Busca evolutiva de um classificador de densidade binário em autômatos celulares quaternários

Guilherme Lopes de Oliveira[1], Leandro Cipolla Contrera[1], Pedro Paulo Balbi de Oliveira[1]

[1]Faculdade de Computação e Informática – Universidade Presbiteriana Mackenzie
Rua da Consolação, 930 – 01.302-907 – São Paulo – SP – Brasil

31672280@mackenzista.com.br, 31618952@mackenzista.com.br,
pedropaulo.oliveira@mackenzie.br

Abstract. *This paper presents a way to find a rule that might satisfactorily solve the binary density classifier problem in a four-state cellular automaton. This issue is considered resolved when a rule that converts a cell automaton that contains all of its cells in the highest state in their initial configuration is found. Cellular automata are focused on using local functions to generate global changes. To perform this search optimally, genetic algorithm techniques were used, which are stochastic search and optimization methods. That seeks two solutions that have a high hit ratio and merging them, thus generating a new solution in the hope of it being better than the previous ones, and which, iteratively, can lead to a solution that satisfies the entire problem or a result that partially satisfies the problem.*

Resumo. *Este artigo apresenta uma forma de encontrar uma regra que resolva satisfatoriamente o problema do classificador de densidade binário em um autômato celular com quatro estados. Esse problema é considerado resolvido quando for encontrada uma regra que convirja um autômato celular que contenha todas as suas células no estado de maior quantidade em sua configuração inicial. Os autômatos celulares têm como foco usar funções locais para gerar mudanças globais. Para realizar essa busca de forma otimizada, foram utilizadas técnicas de algoritmos genéticos, que são métodos estocásticos de busca e de otimização, buscando duas soluções que possuem uma taxa de acerto alta e mesclando-as, gerando assim uma nova solução na esperança de ser ela melhor que as anteriores que, iterativamente, possa levar a uma solução que satisfaça o problema por inteiro ou um resultado que satisfaça o problema parcialmente.*

1. Introdução

Uma das técnicas usadas na computação evolutiva são os algoritmos genéticos. Eles são métodos estocásticos, utilizados em buscas e em otimização de soluções. Esses algoritmos evoluem soluções parciais, trocando partes de umas com as outras de acordo com a probabilidade delas resolverem o problema dado, de forma que, quanto maior a probabilidade de resolução maior é a chance dela ser usada para evoluir para uma nova

solução. Esse processo é realizado até que se ache uma dentre todas que resolva o problema por completo.

O problema abordado nesta pesquisa que deverá ser solucionado é o classificador de densidade binário. Esse é um problema que está ligado aos autômatos celulares. Eles são um tipo de sistemas que usam funções locais para atingir um resultado esperado globalmente. Essas funções são compostas de regras de transição que levam a um estado dentro do espaço amostral utilizado. Esse processo é realizado em um reticulado, no caso desse artigo um vetor unidimensional, mas podendo ser realizado em outros tipos de reticulado, como bidimensional. No caso do classificador de densidade binário, sua função no reticulado é de convergir esse vetor de tamanho ímpar para um vetor de mesmo tamanho mas com todas as células iguais ao estado de maior ocorrência no vetor inicial.

O objetivo deste artigo é estudar a técnica dos algoritmos genéticos e utilizá-los para implementar um algoritmo de busca que seja capaz de encontrar uma solução e aplicá-la no classificador de densidade binário. Mas, diferentemente de sua forma clássica, onde é usando um autômato celular com dois estados, a abordagem aqui é se tentar resolver o problema com autômatos celulares quaternários, ou seja, aqueles com 4 estados, assim aumentando o número de regras e de combinações para as transições de estado, na esperança de tornar esse problema resolvível.

2. Referencial teórico

2.1. Algoritmos genéticos

Algoritmos Genéticos são métodos estocásticos de busca e de otimização, baseados na teoria da seleção natural e na genética natural (GOLDBERG, 1989). O conceito de seleção natural é a de que os indivíduos que melhor se adaptam ao meio em que vivem, vivem mais que os demais e com uma qualidade de vida melhor, tendo assim, maior probabilidade de passar suas características para a próxima geração através de reprodução.

Essas características estão presentes nos cromossomos dos indivíduos. Esses cromossomos são passados de pai para filho através de combinações entre os pais, gerando assim, diferentes filhos com características diferentes uns dos outros, mas tendo em comum algumas características de seus pais. (DE JONG, 2006).

Em algoritmos genéticos são usados esses princípios para chegar em soluções satisfatórias. Fazendo a analogia temos que, os indivíduos são as possíveis soluções, os mais fortes ou os mais adaptados são os indivíduos que estão mais perto de ser uma solução satisfatória para o problema. Eles são iniciados com uma população aleatória e passam por três operações principais antes de passarem para uma próxima geração são eles: Mutação, seleção e reprodução (*crossover*) (DE JONG, 2006).

A mutação acontece com uma baixa probabilidade e com um indivíduo aleatório dentro de sua população inicial, como no mundo real, pode existir um indivíduo que tem uma mutação em alguma de suas características. Dentro da implementação isso não é diferente, o cromossomo pode ter um dos seus valores minimamente alterado por essa

função, que tanto pode ser positivo quanto negativo para a solução final (GOLDBERG, 1989).

Como acontece na natureza, temos a função de seleção, onde os mais adaptados têm uma probabilidade maior de passar suas características adiante. Os indivíduos são avaliados e recebem uma nota (aptidão ou *fitness*) de acordo com sua eficácia para a resolução do problema, a partir dessa nota é retirado a probabilidade dele de entrar na fila para o cruzamento. Quanto mais alta for a aptidão, maior será a probabilidade de passar suas características para a próxima geração.

Essa seleção pode ser implementada de diversas maneiras. Uma delas é a técnica da roleta, que consiste na ideia de ter uma roleta com setores do tamanho das probabilidades obtidas na hora da avaliação, onde aleatoriamente se "gira" o ponteiro e ele indica o setor que será usado para o cruzamento (*crossover*) (GOLDBERG, 1989). O torneio é outra forma de selecionar um indivíduo para o cruzamento, nele são retirados dois indivíduos aleatórios do meio da população e o de maior *fitness* tem maior chance de ser selecionado.

A função de cruzamento é a função que realiza a troca dos "genes" entre os indivíduos, criando assim novos filhos para a próxima geração. Uma das maneiras de realizar esse cruzamento é buscar dois cromossomos, cortar os dois em um determinado ponto aleatório e realizar uma troca dos pedaços um com o outro, gerando assim dois indivíduos diferentes. Essa é a base mais teórica da função, mas é possível trocar e cortar em mais de um ponto ou realizar outras operações entre os dois indivíduos para a criação desses "filhos" para uma próxima geração (DE JONG, 2006).

Após a realização de todas essas etapas, uma nova população é gerada e agora ela está pronta para realizar as mesmas operações novamente, se os requisitos de paradas não forem alcançados. Se o algoritmo estiver na última geração, ele para de gerar uma próxima geração assim podemos avaliar os resultados obtidos. Se todas as partes estiverem funcionando bem poderá ser vista uma evolução dos indivíduos, onde o *fitness* dele será por muitas vezes maior que o *fitness* do indivíduo anterior. Podem ocorrer que algumas vezes esse valor seja menor, por se tratar de um método probabilístico.

2.2. Autômatos celulares

Autômatos celulares (AC) são definidos por uma rede de células e um conjunto de regras de transições de estado. Quando as células têm o mesmo padrão de uma das regras de transição, a célula central é alterada com o estado em que a regra o leva, esse padrão é feito para todas as células do reticulado. Cada célula pode conter um dos k estados de um espaço amostral, podendo ser binário, ternário, quaternário, ou qualquer outro estado dentro do espaço proposto. A regra gera o próximo estado em que a célula deverá ir de acordo com a sua vizinhança. A vizinhança é dada pelas células que estão conectadas a ela, seu tamanho é descrito pelo raio k do autômato conforme a Figura 1 (DE OLIVEIRA, 2014).

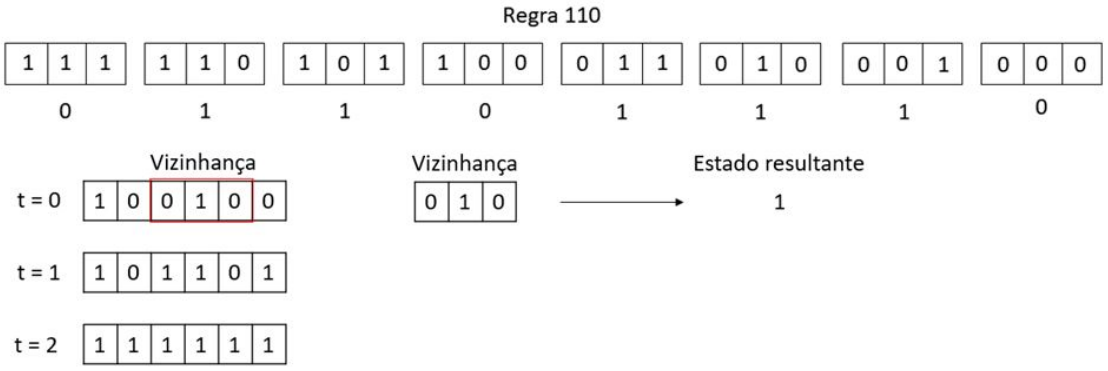

Figura 1. Representação da regra 110 do espaço amostral binário em um autômato celular de raio 1.

Normalmente, para se referir a uma regra, usamos uma representação decimal das saída das transições, em ordem lexicográfica que consiste em ordenar as transições a partir da transição que possui todas as células com o maior estado mais à esquerda e a transição com todas as células com 0 mais à direita conforme a Figura 2 (DE OLIVEIRA, 2014).

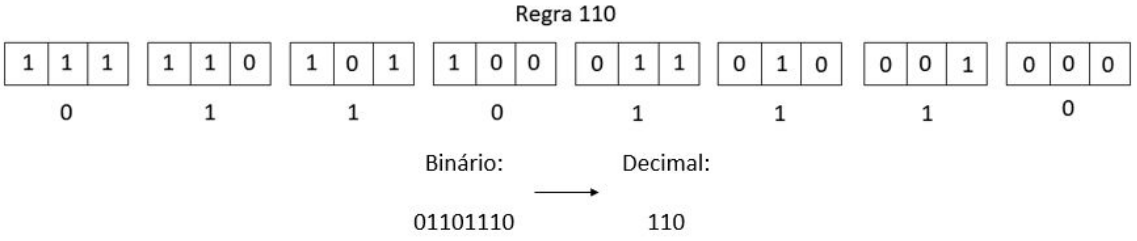

Figura 2. Representação binária da regra 110 e a representação decimal da mesma.

2.3. Classificador de densidade binário

O classificador de densidade binário é um caso de determinação da maioria global da configuração inicial. A formulação clássica e mais simples da tarefa de uma classificação de densidade estabelece que um autômato celular (AC) binário unidimensional de tamanho ímpar deve convergir uma configuração cíclica para outra, onde todas as células estarão no estado 1, quando a configuração inicial possuir mais 1s do que 0s, ou para todas as células no estado 0, caso contrário conforme a Figura 3 (DE OLIVEIRA, 2014).

Essas características foram generalizadas tornando possíveis diversas variações do mesmo problema. Uma variação estudada é a flexibilização do estado binário do classificador, podendo ele agora ser montado em k estados, significando que as regras de transição de estados podem levar a configurações intermediárias (DE OLIVEIRA, 2014). Para essa pesquisa objetivou-se um classificador com 4 estados. Esses estados adjacentes serão usados como estados intermediários, gerando mais regras e facilitando

a convergência final para o estado correto de acordo com a definição clássica do classificador.

Diferentemente dos demais sistemas computacionais, a solução para o classificador de densidade para um autômato celular não é um trabalho trivial, por conta do seu processo totalmente local não torna esse problema fácil resolução. Já foi provado que para a formatação clássica em um autômato celular de dois estados, a regra que resolva essa tarefa não existe solução, tanto para reticulados unidimensionais quanto bidimensionais. Achar essa solução em outras formatações atraiu a atenção, por ser um problema estruturalmente simples e de dar esperança te aplicar esse resultado a outros problemas baseados em autômatos celulares (DE OLIVEIRA, 2014).

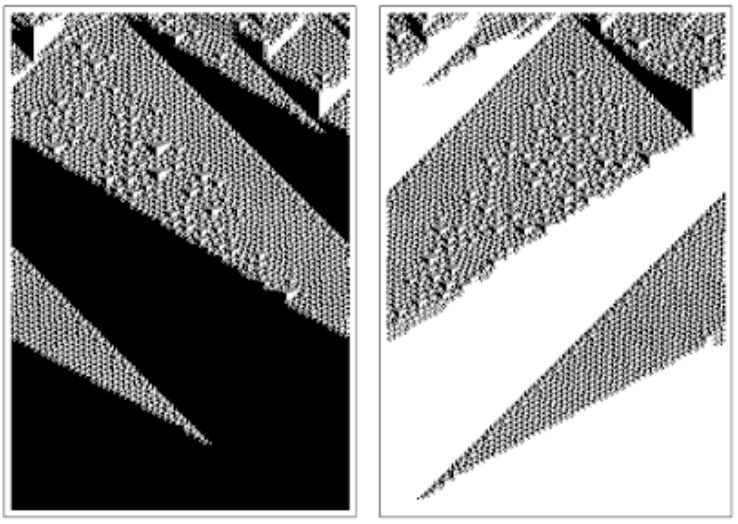

Figura 3. Exemplo de solução do classificador de densidade binário (DE OLIVEIRA, 2014).

3. Metodologia da Pesquisa

Nessa pesquisa foi utilizada a metodologia de experimentação, onde a partir dos conceitos e requisitos do problema, foi implementado um algoritmo de busca evolutiva para realizar a busca otimizada de uma solução que resolvesse o problema dado. Esse algoritmo segue o ciclo básico de um algoritmo genético detalhado e mostrado acima: mutação, seleção e reprodução (Figura 4).

Os dados são organizados em objetos dentro do algoritmo. Um objeto é o indivíduo que guarda os estados de transição da regra e a sua nota. Outro objeto dentro do algoritmo é o de configuração que guarda o vetor do reticulado da configuração e o estado para que ela deve convergir no final. Esses dados são usados durante todo o algoritmo e no final para retirar os resultados que serão avaliados.

O algoritmo inicia gerando uma população de regras de forma aleatória dentro do espaço amostral quaternário, e em seguida é gerada uma população de configurações iniciais também aleatórias, binárias e com um tamanho ímpar de células. Após essa primeira etapa, as configurações são avaliadas somando a quantidades de 1s e 0s de cada configuração, para termos o controle de saber para qual estado elas devem

convergir no final da execução se para todas as células 1s ou para todas as células 0s. Com esses dados é possível avaliar as regras nas próximas etapas do algoritmo.

Iniciando o ciclo que gera uma nova geração de indivíduos (regras), as configurações são submetidas às regras por *t* execuções (*time steps*). Em cada *time step*, para gerar a próxima linha do reticulado é retirado a célula central e seus vizinhos, esse caso 3 vizinhos de cada lado (raio 3), e esse número é traduzido para decimal, tendo assim o número da posição dentro do vetor de regras, que é ordenado em ordem lexicográfica, e essa posição corresponde ao estado para onde a célula central deve estar no próximo *time step*, e isso é repetido para todo o reticulado com contorno periódicas, assim depois de *t* execuções desse processo temos a configuração final que será usada para avaliar a regra.

Para avaliar a regra, é verificada a quantidade de células que tem o seu estado igual ao que elas deveriam convergir no final da execução. A soma de todos os acertos nas configurações é a nota da regra. Para uma regra ser considerada boa em sua população, ela deve atingir, ou estar próxima da regra alvo da população, que é dada pelo resultado da multiplicação do tamanho das configurações pela quantidade delas na execução. Essa avaliação é feita em todas as regras, cada uma com sua respectiva nota de acerto (*Fitness*).

Após a avaliação, as regras são selecionadas para a realização da reprodução (*Crossover*). O algoritmo retira dois indivíduos aleatórios da população, o de maior nota terá maior probabilidade de ser selecionado e esse processo é repetido para obter um par de indivíduos. Depois ele busca outros dois no mesmo processo, formando pares de indivíduos dentro da população. Para cada par é feita a reprodução entre eles, fazendo um corte aleatório no mesmo lugar nas duas regras. Com esse corte realizado é feita a troca das duas partes cortadas, formando assim outros dois indivíduos (regras) que serão incluídos na próxima geração.

Como última função no ciclo, já com a nova geração dada pelo cruzamento passado, é realizada a mutação. A mutação, assim como as outras funções, é feita de forma aleatória. O algoritmo seleciona 10% das regras de forma aleatória de dentro da população e modifica 10% das células dessas regras, com uma probabilidade de 50%. Com a célula selecionada, cada estado tem 25% de ser escolhido ou de se manter inalterado. Essas mínimas modificações podem trazer boas ou más consequências, aumentando ou diminuindo a nota dos indivíduos da geração em uma próxima avaliação.

Terminado o ciclo e tendo uma nova geração de indivíduos (regras), o algoritmo verifica se ele deve retornar a geração atual ou fazer o ciclo novamente. Em teoria, quanto mais gerações, mais perto da regra perfeita nós estaremos, levando em consideração que o algoritmo está fazendo o seu papel de evoluir o indivíduo em cada geração.

No final da última geração da execução, onde o algoritmo termina o ciclo de evolução dos indivíduos, é iniciado o gerador de resultados. Gerado um arquivo com os resultados da execução, a nota de acerto do melhor indivíduo em cada uma das gerações, a melhor regra da última geração e a sua nota. Esse arquivo é usado para

facilitar a análise os resultados posteriormente, não sendo necessário manter o algoritmo rodando para poder visualizar o dados.

A quantidade de regras a serem geradas e o número de gerações são definidas por parâmetros ao rodar o algoritmo, podendo ser alterados de acordo com o necessário em cada caso. A quantidade de configurações bem como o número de *time steps*, o tamanho dessas configurações e o tamanho da regra, que deve ser o número do espaço amostral utilizado, estão definidas por variáveis globais dentro do algoritmo, podendo ser modificadas de acordo com a necessidade do experimento que será realizado.

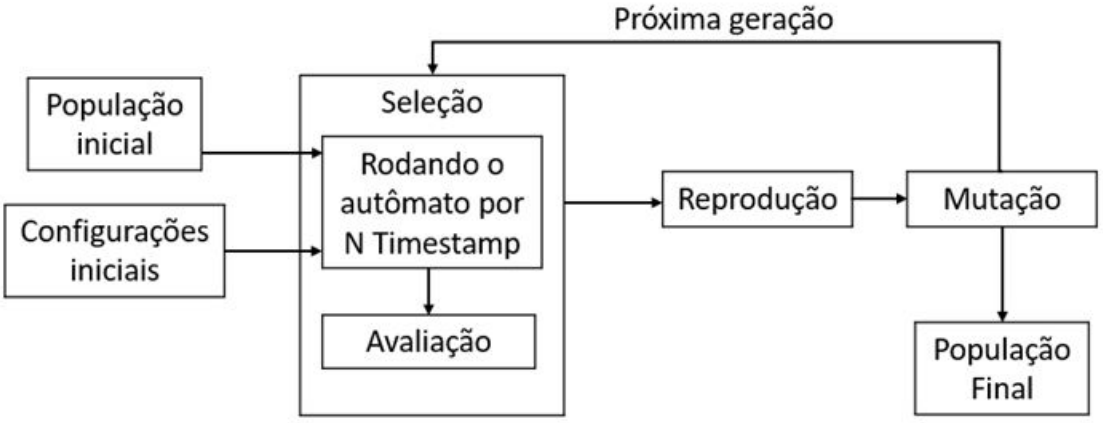

Figura 4. Fluxo do algoritmo que foi implementado para buscar a solução.

4. Resultados

Os dados que foram levados em consideração foram retirados de duas execuções do algoritmo, com os seguintes parâmetros: 50 regras, 100 configurações de tamanho 35 e trabalhados por 100 gerações. Ao final da execução com os parâmetros listados acima foram utilizados os dados retirados do melhor indivíduo da última geração como métrica para avaliar se o algoritmo foi capaz de achar uma solução que atingisse ou não o seu objetivo de resolver o classificador de densidade, de acordo com a nota de acerto (*Fitness*).

No caso dessa implementação foi tratado como nota de acerto (*Fitness*), a quantidade células que se encontram no estado correto no final da execução de cada geração. Para cada célula no estado correto é somado um a essa nota, para que a regra seja considerada boa ela deve estar perto da nota máxima da execução. A nota máxima que a solução pode alcançar é dada pela quantidade de configurações multiplicada pelo tamanho da configuração.

Com os parâmetros da execução acima temos que a quantidade de configurações é de 100 e um tamanho de 35, resultando em uma nota alvo de 3500. Essa nota alvo é alterada se o número de configurações ou o tamanho dos indivíduos forem alterados, sendo necessário refazer a conta sempre que esses dados são modificados. Se um indivíduo chegar a essa nota significa que ele acertou todas as vezes a convergência das configurações.

Analisando os dados retirados dessas execuções, é possível notar que não houve evolução dos indivíduos nas execuções com a atual implementação do algoritmo. A nota de acerto da regra não tem uma mudança significativa de uma geração para outra mostrando assim que as regras não estão ficando melhores com o passar das gerações. O indivíduo não chega a ter sua nota próxima a 50% de acerto, com as regras geradas totalmente aleatórias. Isso significa que com o passar das gerações o melhor indivíduo não é capaz de convergir as configurações para o estado correto, para que a solução resolva nem mesmo o problema parcialmente, pelo menos não nas 100 primeiras gerações que foram geradas nessas execuções. Com a população gerada com base na regra de identidade, essa nota aumenta consideravelmente mas ainda se encontra longe do esperado.

O primeiro gráfico (Figura 5) mostra a nota de acerto do melhor indivíduo com as células de sua regra geradas totalmente aleatórias, coletado no passar das 100 primeiras gerações que foram geradas pelo algoritmo, assim como o segundo (Figura 6) que foi gerado de uma segunda execução, que serviu de prova para mostrar que os dados da primeira execução estavam corretos.

Na coordenada Y desses gráficos, estão as notas do melhor indivíduo e as notas que ele deveria alcançar, foi necessário colocar apenas a metade da nota alvo para que a linha de evolução do indivíduo ficasse mais visível já que os números foram muito menores do que se esperava que seriam, mas o alvo se mantém na nota de 3500 no caso dessa execução.

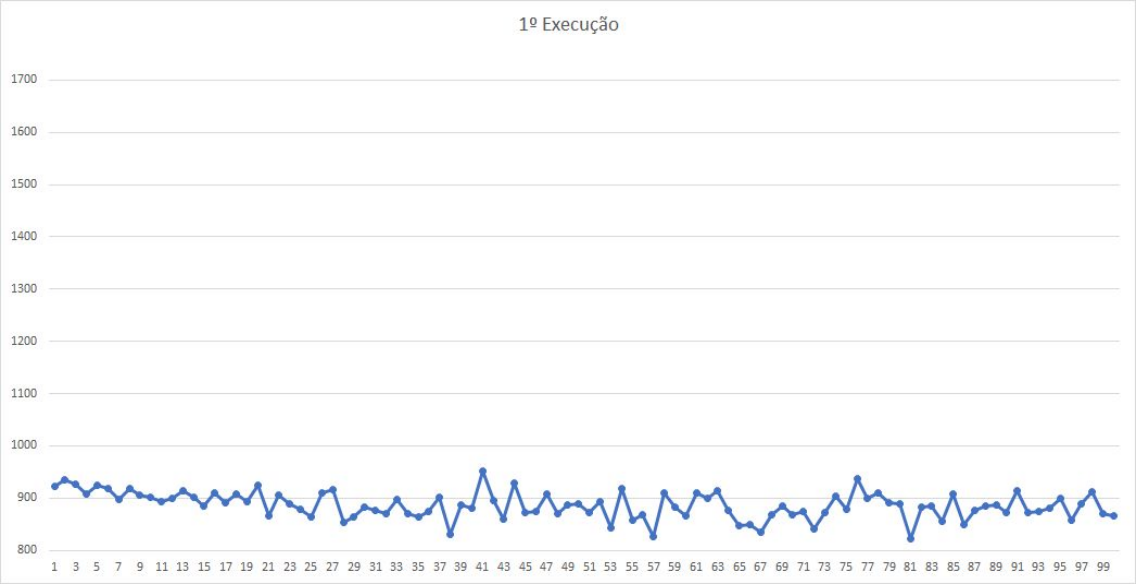

Figura 5. 1ª execução do algoritmo mostrando a nota de acerto (*Fitness*) do melhor indivíduo da geração ao passar de 100 gerações.

Figura 6. 2ª execução do algoritmo mostrando a nota de acerto (*Fitness*) do melhor indivíduo da geração ao passar de 100 gerações.

No terceiro gráfico (Figura 7) foi usado uma população gerada a partir da regra identidade. Essa regra tem como princípio manter o estado final igual a o estado inicial. Cada indivíduo foi criado com essa regra como base, mas com alterações aleatórias em algumas de suas células, sendo que cada célula tem 25% de probabilidade de ser modificada para algum outro estado e 75% de se manter no mesmo estado. Essa modificação na população foi feita na tentativa de aumentar as notas dos indivíduo, tendo assim uma melhor chance de se chegar a um resultado diferente das duas primeiras execuções onde os indivíduos tinham suas regras geradas totalmentes aleatórias. A coordenada Y deste gráfico (Figura 7) foi alterado para poder contemplar os resultados com a nova população, tendo agora, sua menor nota maior que 1600 e a maior nota em 2200.

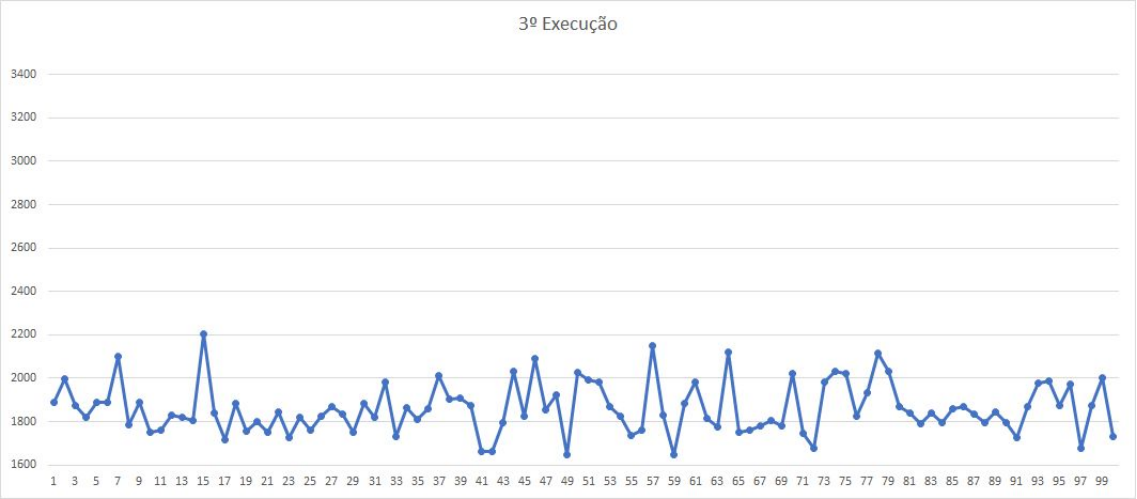

Figura 7. 3ª execução do algoritmo mostrando a nota de acerto (*Fitness*) do melhor indivíduo da geração ao passar de 100 gerações, usando uma população gerada a partir da regra de identidade.

5. Conclusão e recomendações

Analisando os dados que foram gerados no final da execução, eles mostram que o melhor indivíduo está longe de ser uma regra que resolva aceitavelmente o problema. A evolução das regras não tem acontecido como o esperado, a reprodução das regras não está levando a uma regra melhor e sim a uma regra que tem a mesma taxa de acertos que as regras que foram usadas para gerá-la. Sendo assim um resultado negativo para o atual algoritmo com relação ao objetivo proposto.

Concluímos que com a implementação atual não se pode chegar a uma solução nem mesmo parcial para o problema, sendo necessário alterar os parâmetros e as funções do algoritmo para que se possa atingir o objetivo proposto no início de resolver o classificador de densidade. Por algum motivo a implementação não está fazendo o trabalho de evoluir as soluções, assim não sendo possível se aproximar de uma solução válida para o problema. O algoritmo também deve ser rearranjado para otimizar essa busca e possivelmente melhorar esse resultado.

Uma alternativa é mudar o modo de avaliação de cada regra individualmente. Ao invés de registrar o acerto com o número de células no estado correto, pode ser usado o número de configurações que convergiram de forma correta.

A limitação computacional também é um parâmetro que deve ser considerado. As execuções são custosas para a máquina: quanto maior o número de gerações mais tempo ele leva para chegar ao final. O número de regras e de configurações também influenciam muito no tempo de execução. Usar programação paralela para paralelizar as funções que avaliam os indivíduos pode ser uma saída para esse problema, o que reduziria o tempo de execução de cada geração, permitindo que o algoritmo calcule mais gerações no mesmo espaço de tempo.

6. Referências Bibliográficas

De Jong, Kenneth A. (2006). *Evolutionary Computation A Unified Approach*. Bradford: The MIT Press.

De Oliveira, Pedro Paulo B. (2014). *On Density Determination With Cellular Automata: Results, Constructions and Directions*. Jornal of Cellular Automata. Pages 357–385.

Goldberg, David Edward. (1989). *Genetic Algorithms in Search, Optimization, and Machine Learning*. Alabama: Addison-wesley Publishing Company, page 432.