Implementação de uma solução para o Problema da Mochila utilizando Algoritmos Genéticos

Adrisson Fagner da Silva
Universidade Luterana do Brasil (ULBRA) – Campus Canoas
adris.son@hotmail.com

Resumo. Este meta-artigo descreve as atividades de pesquisa e execução de um trabalho para disciplina de Inteligência Artificial II da Universidade Luterana do Brasil (ULBRA), durante o segundo semestre de 2017. Basicamente foi implementado um algoritmo genético para solucionar um problema de otimização que foi um dos temas abordados pela disciplina.

1 Motivação

Após um grande embasamento teórico sobre Algoritmos Genéticos que foi lecionado no segundo período do semestre, popularmente conhecido como G2, surgiu-se a oportunidade de elaborar a resolução de um problema real de forma prática. Abaixo seguem alguns fundamentos para embasar e aprofundar as explicações sobre o que foi aplicado no *Knapsack Problem*.

2 Fundamentação teórica

2.1 Inteligência Artificial

É uma área de pesquisa da ciência da computação que se dedicada a buscar métodos computacionais que possuam ou simulem a capacidade racional de resolver problemas e pensar buscando, como seu próprio nome sugere, ser inteligente.

Essa área começou a se desenvolver logo após a Segunda Guerra Mundial, com a publicação do matemático inglês Alan Turing do artigo "Computing Machinery and Intelligence". Com o avanço tecnológico e o surgimento do computador, a inteligência artificial ganhou meios e massa crítica para se estabelecer como ciência integral, com problemáticas e metodologias próprias. A cada dia, novas área e vem ganhando contribuições da Inteligência Artificial. Desde os clássicos programas de xadrez ou até áreas como visão computacional, análise e síntese da voz, lógica difusa,
redes neurais artificiais e muitas outras.

Na sua fundamentação, a inteligência artificial visa reproduzir o pensamento humano além de reproduzir ideias como criatividade, auto aperfeiçoamento e uso da linguagem. De qualquer forma, esse conceito é bastante complicado de ser definido, o que permite inúmeras interpretações, muitas delas conflitantes.

2.2 Problemas de Otimização e Algoritmo Genético

Pode-se imaginar um problema de otimização como uma caixa com vários botões, onde cada botão é um parâmetro do problema. A saída é baseada no valor de uma função que indica se um determinado conjunto de botões é bom ou não para resolver este problema. Os problemas de otimização são baseados em três pontos principais: a codificação do problema, a função objetivo que se deseja maximizar ou minimizar e o espaço de soluções associado.

As técnicas de otimização tradicionais partem com um único candidato, este é manipulado de maneira iterativa coma heurísticas diretamente ligadas ao problema a ser solucionado. De forma geral os processos heurísticos não são algorítmicos, portanto pode ser bastante complexa a sua simulação. Na pratica, estes métodos são de grande importância em inúmeras aplicações de mesmo não sendo suficientemente robustos para todos os casos.

Um algoritmo genético é uma técnica de busca utilizada para achar soluções aproximadas em problemas de otimização e busca. São uma classe de algoritmos que usam técnicas inspiradas na biologia evolutiva com características de hereditariedade, mutação, seleção natural e recombinação. Os algoritmos genéticos são implementados como uma simulação, onde um conjunto abstrato de representações de solução é selecionado. A partir desse conjunto, existe uma busca de soluções ótimas. O algoritmo evolui por meio de gerações e em cada uma delas ocorre a avaliação de cada solução na população, alguns indivíduos são selecionados para a próxima geração, recombinados ou mutados para formar uma nova população. As gerações ocorrem numa estrutura semelhante a de um cromossomo. Uma de suas vantagens é a simplificação que eles permitem na formulação e solução de problemas de otimização. Normalmente trabalham com descrições de entrada formadas por cadeias de bits de tamanho fixo podendo trabalhar com bits de tamanho variável. Eles também possuem um paralelismo implícito decorrente da avaliação independente de cada uma dessas cadeias de bits. É indicado para a solução de problemas de otimização complexos, NP-Completos,
como o "caixeiro viajante". Além disso, em casos onde outras estratégias de otimização falham na busca de uma solução, os algoritmos genéticos poder convergir. Numericamente, não são sensíveis a erros de arredondamento no que se refere aos seus resultados finais.

3 Proposta

Realizado o estudo do assunto tomado como base teórica para realização das atividades, foi definido pela professora o desenvolvimento de uma aplicação prática do Problema da Mochila.

O problema da mochila, também conhecido como "knapsack problem" é um problema de otimização combinatória. O nome dá-se devido ao modelo de uma situação em que é necessário preencher uma mochila com objetos de diferentes pesos e valores. O objetivo é que se preencha a mochila com o maior valor possível, não ultrapassando o peso máximo.

Faz parte dos 21 problemas NP-completos de Richard Karp, exposto em 1972. A formulação do problema é extremamente simples, porém sua solução é mais complexa. Este problema é a base do primeiro algoritmo de chave pública (chaves assimétricas).

4 Objetivos

Tendo uma base teórica do assunto abordado e um problema a ser resolvido, será buscado, a partir das técnicas de inteligência artificial, elaborar uma solução para o problema da mochila. O objetivo é implementar uma ferramenta computacional que avalie possíveis soluções para o problema da mochila e realize as operações definidas nos algoritmos genéticos para alcançar a melhor solução de acordo com os critérios definidos.

5 Trabalho Prático

A implementação foi feita em Java utilizando-se do framework Spring Boot. A seguir são descritas as características e particularidades do programa.

5.1 Descrição

Nesta implementação da solução para o problema, foram utilizadas apenas duas métricas como base para a verificação da solução do problema da mochila: o peso e o valor do objetos.

Portanto o Algoritmo Genético se baseia nestes dois itens para calcular o preenchimento da mochila. Há duas formas de aplicação da solução, uma delas se baseia em iterar todas a gerações a fim de buscar uma solução ótima ao término de todas as gerações; a outra busca um valor que esteja dentro da taxa de aceitação levando em consideração o peso ideal, nesta solução além do valor e peso total, também é exibido quais itens fazem parte da mochila dada como ideal.

5.2 Implementação

Conforme definição previa, o algoritmo gera aleatoriamente o peso e o valor dos objetos que serão inseridos na mochila respeitando os parâmetros fornecidos pelo usuário. O crossover é do tipo two-point e a mutação é feita por troca simples. Como parâmetros de entrada há os seguintes atributos:

- ➔ População: determina o espaço de busca do algoritmo genético.
- ➔ Iterações: determina o número de gerações que o algoritmo irá gerar na busca da solução.
- ➔ Intervalo de Gerações: determina a quantidade de indivíduos serão cruzados, e assim, de novos indivíduos.
- ➔ Taxa de Mutação: determina o percentual dos indivíduos que serão mutados.
- ➔ Taxa de Aceitação: determina o percentual da solução ideal que será aceito como solução.
- ➔ Valor Ideal: determina o valor ideal da mochila (este atributo só será utilizado caso esteja marcado como “Sim” o campo “Utilizar Valor ideal?”).
- ➔ Peso Máximo: determina o peso máximo da mochila.
- ➔ Peso Máximo do Objeto: determina o peso máximo dos objetos.
- ➔ Peso Mínimo do Objeto: determina o peso mínimo dos objetos.
- ➔ Valor Máximo do Objeto: determina o valor máximo dos objetos.
- ➔ Valor Mínimo do Objeto: determina o valor mínimo dos objetos.

5.2 Resultados

Foram feitas avaliações em cima da solução com taxa de aceitação e peso ideal para que se tenha uma maior dimensão do que a alteração dos parâmetros pode causar no comportamento do algoritmo. A tela foi inicializada com valores pré-definidos, e tem uma média de 1000 interações com esses valores conforme Figura 1.

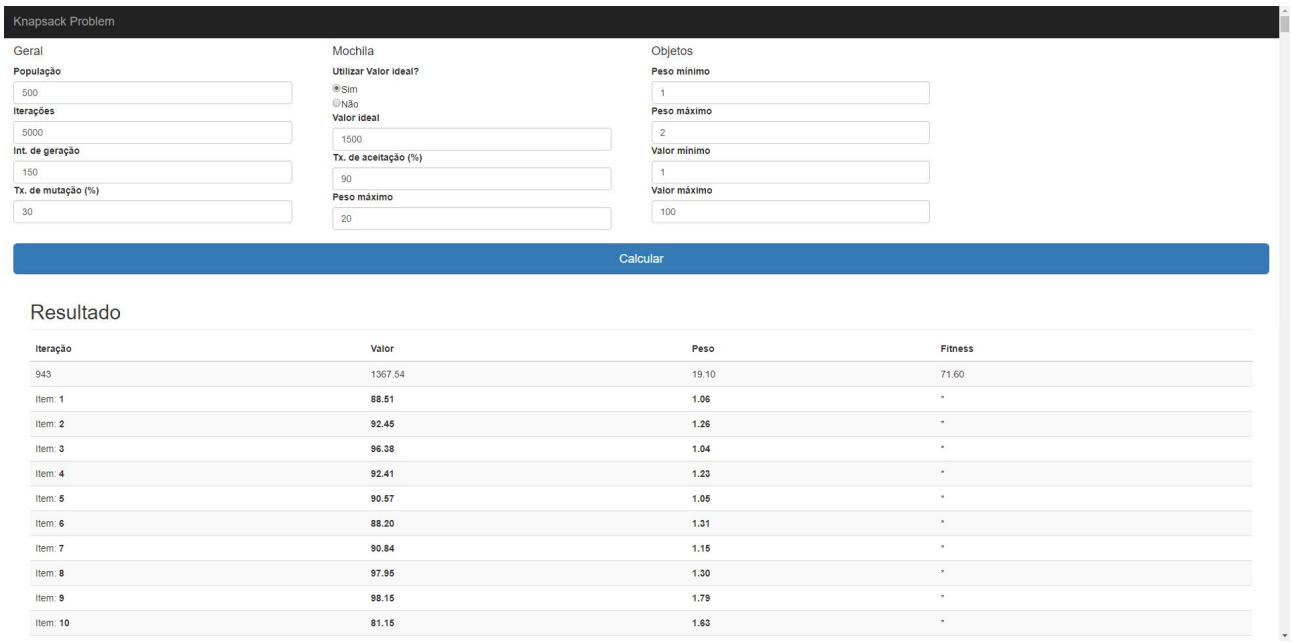

Figura 1: Parâmetros pré-definidos.

Após, foi aumentada a taxa de aceitação em 5% chegando ao valor de 95%, e com isso o algoritmo precisou percorrer muito mais gerações para achar um indivíduo aceitável, conforme Figura 2.

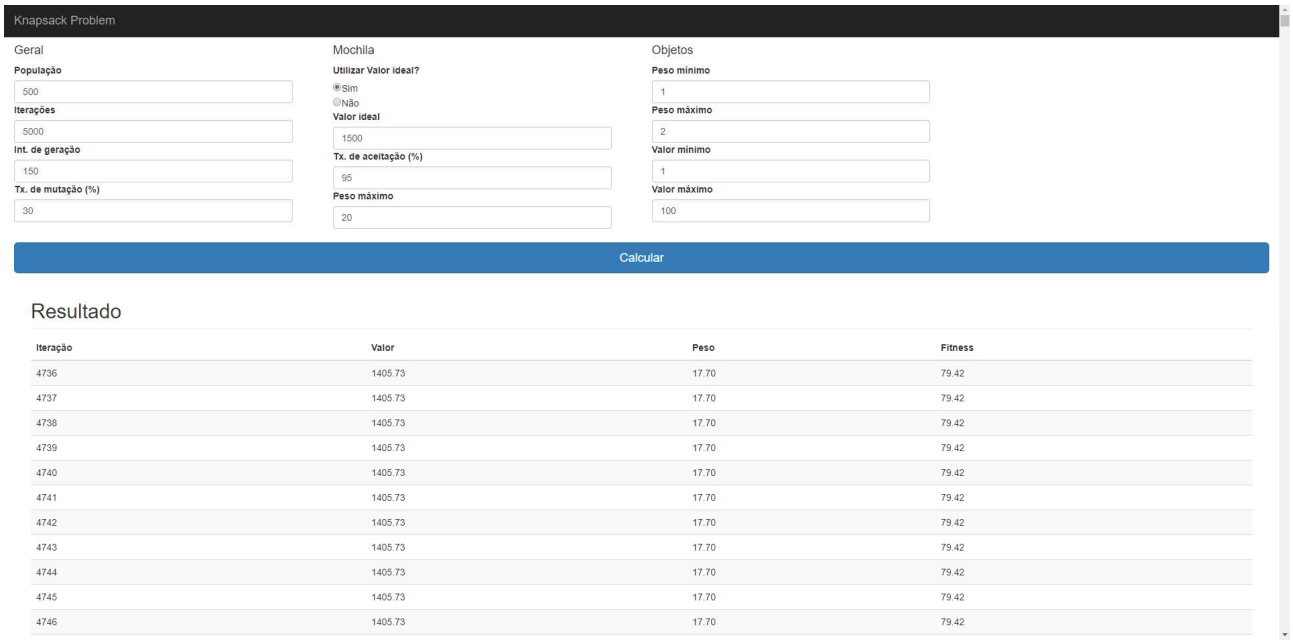

Figura 2: Taxa de aceitação em 95%.

Sendo assim, após foi diminuída a taxa de aceitação para 85%, e foi percebido uma enorme mudança nas interações, alcançando a casa das 300 interações em média, conforme Figura 3. O mais interessante nesse caso é que por questão de 5% o algoritmo aumenta sua performance em 300%.

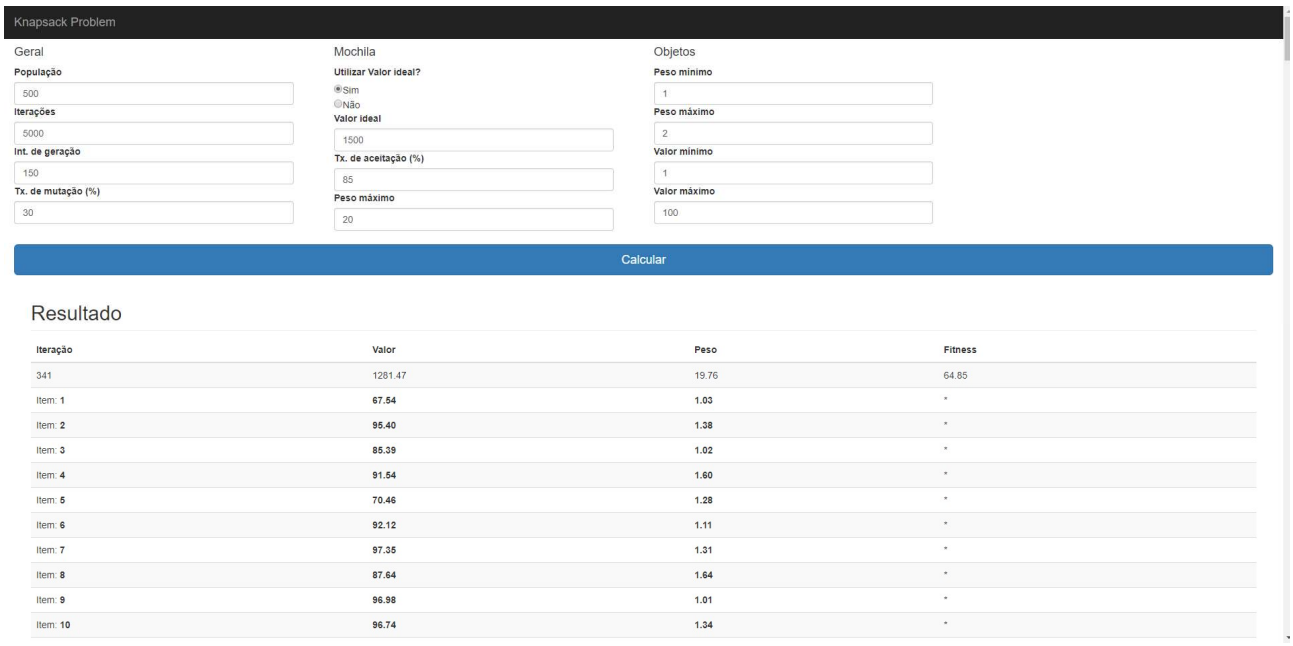

Figura 3: Taxa de aceitação em 85%.

Então foi alterado o espaço de busca do algoritmo, aumentando sua população de 500 para 800 indivíduos. Com isso o algoritmo melhorou ainda mais sua performance, chegando em 200 interações em média, conforme Figura 4.

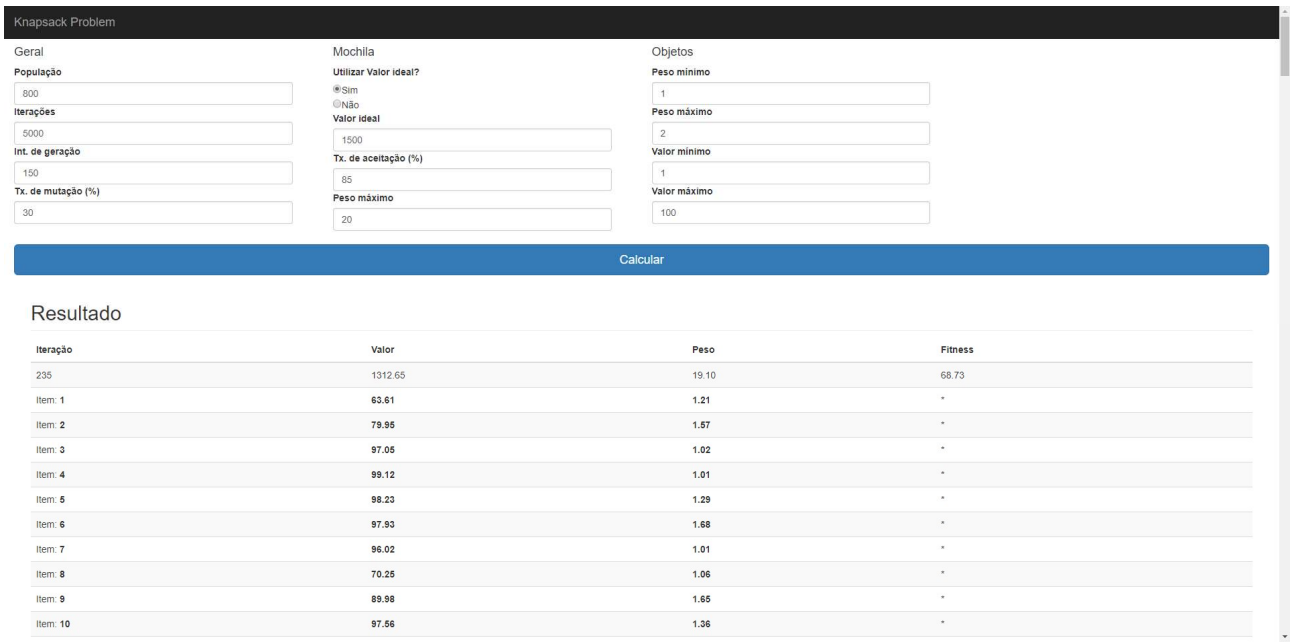

Figura 4: População de 800 indivíduos.

E como último teste documentado, foi feita a alteração da taxa de mutação, elevando o seu valor para 80%, com isso o algoritmo teve uma grande perda de desempenho, tornando-se aleatório, conforme Figura 5.

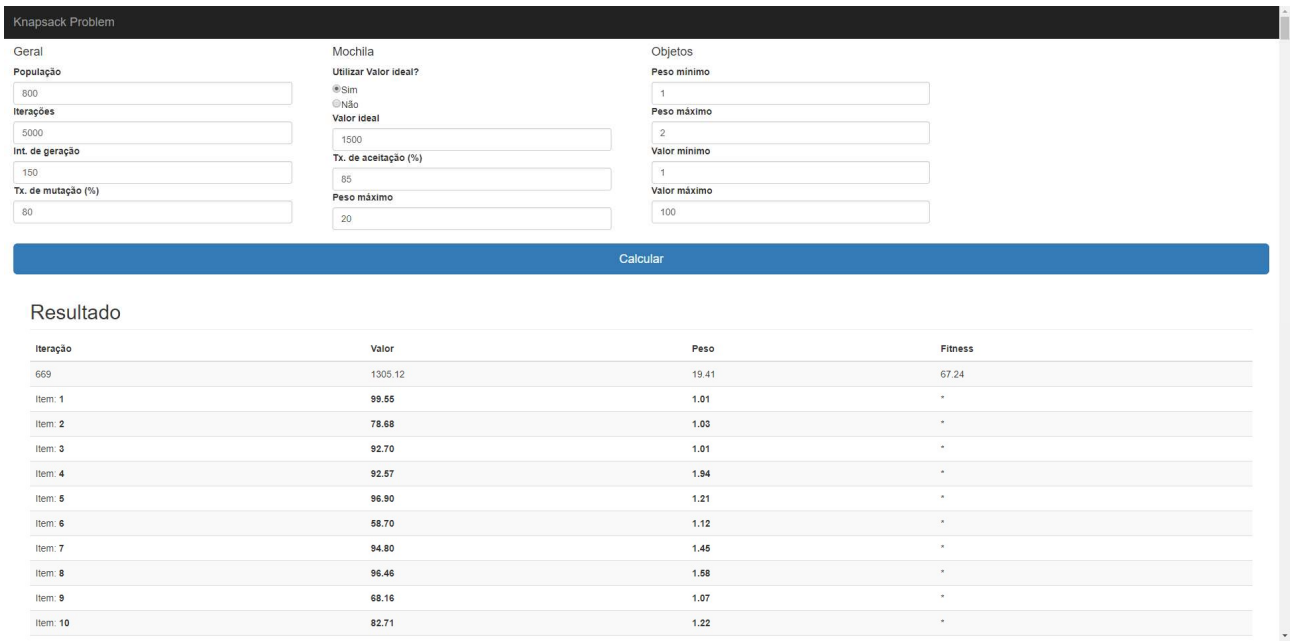

Figura 5: Taxa de mutação em 80%.

6 Conclusão

Este trabalho mostrou o funcionamento e eficiência dos Algoritmos Genéticos no problema da mochila, podendo ser feito uma análise mais minuciosa através dos resultados gerados pelos testes. Mesmo este sendo um problema menor, é viável que sejam elaborados cromossomos mais robustos em vista de obter-se resultados mais úteis em cenários reais, a mudança maior em um cenário deste tipo seria o cálculo do fator aptidão, mas o funcionamento geral do ponto de vista da IA seria exatamente o mesmo.

7 Referências

Site https://pt.wikipedia.org/wiki/Intelig%C3%AAncia_artificial, acessado em dezembro de 2017.
Site https://pt.wikipedia.org/wiki/Algoritmo_gen%C3%A9tico, acessado em dezembro de 2017.
Site http://conteudo.icmc.usp.br/pessoas/andre/research/genetic/index.htm, acessado em dezembro de 2017.
Site http://www.din.uem.br/ia/geneticos/, acessado em junho de 2017.