontadas, a arquitetura apresentada nesta dissertação demonstrou-se condizente com os objetivos propostos, gerando melodias diversificadas que contemplaram uma boa variação de resultados em termos de sequenciamento de notas e escolha de intervalos, permitindo a identificação de algumas de suas aplicabilidades. Os resultados acerca dos instrumentos emulados foram omitidos em função do fato de que a análise da qualidade sonora abordade foi realizada com base na notação musical, sendo descartados os dados de notação instrumental. Os instrumentos de percussão não foram utilizados neste trabalho pelo fato destes executarem apenas uma nota musical, inviabilizando sua utilização em função do formato proposto para a arquitetura.

É importante salientar que todas as melodias geradas são um reflexo da evolução dos autômatos celulares presentes na arquitetura, e, para melhor compreensão da influência das configurações do usuário nos resultados musicais, foram escolhidos para realização dos testes alguns conjuntos de regras publicados em artigos diversos (CONWAY, 1970; WOLFRAM, 2006; MARTINS; FYNN; OLIVEIRA, 2011).

6.2 Trabalhos Futuros

Nesta seção são citadas algumas propostas de possíveis modificações, com o intuito de melhorar a aplicabilidade da arquitetura proposta de forma que seu processamento resulte não apenas em composições a partir de intervalos melódicos, mas também, em composições de maior complexidade que utilizem intervalos harmônicos, ampliando os métodos para composição de música aleatória.

A primeira sugestão é a do aumento do número de autômatos celulares contidos na arquitetura, de forma a vincular cada dupla de autômatos celulares a uma das classe de instrumentos MIDI. Esta abordagem permitirá a caracterização dos instrumentos com a indicação das notas musicais que podem ser executadas por estes. Desta forma, serão eliminadas as incidências de notas graves ou agudas demais para determinado instrumento. Esta nova abordagem também permite a geração de melodias complexas, com a execução simultânea de dois ou mais instrumentos, aproximando a música aleatória do formato da música popular. Em termos práticos esta abordagem conduziria a formação de uma estrutura similar a uma banda musical, na qual cada dupla de autômatos celulares será responsável pela inserção de notas musicais a partir de um determinado instrumento, de forma a compor a melodia final.

Um segunda sugestão é a criação de uma *Interface Homem Máquina*, IHM, tornando mais simples para o usuário as tarefas de configuração da arquitetura e de conversão de arqui-

vos para o usuário. Desta forma a arquitetura proposta poderia ser acoplada diretamente a diferentes equipamentos, ampliando seu potencial de aplicabilidade.

Uma terceira sugestão é a melhoria do sistema rítmico da arquitetura, através de novas regras de transição para o componente `TEM`. Desta forma a escolha dos volumes e alturas musicais a serem utilizados ao longo da melodia seria efetuada com maior precisão, utilizando os conceitos de intervalos harmônicos e outros parâmetros de harmonias musicais.

Uma quarta sugestão é a experimentação de outros formatos musicais para exportação da melodia, tais como *.wav* e *.mp3*, uma vez que o uso de formatos denominados populares poderia contribuir para a difusão da arquitetura proposta e a pesquisa de novas implementações desta.

Por fim, seria interessante a utilização de um mecanismo decisório automatizado, com base em técnicas de reconhecimento de padrões, que classifique as melodias geradas através dos modelos de CA utilizados, dos conjuntos de regras e das configurações iniciais, com o intuito de fornecer ao usuário exemplos de parâmetros mais robustos para configuração da arquitetura em função do tipo de melodia desejado. Considerando a abrangência melódica da arquitetura proposta, em função do uso do protocolo MIDI, seria possível moldar a utilização da arquitetura proposta a finalidades específicas, como por exemplo, caso a atividade fim fosse o entretenimento a arquitetura seria configuradas para gerar melodias mais rápidas, caso o objetivo fosse pesquisas relacionadas ao processamento cerebral do som a arquitetura poderia ser programada para gerar tons específicos.

Apesar desta alternativa não ter sido explorada neste trabalho, uma vez que a geração de imagens não foi um dos objetivos do mesmo, os espelhos das evoluções dos CAs, utilizados em alguns pontos do Capítulo 1 para fins de exemplificação, podem ser utilizados para visualização, ampliando as possíveis aplicações da arquitetura proposta.

REFERÊNCIAS

ADOLFO, A. *O livro do músico: harmonia e improvisação para teclado, piano e outros instrumentos*. Rio de Janeiro, RJ, BRA: Irmãos Vitale, 1989. ISBN 8585426748.

ADOLFO, A. *Musica: Leitura, Conceitos, Exercicios*. Cascadura, RJ, BRA: Irmãos Vitale, 2002. ISBN 8585426780.

ALVES, A.; ASSIS, W.; ALVES, C.; SOUZA, D.; RIOS, P. Estudo do efeito da não aleatoriedade dos núcleos em nucleações e crescimento de reações simultâneas via autômato celular. *20º CBECIMAT - Congresso Brasileiro de Engenharia e Ciências dos Materiais*, Joinville, SC, BRA, Novembro 2012. Disponível em: <http://cbecimat.com.br/resumos/trabalhos_completos/503-011.doc>. Acesso em: 20/01/2013.

ANDERTON, C. The midi protocol. In: *Audio Engineering Society Conference: 5th International Conference: Music and Digital Technology*. [s.n.], 1987. Disponível em: <http://www.aes.org/e-lib/browse.cfm?elib=4666>. Acesso em: 15/07/2011.

ANDRADE, P. E. Uma abordagem evolucionária e neurocientífica da música. *Neurociencias*, v. 1, n. 1, p. 21–33, Agosto 2004. Disponível em: <http://www.katiachedid.com.br/files/noticias/4fdab6aab5813ffc49c7139840b20359.pdf>.

ARATA, H.; TAKAI, Y.; TAKAI, N. K.; YAMAMOTO, T. Free-form shape modeling by 3d cellular automata. In: IEEE. *Shape Modeling and Applications, 1999. Proceedings. Shape Modeling International'99. International Conference on*. Aizu-Wakamatsu, 1999. p. 242–247. ISBN 0-7695-0065-X.

ARCELA, A. Sobre o lado estritamente científico da música. *Humanidades*, Universidade de Brasília, v. 9, n. 3, p. 1–6, 1994. Disponível em: <http://www.cic.unb.br/docentes/arcela/publicacoes/huma1994/humanidades94.pdf>. Acesso em: 01/01/2011.

ASHENDEN, P. J. *The VHDL cookbook*. First edition. South Australia: Department of Computer Science, University of Adelaide, 1990.

ASSIS, W. *Investigação do efeito da nucleação, da velocidade de crescimento e da distribuição da energia armazenada na recristalização pelo método do autômato celular em três dimensões.* Dissertação (Mestrado) — Universidade Federal Fluminense: Campus Volta Redonda, Volta Redonda, RJ, BRA, 2006. Disponível em: <http://www.bdtd.ndc.uff.br/tde_arquivos/35/TDE-2007-12-19T094431Z-1130/Publico/Dissert-WeslleyAssis.pdf>. Acesso em: 28/03/2012.

ASSOCIATION, M. M. *General MIDI Level 1 Sound Set*. Los Angeles, CA, USA, 2013. Disponível em: <http://www.midi.org/techspecs/gm1sound.php#instrument>. Acesso em: 28/06/2013.

ASSOCIATION, M. M. et al. The complete midi 1.0 detailed specification. *Los Angeles, CA, The MIDI Manufacturers Association*, 1996.

AULETE, F. J. C.; GARCIA, H. de; NASCENTES, A. *Dicionário contemporâneo da língua portuguesa Caldas Aulete*. Rio de Janeiro, RJ, BRA: Editora Delta, 1986.

BAH, P.; CHEN, K.; CREUTZ, M. Self-organized criticality in the game of life. *Nature*, Nature Publisher Group, New York, USA, v. 342, p. 14, 1939. ISSN 0028-0836.

BAIANU, I. *Quantum Algebra and Symmetry*. [S.l.]: Symmetry, 2010. 578 p.

BARALDI, G. d. S.; ALMEIDA, L. C. d.; BORGES, A. C. d. C. Hearing loss in aging. *Revista Brasileira de Otorrinolaringologia*, ABORL-CCF Associação Brasileira de Otorrinolaringologia e Cirurgia Cérvico-Facial, São Paulo, SP, BRA, v. 73, n. 1, p. 64–70, 2007. ISSN 0034-7299.

BAYS, C. Candidates for the game of life in three dimensions. *Complex Systems*, Complex Systems Publications, Inc., Champaign, IL, USA, v. 1, n. 3, p. 373–400, 1987. ISSN 0891-2513.

BAYS, C. A new game of three-dimensional life. *Complex Systems*, Complex Systems Publications, Inc., Champaign, IL, USA, v. 5, n. 1, p. 15–18, 1991. ISSN 0891-2513.

BERTON, C. G. *Inventividade melódica: Uma outra abordagem das técnicas de análise, composiçãoe improvisação em música popular*. Dissertação (Mestrado) — Instituto de Artes - UNICAMP, Campinas, SP, BRA, 2005. Disponível em: <http://www.bibliotecadigital.unicamp.br/document/?code=vtls000385601>. Acesso em: 30/06/2011.

BEYER, W. A.; SELLERS, P. H.; WATERMAN, M. S. Stanislaw m. ulam's contributions to theoretical theory. *letters in mathematical physics*, Kluwer Academic Publishers, v. 10, n. 2, p. 231–242, 1985. ISSN 1573-0530. Disponível em: <http://link.springer.com/article/10.1007em: 30/01/2012.

BEZERRA, J.; NOGUEIRA, P. H. A. da S. A matemática e a transmissao de informaçao uma introduçao aos códigos. *Cadernos do IME - Série Matemática*, Instituto de Matemática e Estatística - UERJ, Rio de Janeiro, RJ, BRA, v. 18, 2006. ISSN 2236-2797.

BIGAND, E. Ouvido afinado. *Viver Mente & Cérebro*, Scientific American Brasil, São Paulo, SP, BRA, p. 58–63, 2005. ISSN 1807-1562.

BONINI-ROCHA, A. C.; CHIARAMONTE, M.; ZARO, M. A.; TIMM, M. I.; WOLFF, D. Observação das evidências cognitivas de aprendizagem motora no desempenho de jovens violonistas monitoradas por eletroencefalograma: um estudo piloto. *Ciências & Cognição*, Instituto de Ciências Cognitivas - UFRJ, Rio de Janeiro, RJ, BRA, v. 14, n. 1, p. 103–120, 2009. ISSN 1806-5821.

BORCHGREVINK, H. M. O cérebro por trás do potencial terapêutico da música. *Musica e Saúde*, Grupo Editorial Summus, São Paulo, SP, BRA, p. 57, 1991.

BRANCO, A.; FEKETE, S. M. W.; RUGOLO, L.; REHDER, M. I. Valor e variações da frequência fundamental no choro de dor de recém-nascidos. *Revista CEFAC*, CEFAC, São Paulo, SP, BRA, v. 8, n. 4, p. 529–35, 2006. ISSN 1982-0216.

BRAUT, C.; BROWN, K. *The Musician's Guide to MIDI*. USA: Sybex Inc., 1994. (Sybex Macintosh Library). ISBN 0782112854.

BRÄUTIGAM, A. B. Sons tônicos e complexos: Artemiev e tarkovski em stalker. *15 Congresso da Associação Nacional de Pesquisa e Pós Graduação em Música*, ANPPOM, Rio de Janeiro, RJ, BRA, 2005.

BRIA, O. N.; SOSA, R. F.; THOMAS, P. J. Compresión de audio en el espacio transformado. In: *I Congreso Argentino de Ciencias de la Computación*. Buenos Aires, ARG: Red de Universidades con Carreras en Informática (RedUNCI), 1995. Disponível em: <http://sedici.unlp.edu.ar/bitstream/handle/10915/24287/Documento_completo.pdf?sequence=1> Acesso em: 15/07/2012.

BROWN, A. R.; SORENSEN, A. C. Introducing jmusic. *Australian Computer Music Conference*, Australasian Computer Music Association - ACMA, Brisbane, AUS, 2000.

BURRASTON, D.; EDMONDS, E.; LIVINGSTONE, D.; MIRANDA, E. R. Cellular automata in midi based computer music. In: *Proceedings of the International Computer Music Conference*. Flórida, USA: The International Computer Music Association, 2004. p. 71–78.

CALDEIRA, Z. A. Victor flusser - a mísica é extraordinária. *ouvirOUver*, Uberlândia, MG, BRA, v. 1, n. 1, 2007. ISSN 1983-1005.

CAMPOS, A.; LIMA, L. V.; PINTO, M. M. *Composição Musical Encore 4.5.4 Editoração de Partitutras*. 1. ed. São Paulo, SP, BRA: Editora Erica, 2003. ISBN 8571949514.

CARDOSO, B.; MASCARANHAS, M. *Curso completo de teoria musical e solfejo*. Rio de Janeiro, RJ, BRA: Irmãos Vitale, 1973. ISBN 978-85-85188-17-7.

CARDOZO, G. d. O. *Dinâmica de populações em autômatos celulares*. Tese (Doutorado) — Instituto de Física de São Carlos - USP, São Carlos, SP, BRA, 2006. Disponível em: <http://www.teses.usp.br/teses/disponiveis/76/76131/tde-18122006-142719/pt-br.php>. Acesso em: 15/05/2012.

CARMO, L. I. C. D. *Efeitos do ruído ambiental no organismo humano e suas manifestações auditivas*. Goiânia, GO, BRA, 1999.

CARPENTER, R. H. *Neurophysiology*. Illinois, IL, USA: University Park Press, 1984. ISBN 0839120478.

CASTRO, A. J. B. D.; ALFENAS, D. A.; SHIBATA, D. P.; SOEJIMA, H. T. *Sintetizador texto-voz com autômatos adaptativos*. São Paulo, SP, BRA, 2004.

CASTRO, M.; CASTRO, R. Autômatos celulares: implementações de von neumann, conway e wolfram. *Revista de Ciências Exatas e Tecnologia*, Sistema Anhanguera de Revistas Eletrônica, v. 3, n. 3, p. 89–106, 2008. ISSN 1980-1793.

CHIARELLI, L. K. M.; BARRETO, S. d. J. A importância da musicalização na educação infantil e no ensino fundamental: A música como meio de desenvolver a inteligência ea integração do ser. *Revista Recrerte*, IACAT-CI, Santiago de Compostela , ESP, v. 3, 2005. ISSN 1699-1834. Disponível em: <http://www.iacat.com/revista/recrearte/recrearte03/musicoterapia.htm>. Acesso em: 30/07/12.

COLOMBO, F. B. *Desenvolvimento de um software para simulação atomística de processos de microfabricação baseado em autômatos celulares*. Dissertação (Mestrado) — Escola Politécnica de São Paulo, 2011.

CONWAY, J. The game of life. *Scientific American*, ACM Press, USA, v. 223, n. 4, p. 4, 1970. ISSN 0036-8733.

COPE, D. Recombinant music: using the computer to explore musical style. *Computer*, IEEE, California, CA, USA, v. 24, n. 7, p. 22–28, 1991. ISSN 0018-9162.

COPE, D.; MAYER, M. J. *Experiments in musical intelligence*. Middleton, WI, USA: AR Editions Madison, WI, 1996. ISSN 0895793377.

CORRÊA, A. G. D.; ASSIS, G. A. de; NASCIMENTO, M. do; LOPES, R. d. D. Genvirtual: Um jogo musical para reabilitação de indivíduos com necessidades especiais. *Revista Brasileira de Informática na Educação*, Comissão Especial de Informática na Educação da Sociedade Brasileira de Computação, Porto Alegre, RS, BRA, v. 16, n. 01, 2008. ISSN 1414-5685.

COSTA, C. M. *MUSICOTERAPIA E TRAUMAS DE GUERRA*. [S.l.], 1992. Disponível em: <http://biblioteca-da-musicoterapia.com/biblioteca/arquivos/artigo//1992_Clarice_Moura_Costa_traumas_de_guerra.pdf> Acesso em: 30/06/2011.

COSTA, M.; FINE, P.; BITTI, P. E. R. Interval distributions, mode, and tonal strength of melodies as predictors of perceived emotion. *Music Perception: An Interdisciplinary Journal*, University Carolina Press, USA, v. 22, n. 1, p. 1–14, 2004. ISSN 0730-7829.

COSTA, R. L. M. Apontamentos sobre o estudo da harmonia: por uma abordagem abrangente. In: *Congresso da Associação de Pesquisa e Pós Graduação em Música*. Rio de Janeiro, RJ, BRA: ANPPOM, 2005. v. 15.

CROSBY, A. W. *A Mensuração da Realidade: A quantificação e a sociedade ocidental 1250 - 1600*. São Paulo, SP, BRA: Editora UNESP/Cambridge, 1999. ISBN 8571392447.

DAVIES, P. 3d cellular automata. *5th Annual BPC Research Conference*, Bournemouth University, Bournemouth, ENG, n. 5, 2009.

DILãO, R. Autômatos celulares, máquinas de turing ou a natureza como máquina de cálculo. *Colóquio Ciências*, Fundação Calouste Gulbenkian, Lisoa, PRT, v. 12, p. 3–20, 1993. ISSN 0870-7650.

DOOLEN, G. D. *Lattice gas methods: theory, applications, and hardware*. Cambridge, MA, USA: The MIT Press, 1991. ISBN 0262540630.

DUDEQUE, N. *Harmonia Tonal II*. 2008. Disponível em: <http://www.artes.ufpr.br/material/norton/Apostila%20HTII.pdf>. Acesso em: 20/03/2013.

ECCLES, J. C. The physiology of imagination. *Scientific American*, USA, v. 199, n. 3, p. 135–42, 1958. ISSN 0036-8733.

EEROLA, T.; TOIVIAINEN, P. *MIDI toolbox: MATLAB tools for music research*. Jyväskylä, FIN, 2004. Disponível em: <https://www.jyu.fi/hum/laitokset/musiikki/en/research/coe/materials/miditoolbox/>. Acesso em: 30/03/2012.

FALCÓN, J. A. *Quatro critérios para a análise musical baseada na percepção auditiva*. Dissertação (Mestrado) — Universidade Federal do Paraná, Curitiba, PR, BRA, 2012.

FARIA, N. *A arte da improvisação*. Rio de Janeiro, RJ, BRA: Irmãos Vitale, 1991. ISBN 978-85-7407-271-5.

FIGUEIREDO, M. B.; JÚNIOR, C. T.; TRAINA, A. J. M. Partitura musical como um tipo de dado em um gerenciador de objetos. São Paulo, SP, BRA, 1997. Disponível em: <http://www2.icmc.usp.br/ caetano/artigos/Figueiredo1997_JAIIO_TDPartitura.pdf>.

FILHO, C. F. *História da computação: O Caminho do Pensamento e da Tecnologia*. Rio Grande do Sul, RS, BRA: Edipucrs, 2007. ISBN 978-85-7430-691-9.

FILHO, M. T.; HENRIQUES, M. A. A. Estudo sobre a aplicação de autômatos celulares caóticos em criptografia. *IV Encontro dos Alunos e Docentes do Departamento de Engenharia de Computação e Automação Industrial - EADCA*, Campinas, SP, Brasil, 2011.

FINCH, C. E.; PIKE, M. C. Maximum life span predictions from the gompertz mortality model. *The Journals of Gerontology Series A: Biological Sciences and Medical Sciences*, Oxford University Press, v. 51, n. 3, p. B183–B194, 1996.

FOLINO, G.; MENDICINO, G.; SENATORE, A.; SPEZZANO, G.; STRAFACE, S. A model based on cellular automata for the parallel simulation of 3d unsaturated flow. *Parallel Computing*, Elsevier, v. 32, n. 5, p. 357–376, 2006. ISSN 0167-8191.

FONSECA, N. Introdução à engenharia de som. *Lisboa: FCA– Editora de Informática, Lda., cop*, PRT, 2007.

FREITAS, J.; JAFELICE, R. d. M. Uso de autômato celular no estudo da evolução da aids. *FAMAT em revista*, MG, BRA, v. 6, p. 159–167, 2006. ISSN 1806-1958. Disponível em: <www.famat.ufu.br>. Acesso em: 30/08/2012.

FREITAS, J. A. F. d. *Redes booleanas como modelos discretos da realidade*. Dissertação (Mestrado) — Universidade do Minho, PRT, 2010. Disponível em: <http://hdl.handle.net/1822/13383>. Acesso em: 27/02/2012.

GANDIN, C.-A.; RAPPAZ, M. A 3d cellular automaton algorithm for the prediction of dendritic grain growth. *Acta Materialia*, Elsevier, v. 45, n. 5, p. 2187–2195, 1997. ISSN 1359-6454.

GARIS, H. de. An artificial brain atr's cam-brain project aims to build/evolve an artificial brain with a million neural net modules inside a trillion cell cellular automata machine. *New Generation Computing*, Springer, v. 12, n. 2, p. 215–221, 1994. ISSN 1882-7055.

GOMES, T.; FERRACIOLI, L. A investigaçao da construçao de modelos no estudo de um tópico de fisica utilizando um ambiente de modelagem computacional qualitativo. *Revista Brasileira de Ensino de Fısica*, SciELO Brasil, São Paulo, SP, BRA, v. 28, n. 4, p. 453–461, 2006. ISSN 1806-1117.

GREMONINI, L.; VICENTINI, E. Autômatos celulares: revisão bibliográfica e exemplos de implementações. *Revista Eletrônica Lato Sensu*, Editora Unicentro, Paraná, PR, BRA, v. 6, 2008. ISSN 1980-6116. Disponível em: <http://ffb.virtual.ufc.br/solar/arquivos/curso/398/aula16_1-ed6_ce-autom.pdf>. Acesso em: 31/12/2010.

GUÉRET, C.; MONMARCHÉ, N.; SLIMANE, M. Ants can play music. *Ant Colony Optimization and Swarm Intelligence*, Springer, p. 310–317, 2004.

HAANDEL, J. C. van. *Formatos emergentes de criação e transmissão de áudio online: a construção do webcasting sonoro*. Dissertação (Mestrado) — PUC/SP, 2009. Disponível em: <http://www.dominiopublico.gov.br/pesquisa/DetalheObraForm.do?select_action=&co_obra=1490 Acesso em: 30/04/2011.

HATEM, T. P.; LIRA, P. I.; MATTOS, S. S. Efeito terapêutico da música em crianças em pós-operatório de cirurgia cardíaca. *Jornal de Pediatria*, SBP, v. 82, n. 3, p. 186–92, 2006. ISSN 1678-4782.

HERRERA, C. G. Sintetizador musical analógico-uma releitura sob a ótica da eletrônica moderna. *1 Seminário Brasileiro de Engenharia de Áudio - SEMEA*, MG, BRA, 2002.

HILLER, L.; ISAACSON, L. M. *Illiac suite, for string quartet*. [S.l.]: New Music Edition, 1957.

HODGES, A. *Turing: the enigma*. Princenton, NJ, USA: Princeton University Press, 1999. ISBN 069115564X.

HOSHEN, J.; KOPELMAN, R. Percolation and cluster distribution. i. cluster multiple labeling technique and critical concentration algorithm. *Physical Review B*, APS, v. 14, n. 8, p. 3438, 1976.

IAZZETTA, F. A música, o corpo e as máquinas. *Revista Opus*, v. 4, n. 4, p. 27–44, 1997. ISSN 1517-7017.

IAZZETTA, F. O que é a música (hoje). *I Fórum Catarinense de Musicoterapia*, Associação Catarinense de Musicoterapia, Florianópolis, SC, BRA, v. 1, p. 5–14, 2001.

IAZZETTA, F. A performance interativa em pele. In: *Proceedings of the 9th Brazilian Symposium on Computer Music*. Campinas, SP, BRA: [s.n.], 2003. p. 249–255.

IAZZETTA, F.; KON, F. A música efêmera da internet. *XII Encontro Anual da Associação Nacional de Pesquisa e Pós Graduação em Música*, Associação Nacional de Pesquisa e Pós-Graduação em Música, Campinas, SP, BRA, 1998.

II, C. P. *Apostila de Educação Musical*. http://www.portaledumusicalcp2.mus.br/apostilas/pdfs/9ano_00_apostila2011.

ILARI, B. Música, comportamento social e relações interpessoais. *Psicologia em Estudo*, Departamento de Psicologia - Universidade Estadual de Maringá, Maringá, PR, BRA, v. 11, n. 1, p. 191–198, 2006. ISSN 1413-7372.

JUILLE, H.; POLLACK, J. B. Coevolving the ideal trainer: Application to the discovery of cellular automata rules. In: *Proceedings of the Fifth Annual Conference on Evolutionary Programming*. USA: MIT Press, 1998.

JUNIOR, H. C. *Desenvolvimento de ferramentas para manipulação e geração de arquivos MIDI-SMF em linguagem funcional clean*. Dissertação (Mestrado) — Universidade Federal de Uberlândia, Uberlândia, MG, BRA, 2007. Disponível em: <http://repositorio.ufu.br/handle/123456789/239>. Acesso em: 30/06/2012.

KORKIN, M.; GARIS, H. de; GERS, F.; HEMMI, H. Cam-brain machine (cbm): A hardware tool which evolves a neural net module in a fraction of a second and runs a million neuron artificial brain in real time. *Autonomous Robots*, Kluwer Academic Publishers, MA, USA, v. 10, p. 235–249, 1997. ISSN 0929-5593.

LEVEK, K.; SANTIAGO, D. A percepção de emoção em música com relação a estilos musicais: um estudo com crianças de 6-11 anos de idade na cidade de salvador-ba. In: *VIII Simpósio de Cognição e Artes Musicais*. Campinas, SP, BRA: Núcleo Interdisciplinar de Comunicação Sonora da Universidade de Campinas, 2012.

LI, W.; PACKARD, N. et al. The structure of the elementary cellular automata rule space. *Complex Systems*, Santa Fe Institute, v. 4, n. 3, p. 281–297, 1990.

LIMA, A. C.; FERREIRA, J. C. Automatic generation of cellular automata on fpga. *REC 2013 - IX Jornadas sobre Sistemas Reconfiguráveis*, Instituto de Sistemas e Robótica - Universidade de Coimbra, Coimbra, PRT, p. 51, 2013.

LIMA, E. B. *Modelos microscópicos para simulação do tráfego baseados em autômatos celulares*. Tese (Doutorado) — Universidade Federal Fluminense, 2007.

LIMA, L. V.; MACHADO, A. C.; PINTO, M. M. *Cakewalk Sonar 2.0: Seqüenciamento e Técnicas de Estúdio Audiodigital*. 2. ed. São Paulo, SP, BRA: Editora Érica Ltda, 2002. ISBN 8571949107.

LISBOA, L. Música e barulho: A bela e a fera. *Revista Superinteressante*, n. 131, agosto 1998. ISSN 1677-4280.

LOUREIRO, M. A. A pesquisa empírica em expressividade musical: métodos e modelos de representação e extração de informação de conteúdo expressivo musical. *Associação Nacional de Pesquisa e Pós Graduação em Música -ANPPOM*, Brasília, DF, BRA, p. 7, 2006.

MACEDO, A. *Espectro Audível e Visivel - proposta de correspondência*. Tese (Doutorado) — Universidade Católica Portuguesa, 2009.

MACHADO, R. C. *Abc Musical*. Rio de Janeiro, RJ, BRA: Irmãos Vitale, 1941. ISBN 978-85-85188-92-4.

MACHADO, R. S. A. A composição na educação musical: Sentidos, caminhos e descobertas. *IX Seminário Nacional de Pesquisa em Música da UFG*, Universidade Federal de Goiás, p. 52, 2009. ISSN 1982-3061.

MANCINI, O. M. M. *Uma breve história do MIDI*. Campinas, SP, BRA: [s.n.], 2003. Disponível em: <http://www.iar.unicamp.br/disciplinas/am005_2003/midi.pdf>. Acesso em: 30/06/2011.

MANSON, M. *The beautiful people*. http://www.myworldit.com.br/bateria/partituras/Marylin%20Maison%20-%20The%20Beautiful%20People.pdf: Nothing/Interscope, 1997.

MARIA, G. C. J. e N. T. *Vida Artificial*. SC, BRA, 2003. Trabalho do Curso de Ciências da Computação 6ª Fase.

MARTINS, C. C.; GROSSA, P. *Autômato Celular aplicado no crescimento de câncer*. Dissertação (Mestrado) — Universidade Estadual de Ponta Grossa, Ponta Grossa, PR, BRA, 2010. Disponível em: <http://fisica.uepg.br/ppgfisica/Public/Projetos/1316461681_.pdf>. Acesso em: 30/07/2012.

MARTINS, L. G.; FYNN, E.; OLIVEIRA, G. Algoritmos genéticos usados na obtenção de autômatos celulares para resolução da tarefa de classificação de densidade no espaço bidimensional. *Horizonte Científico*, Universidade Federal de Uberlândia, Uberlândia, MG, BRA, v. 5, n. 2, 2011. ISSN 1808-3064. Disponível em: <http://www.seer.ufu.br/index.php/horizontecientifico/article/view/7288>. Acesso em: 30/05/12.

MASCARENHAS, M.; CARDOSO, B. *Curso Completo de Teoria Musical e Solfejo*. Rio de Janeiro, RJ, BRA: Irmãos Vitale, 1973. ISBN 978-85-85188-17-7.

MATHWORKS, M. U. G. the mathworks. *Inc., Natick, MA*, 1992.

MATTOS, F. L. D. *SISTEMAS DE CIFRAGEM DE ACORDES*. RS, BRA, 2010. Disponível em: <http://prolicenmus.ufrgs.br/repositorio/moodle/material_didatico/sistemas_organizacao_sonora/tu Acesso em: 30/06/2012.

MELOTTI, G. *Aplicação de Autômatos Celulares em Sistemas Complexos: Um Estudo de Caso em Espalhamento de Epidemias*. Dissertação (Mestrado) — PPGEE, UFMG, Belo Horizonte, MG, BRA, 2009.

MESQUITA, F. Significado musical e significado linguístico: vantagens metodológicas na comparação. In: *I Workshop Internacional de Pragmática*. Curitiba, PR, BRA: Universidade Federal do Paraná, 2012. p. 94.

MILETTO, E. M.; COSTALONGA, L.; FLORES, L.; FRITSCH, E.; PIMENTA, M.; VICARI, R. Introdução à computação musical. In: *IV Congresso Brasileiro de Computação*. Rio Grande do Sul, RS, BRA: Universidade Federal do Rio Grande do Sul, 2004.

MILIES, C. P. Breve introduçaoa teoria dos códigos corretores de erros. *COL OQUIO DE MATEM ÁTICA DA REGI AO Centro Oeste*, Sociedade Brasileira de Matemática, Campo Grande, MS, BRA, 2009.

MIRANDA, B. A.; MELO, C.; ALBUQUERQUE, J.; BOCANEGRA, S.; SOUZA, H.; SANTOS, R. et al. Autômatos celulares aplicados à epidemiologia da esquistossomose em pernambuco-uma análise comparativa do processo de coleta de moluscos. In: *Anais do XXXI Congresso Nacional de Matemática Aplicada e Computacional.* Belém, PA, BRA: Sociedade Brasileira de Matemática Aplicada e Computacional, 2008. p. 630–6.

MIRANDA, E. R. Cellular automata music: an interdisciplinary project. *Journal of New Music Research*, Taylor & Francis, v. 22, n. 1, p. 3–21, 1993. ISSN 0929-8215. Disponível em: <http://www.tandfonline.com/toc/nnmr20/current#.UjsZ9dKwXE0>. Acesso em: 13/09/2012.

MIRANDA, E. R. Granular synthesis of sounds by means of a cellular automaton. *Leonardo Music Journal*, MIT Press, USA, p. 297–300, 1995. ISSN 09611215.

MIRANDA, E. R. Evolving cellular automata music: From sound synthesis to composition. In: *Proceedings of 2001 Workshop on Artificial Life Models for Musical Applications*. Prague, CZE: [s.n.], 2001.

MORI, P.; DEUTSCH, S. Alterando estados de ânimo nas aulas de ginástica rítmica com e sem a utilização de música. *Motriz*, Universidade Estadual Paulista, Rio Claro, SP, BRA, v. 11, n. 3, p. 161–6, 2005. ISSN 1980-6574.

MULLER, A. P. O. *Dinâmica de domínios e difusão em autômatos celulares determinísticos*. Dissertação (Mestrado) — Universidade Federal do Rio Grande do Sul. Instituto de Física. Programa de Pós-Graduação em Física., 2008.

MUSZKAT, M.; CORREIA, C. M.; CAMPOS, S. M. Música e neurociências. *Rev Neurociências*, São Paulo, SP, BRA, v. 8, n. 2, p. 70–75, 2000. ISSN 0104-3579.

NAVABI, Z. *VHDL: Analysis and modeling of digital systems*. http://www.mhprofessional.com/: McGraw-Hill Professional, 1997. ISSN 0070464790.

NIERHAUS, G. Cellular automata. *Algorithmic Composition: Paradigms of Automated Music Generation*, Springer, p. 187–204, 2009.

NOBRE, J. *Apostila de Teoria Musical*. CE, BRA, 2008. Projeto Fortalecimento Musical. Disponível em: <http://www2.secult.ce.gov.br/Recursos/PublicWebBanco/Partituraacervo/Apt000002.pdf>. Acesso em: 25/07/2012.

NOCITI, J. R. Music and anesthesia. *Revista Brasileira de Anestesiologia*, Sociedade Brasileira de Anestesiologia, Botafogo, RJ, BRA, v. 60, n. 5, p. 455–456, 2010. ISSN 0034-7094.

ODA, A. K.; BERNARDI, A.; AZEVEDO, M. Comparação dos limiares auditivos tonais determinados por tom puro e por tom modulado. *Revista CEFAC*, CEFAC, São Paulo, SP, BRA, v. 5, n. 2, p. 149–156, 2003. ISSN 1982-0216.

OLIVEIRA, E. I. d. *Um sistema inteligente de ensino assistido por computador*. Dissertação (Mestrado) — Universidade de Aveiro, PRT, 2009. Disponível em: <http://ria.ua.pt/handle/10773/2914>. Acesso em: 15/04/2012.

OLIVEIRA, G.; OMAR, N.; OLIVEIRA, P. P. de. Computação evolucionária em autômatos celulares unidimensionais. *Revista Mackenzie de Engenharia e Computação*, Editora Mackenzie, São Paulo, SP, BRA, v. 1, n. 1, 2010. ISSN 1518-7470.

OLIVEIRA, G. M. B. de. *Dinâmica e evolução de autômatos celulares unidimensionais*. Tese (Doutorado) — Centro de Tecnologia Aeroespacial - CTA/ITA, São José dos Campos, SP, BRA, 1999.

OLIVEIRA, N. A física da música. *Revista Eletrônica de ciências*, USP, São Carlos, SP, BRA, n. 25, abril 2004. Disponível em: <http://www.cdcc.usp.br/ciencia/artigos/art_25/musica.html>. Acesso em: 02/02/2012.

OLIVEIRA, V.; OLIVEIRA, L.; RIOS, P.; CASTRO, J. Desenvolvimento de um código em 3d para simular a recristalização pelo método do autômato celular. *Tecnologia em Metalurgia e Materiais*, Associação Brasileira de Metalurgia, Materiais e Mineração, São Paulo, SP, BRA, v. 2, n. 4, p. 24–39, 2006. ISSN 2176-1523.

PACKARD, N. H. *Adaptation toward the edge of chaos*. IL, USA: University of Illinois at Urbana-Champaign, Center for Complex Systems Research, 1988.

PASCOAL, F. S. Sociedade artificial fight4life: Autômato celular modelando vida artificial. *Relatório Final de Projeto de Iniciação Científica-PIBIC/CNPq/INPE. Instituto Nacional de Pesquisas Espaciais*, 2005.

PASCOAL, M. L.; PASCOAL, A. Estrutura tonal: harmonia. *Livro eletrônico: www. cultvox. com. br*, 2012.

PEREIRA, A. L. *Notas de aula. Curso de Teoria do Fluxo de Tráfego*. Rio de Janeiro,RJ, BRA, 1988.

PEREIRA, L. D. *Processamento auditivo central: manual de avaliação; Central auditory processing: evaluation manual*. São Paulo, SP, BRA: Lovise, 1997.

PIGNATARI, D. *O que é comunicação poética*. Cotia, SP, BRA: Atelie Editorial, 2005. ISBN 85-7480-208-5.

PINTO, T. d. O. Som e música. questões de uma antropologia sonora. *Revista de Antropologia*, Departamento de Antropologia - FFLCH/USP, São Paulo, SP, BRA, v. 44, n. 1, p. 222–286, 2001.

QI, A.-S.; ZHENG, X.; DU, C.-Y.; AN, B.-S. A cellular automaton model of cancerous growth. *Journal of theoretical biology*, Elsevier, v. 161, n. 1, p. 1–12, 1993. ISSN 0022-5193.

RANGEL, J. J. de A.; BALTA, A. G. C.; VICENTE, L. C. Simulação computacional de microrganismos com autômatos celulares. *XXXVIII SIMPÓSIO BRASILEIRO DE PESQUISA OPERACIONAL*, Sociedade Brasileira de Pesquisa Operacional, Goiânia,GO, BRA, 2006.

RANSBEECK, S. V. *Composição com Autômatos Celulares*. Dissertação (Mestrado) — ESMAE: Escola Superior de Música e das Artes do Espetáculo. Instituto Politécnico do Porto, Porto, PRT, 2009.

RATTON, M. Escalas musicais-quando a matemática rege a música. *Áudio, Música e Tecnologia*, Rio de Janeiro, RJ, BRA, 1993. ISSN 1414-2821.

REINERS, P. Automatous monk. 2004. Disponível em: <http://www.automatous-monk.com/index.shtml>.

ROSSUM, D.; JOINT, E. The soundfont® 2.0 file format. *Disponível em: http://www. soundfont. com*, 1995.

SCHOENBERG, A. *Fundamentos da composição musical*. São Paulo, SP, BRA: Edusp, 1990. ISBN 9788531400452.

SCLIAR, E. *Elementos de teoria musical*. 7. ed. São Paulo, SP, BRA: Novas Metas, 1986. (Didática).

SELFRIDGE-FIELD, E. *Beyond MIDI: the handbook of musical codes*. USA: MIT press, 1997. ISBN 0262193949.

SETTI, M. d. O. G. *O Processo de Discretização do Raciocínio Matemático na Tradução para o Raciocínio Computacional: Um Estudo de Caso no Ensino/Aprendizagem de Algoritmos*. Tese (Doutorado) — Universidade Federal do Paraná, Curitiba, PR, BRA, 2009.

SHAW, P.; COCKSHOTT, P.; BARRIE, P. Implementation of lattice gases using fpgas. *Journal of VLSI signal processing systems for signal, image and video technology*, Springer, v. 12, n. 1, p. 51–66, 1996.

SILVA, A. Konzen da. *Máquina de Turing*. Disponível em: http://cesta2.cinted.ufrgs.br/xmlui/handle/123456789/63: [s.n.], 2010.

SILVA, H.; MARTINS, M. A cellular automata model for cell differentiation. *Physica A: Statistical Mechanics and its Applications*, Elsevier, v. 322, p. 555–566, 2003.

SILVA, M. A. B. da. Padrão midi. *Laboratório MIDIACOM*, UFF, Niterói, RJ, BRA, 2007. Disponível em: <http://www.midiacom.uff.br/ debora/fsmm/trab-2007-1/midi.pdf>.

SMIT, C. *Characterization of the singing voice from polyphonic recordings*. Tese (Doutorado) — Columbia University, 2011.

SOLIMANDO, S. *Improvisação baseada nos conceitos do Campo harmônico ou Centro Tonal (Improvisação Diatônica)*. 2013. Disponível em: <http://www.solimando.com/>. Acesso em: 01/07/2013.

SOUSA, A.; BESSA, F. Podcast e utilização do software audacity. *2008). Manual de Ferramentas da Web*, v. 2, p. 41–56, 2008.

STANDARD, M. File (smf) format. *MIDI Manufacturer's Association*, 1999. Disponível em: <http://www.midi.org/techspecs/smf.php>.

SUHADOLNIK, N. P. et al. *Eventos extremos em mercados complexos*. Tese (Doutorado) — Programa de Pós-Graduação em Economia - Universidade Federal de Santa Catarina, SC, BRA, 2012.

TADDEI, R. d. C. A teoria de hugo riemann: Além da harmonia simplificada ou funções tonais dos acordes. *Anais do SIMPOM*, UNIRIO, Rio de Janeiro, RJ, BRA, v. 2, n. 2, 2012.

TAVARES, L. D. *Um simulador de tráfego urbano baseado em autômatos celulares*. Dissertação (Mestrado) — Escola de Engenharia - Universidade Federal de Minas Gerais, 2010.

TERRA, V. *Acaso e aleatório na música*. SP, BRA: Editora da PUC-SP, 2000. ISBN 85-283-0201-6.

TOFFOLI, T.; MARGOLUS, N. *Cellular Automata Machines: A new environment for modelling*. USA: MIT press, 1987. (Mit Press Series in Scientific Computation). ISBN 0262200600.

TOIVIAINEN, P. Visualization of tonal content with self-organizing maps and self-similarity matrices. *Computers in Entertainment (CIE)*, ACM, v. 3, n. 4, p. 1–10, 2005.

TURING, A. M. On computable numbers, with an application to the entscheidungsproblem. *Proceedings of the London mathematical society*, Bonhams, London, UK, GBR, v. 42, n. 2, p. 230–265, 1936.

TURING, A. M. *On computable numbers, with an application to the Entscheidungsproblem: A correction*. [S.l.]: CF Hodgson & Son, 1937.

TURING, A. M. Jugene - configuration. 1950. Disponível em: <http://www.fz-juelich.de/ias/jsc/ EN/Expertise/Supercomputers/JUGENE/Configuration/Configuration-node.html>.

UG349, X. *ML505/ML506/ML507 Reference Design, User Guide*. 2010.

VIEIRA JOãO PAULO SCHWARZ SCHüLER, K. F. M. M.-H. T. *Autômatos Celulares*. 2012. Disponível em: <http://schulers.com/jpss/estudos/CellularAutomata/clickhere.html>. Acesso em: 10/03/2012.

VLASSOPOULOS, N.; FATES, N.; BERRY, H.; GIRAU, B. An fpga design for the stochastic greenberg-hastings cellular automata. In: IEEE. *International Conference on High Performance Computing and Simulation (HPCS)*. Caen, FRA, 2010. p. 565–574.

WANDERLEY, M. M. Instrumentos musicais digitais: Gestos, sensores e interfaces. *Em Busca da Mente Musical. Curitiba: Editora da Universidade Federal do Paraná*, Editora da Universidade Federal do Paraná, Curitiba, PR, BRA, 2006.

WEINERT, W. *Computação evolucionária para indução de regras de autômatos celulares multidimensionais*. Tese (Doutorado) — Universidade Tecnológica Federal do Paraná, Curitiba, PR, BRA, 2010.

WEINERT, W. R.; BENITEZ, C.; LOPES, H. S.; LIMA, C. R. E. Simulation of the dynamic behavior of one-dimensional cellular automata using reconfigurable computing. In: *3º Reconfigurable Computing: Architectures, Tools and Applications*. Mangaratiba, RJ, BRA: Springer, 2007. p. 385–390. ISBN 978-3-540-71430-9.

WILLIS, H. *Spatial electric load forecasting*. 2. ed. USA: CRC Press, 2002. (Power Engineering (Willis) (Book 14)). ISBN 0824708407.

WISNIK, J. M. *O Som e o sentido : Uma outra história das músicas*. São Paulo, SP, BRA: Editora Schwarcz, 2007. ISBN 9788571640429.

WOLF-GLADROW, D. A. *Lattice-gas cellular automata and lattice Boltzmann models: an introduction*. USA: Springer, 2000. (Lecture Notes in Mathematics (Book 1725), 1725). ISBN 3540669736.

WOLFRAM, S. *Cellular automata and complexity: collected papers*. USA: Addison-Wesley Reading/Westview Press, 1994. ISBN 0201626640.

WOLFRAM, S. A new kind of science. Wolfram media Champaign, IL, 2002.

WOLFRAM, S. Cellular automata and complexity. {Perseus Books Group}, 2006.

WOLFRAM, S. *Wolfram Tones*. 2009. Disponível em: <http://tones.wolfram.com>.

WUENSCHE, C. A. *A Física da Música*. São José dos Campos, SP, BRA: [s.n.], 2006. Disponível em: <http://www.das.inpe.br/ alex/FisicadaMusica/fismus_indice.htm>.

XILINX. *ISE Design Suite Software Manuals v 12.1*. Disponível em: http://www.xilinx.com/support/documentation/sw_manuals/xilinx12_2/irn.pdf, 2010.

XILINX. *Foundation series software*. 2013. Disponível em: <www.xilinx.com>.

XILINX. *ISE Design Suite Software Manuals v 14.2*. Disponível em: http://www.xilinx.com/support/documentation/sw_manuals/xilinx14_5/irn.pdf, 2013.

ZUBEN, P. *Música e tecnologia: o som e seus novos instrumentos*. Rio de Janeiro, RJ, BRA: Irmãos Vitale, 2004. ISBN 978-85-7407-178-7.