Identificação do Operador de Seleção do Algoritmo Genético Paralelo para o Problema de Corte Bidimensional

Mayrton D. de Queiroz[1], **Elaine C. M. Marques**[2]

[1]Centro de Informática – Universidade Federal da Paraíba (UFPB)
João Pessoa – PB – Brasil

[2]Departamento de Estatística e Informática – Universidade Federal
Rural de Pernambuco (UFRPE) Recife – PE – Brasil

mayrtondias@gmail.com, elaine.marques@ufrpe.edu.br

Abstract. *When constructing a product, we must take into account, among other factors, the best use of the raw material. With the gradual increase of the population, it becomes increasingly necessary to optimize the use of natural resources. The cutting problem is well known in the literature and can be applied in various materials such as glass, iron, wood, among others. However, in this work, we will emphasize the cut of a piece of wood, specifically. Before the This article aims to identify the best type of roulette selection operator to be integrated to the Parallel Genetic Algorithm, seeking to find the best way to cut pieces of wood. To reduce processing speed, the Genetic Algorithm was developed in a distributed way with the aid of the API Message Passing Interface (MPI), and the selection operators adopted were sequential addict roulette and addicted roulette with and without migration, in order to to verify which selection operator best meets the solution to the problem of cuts. With the experiments, it was possible to identify that the method of selection of the roulette with migration stood out when compared to the others.*

Resumo. *Ao construir um produto, deve-se atentar, dentre outros fatores, para a melhor utilização da matéria prima. O problema de corte é bastante conhecido na literatura, e pode ser aplicado em vários materiais como vidro, ferro, madeira, entre outros. No entanto, nesse trabalho daremos ênfase ao corte em peça de madeira, especificamente. Diante da problemática citada, esse artigo tem como objetivo identificar um tipo de operador de seleção da roleta a ser integrado ao Algoritmo Genético Paralelo, buscando encontrar a melhor forma de corte em peças de madeira. Para diminuir o custo do processamento, o Algoritmo Genético foi desenvolvido de forma distribuída com a auxilio da API Message Passing Interface (MPI), e os operadores de seleção adotados foram a roleta viciada sequencial e roleta viciada com e sem migração, a fim de se verificar qual operador de seleção atende de forma mais adequada a solução do problema de corte. Com os experimentos, foi possível identificar que o método de seleção da roleta com migração se destacou quando comparado aos outros.*

1. Introdução

As indústrias, em geral, se deparam com o Problema de Corte ao produzir determinados produtos, ou seja, necessitam realizar cortes em determinada peça com o intuito de retirar os itens desejados. Esse problema tem motivado pesquisadores devido as diversas

aplicações [Rangel and Figueiredo 2008]. Em [Wäscher et al. 2007], os autores apresentam uma tipologia com base em critérios como: tipo de atribuição, natureza das dimensões e sortimento de itens. Após a análise, os autores identificaram 6 grupos, entre eles o problema de posicionamento, que consiste em maximizar a quantidade de itens fracamente heterogêneos em uma determinada peça.

Em [Parmar et al. 2014], os autores realizaram um comparativo entre as principais técnicas adotadas para encontrar uma solução para o Problema de Corte, entre elas os Algoritmos Genéticos. Os Algoritmos Genéticos apresentaram bons resultados, no entanto, tiveram um custo computacional elevado quando comparado as duas melhores técnicas analisadas. Um alternativa possível é a adoção dos Algoritmos Genéticos Paralelos, a fim de que o tempo gasto para encontrar a solução seja reduzido. É válido ressaltar que ao realizar mudanças no Algoritmo Genético Sequencial para torna-lo paralelo e distribuído, um dos operadores que merece atenção é o operador de seleção. Dessa forma, o objetivo desse trabalho consiste em identificar um operador de seleção viável para o problema, dentre eles, o operador da roleta, ou seja o sequencial, e o paralelo sem migração e com migração, aplicados no problema de corte para madeira.

2. Problema de Corte

Conforme o trabalho de [Jorge et al. 2015], o problema de corte e empacotamento busca determinar um arranjo ótimo de objetos pequenos (itens) dentro de objetos maiores (recipientes), obedecendo a certas restrições e visando melhorar a qualidade da solução sob alguma perspectiva. Estes problemas são de difícil resolução, principalmente pelas restrições de não sobreposição entre os itens e posicionamentos dos itens de forma que não ultrapasse a capacidade do recipiente.

Na Figura 1 é possível observar uma representação de uma instância do problema. No lado esquerdo da figura, tem-se uma placa (recipiente) com dimensões c referente ao comprimento e l representando a largura do recipiente; no lado direito, encontram-se as peças (itens) que serão organizados de forma que se gaste o mínimo possível da placa.

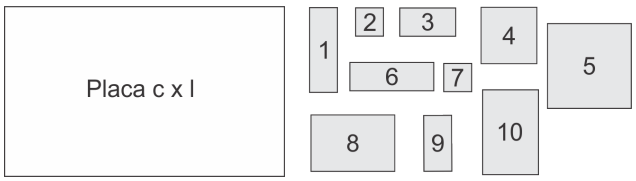
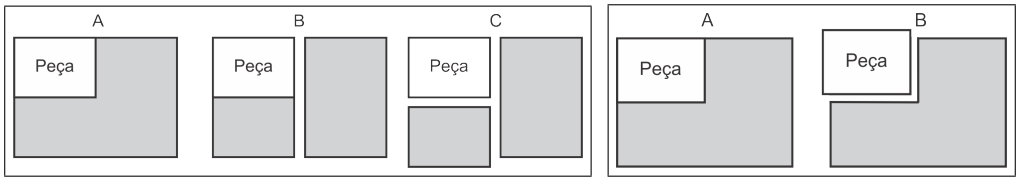


Figura 1. Representação de uma instância do problema de corte.

Ao fazer uma análise dos tipos de corte, pode-se destacar duas estratégias, sendo a primeira o corte guilhotinado, como mostrado na Figura 2a. Então, nesse tipo de corte, verifica-se a disponibilidade de espaço para encaixar a peça na placa. Realiza-se um corte verticalmente em toda placa e um corte horizontalmente em toda placa. O segundo tipo é o corte não guilhotinado, que pode ser visto na Figura 2b, esse corte não é feito por toda placa, ou seja, o tamanho do corte será conforme o tamanho da peça desejada, sendo assim ao final do corte obtém-se uma peça e um saldo, diferente do corte guilhotinado que, no exemplo anterior, resultou em uma peça e dois saldos.

Em termos práticos, é interessante destacar a importância do corte guilhotinado, visto que em marcenarias de grande porte, pode não ser adequado fazer muitas trocas



(a) Corte guilhotinado **(b)** Corte não guilhotinado

Figura 2. Tipos de corte

de medidas na serra. Deve-se organizar as peças de forma que ao fazer um corte já seja possível obter o máximo de peças, contribuindo para diminuição do gasto com a madeira, bem como energia elétrica, mão de obra, tempo de corte, entre outros. Sendo assim, nesse trabalho será adotado o corte guilhotinado.

Mesmo escolhendo o corte guilhotinado, é necessário a adoção de um padrão para inserção das peças, isto é, as peças podem ser inseridas da esquerda para direita e de baixo para cima. Na Figura 3 encontra-se uma possível solução para o problema. Aplicando o método guilhotinado, a inserção inicia no canto inferior esquerdo e cada peça é colocada uma ao lado da outra. Quando não tiver espaço na horizontal, será verificado a altura da maior peça inserida no bloco atual e a próxima peça ser inserida logo acima de todas, iniciando mais uma vez pelo lado esquerdo.

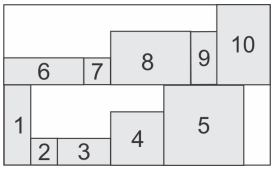


Figura 3. Detalhe de uma solução aplicando o método guilhotinado.

Diante do problema, se faz necessário escolher uma técnica para soluciona-lo. Existem diversos trabalhos na literatura sobre o Problema de Corte. Dentre eles, um alinhado a esse trabalho é o *survey* de [Parmar et al. 2014], onde os autores fazem um levantamento sobre os principais trabalhos relacionados ao Problema de Corte. Em seguida, são feitos vários comparativos entre as técnicas de solução abordadas: GRASP (*Greedy Randomized Adaptive Search Procedure*); EP (*Evolutionary Programming*); GA (*Genetic Algorithm*); ACO (*Ant Colony Optimization*); PSO (*Particle Swarm Optimization*).

Para realizar o comparativo entre as meta-heurística, os autores utilizaram duas principais métricas que são a precisão da solução e a velocidade de convergência. Após o comparativo foi classificado as três melhores dentre elas foram classificadas EP, GA e ACO. Um dos levantamentos realizado foi sobre alguns pontos positivos e negativos acerca cada técnica, onde o GA obteve bons resultados conforme as métricas avaliadas, mas um ponto negativo encontrado foi com relação do GA ter se mostrado mais lento quando comparado ao EP e ao ACO. Diante desse fato, nesse trabalho será implementado um GA Paralelo e Distribuído com uso da API MPI.

3. Algoritmo Genético

Algoritmos Genéticos (do inglês, *Genetic Algorithm* (GA)) é um ramo dos Algoritmos Evolucionários, e, como tal, podem ser definidos como uma técnica de busca baseada numa metáfora do processo biológico de evolução natural [Linden 2012]. Para solucionar um problema com GA, tem-se a necessidade de analisar os passos tradicionais do algoritmo (Conforme a Figura 4), a fim de identificar quais as adaptações necessárias para que o mesmo atenda de maneira efetiva ao problema.

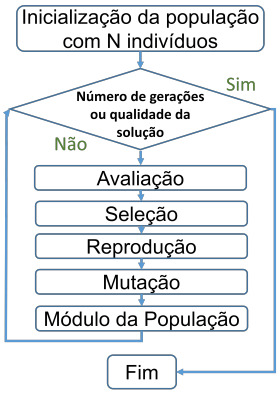


Figura 4. Fluxograma com um Algoritmo Genético Tradicional. Fonte: Baseado no Fluxograma de [Linden 2012]

3.1. Representação Cromossômica

A representação cromossômica é fundamental para o GA, sendo uma forma de traduzir a informação do problema em uma maneira viável para o tratamento pelo computador [Linden 2012]. Em outras palavras, a representação cromossômica é a forma de conectar as propriedades relevantes do mundo real ao GA, com a finalidade de aprimorar a solução.

Na Figura 5, mostra-se a representação cromossômica utilizada nesse trabalho. Um cromossomo é uma possível solução para o problema, e o mesmo é formado por vários genes, onde cada gene representa uma peça a ser inserida na placa, de tal forma é necessário armazenar informações sobre cada peça, como um identificador, o comprimento e a largura. O identificador de cada peça é útil para o operador de *crossover*, pois sem essa informação o operador pode excluir ou repetir uma peça no filho gerado.

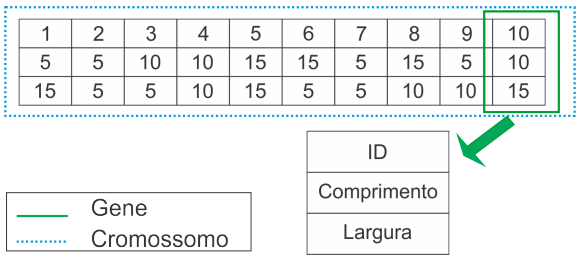


Figura 5. Representação cromossomial adotada nesse trabalho.

3.2. Escolha da população inicial

A técnica comumente usada é da criação aleatória [Linden 2012]. Por ser mais simples e em linhas gerais, ela gera uma boa distribuição e, juntamente com os operadores de mutação, devolve uma boa exploração do espaço de busca.

3.3. Função de avaliação

Conforme afirma [Linden 2012], a função de avaliação é a maneira utilizada pelos GAs para determinar a qualidade de um indivíduo como solução do problema em questão. O foco deste trabalho é em relação à busca do menor gasto ao fazer o corte das peças. Sendo assim, quanto mais material sobrar, mais as peças se encontram encaixadas. Suponha os dois indivíduos, o indivíduo 1 com a sequência de ID 1 2 3 4 5 6 7 8 9 10 e o Indivíduo 2 com a seguinte sequência de ID 1 2 3 10 5 6 7 8 9 4.

Ao fazer o posicionamento de cada peça seguindo o corte guilhotinado com o padrão da esquerda para direita e de baixo para cima, no final o valor a ser verificado é o quanto de material não foi utilizado na altura. Sendo assim, na Figura 6 é apresentado os dois indivíduos posicionados na placa. Então ao analisar a figura, é notável que o Indivíduo 1 possui avaliação 0, pois as peças chegaram na borda da placa restando um espaço 0, já no Individuo 2 sua avaliação será 5, que foi o valor não utilizado da placa. Dessa forma o indivíduo 2 terá melhor avaliação por consumir menos material.

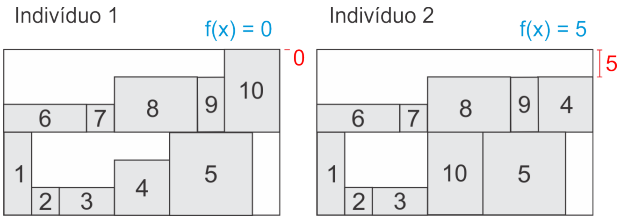


Figura 6. Processo de avaliação de dois indivíduos.

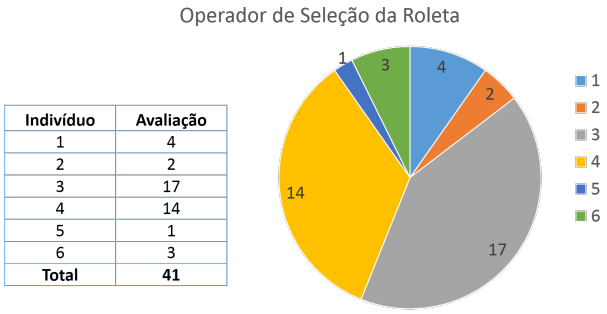
3.4. Operador de Seleção

O método de seleção de pais deve simular o mecanismo de seleção natural, a qual atua sobre as espécies biológicas, em que os pais mais capazes geram mais filhos. Ao mesmo tempo, ela permite que os pais menos aptos também gerem descendentes [Linden 2012]. O método da roleta, como mostrado na Figura 9, que possui uma tabela com seis indivíduos e suas respectivas avaliações. Nesse método é criado uma roleta (virtual) na qual cada cromossomo recebe um pedaço proporcional à sua avaliação. De posse do valor da soma total das avaliações dos indivíduos, pode-se verificar a proporção de cada indivíduo na roleta, como mostrado na Figura 9. No passo seguinte, é sorteado um valor aleatoriamente de 1 até o valor da soma total das avaliações, a fim de se verificar em qual pedaço da roleta ficou o número sorteado e assim descobrir qual o indivíduo que corresponde ao pedaço da roleta.

3.5. Operador de Crossover

Esse operador é baseado no modo como as cadeias de DNA recombinam umas com as outras na reprodução humana para combinar características de cada pai em um filho [Simon 2013]. Já [Souza 2014] complementa que nos GAs contínuos há vários modos diferentes de realizar o cruzamento. O cruzamento é dependente da forma como o cromossomo é representado.

O operador de *crossover* adotado nesse trabalho pode ser observado na Figura 8 onde é aplicado em dois cromossomos de mesmo comprimento. Inicialmente é selecionado um ponto aleatório para o cruzamento, em seguida, separa-se cada cromossomo



Indivíduo	Avaliação
1	4
2	2
3	17
4	14
5	1
6	3
Total	41

Figura 7. Tabela com os indivíduos com suas avaliações e a roleta representando os indivíduos de forma proporcional.

em duas partes, dividindo-o no ponto de cruzamento. Dessa forma, a primeira parte do primeiro pai será copiado no início do cromossomo do novo filho. O restante do cromossomo seguirá a ordem que aprece no segundo pai, é válido ressaltar que não poderá repetir peças nem faltar uma peça, o cromossomo do filho será do mesmo tamanho dos pais.

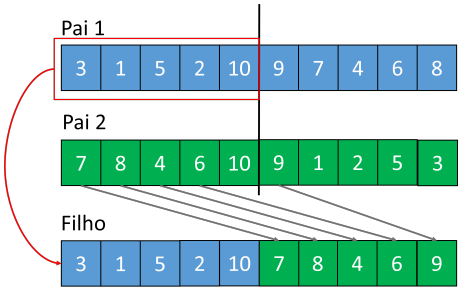


Figura 8. Detalhes sobre o processo de *crossover*.

3.6. Operador de Mutação

Operador de mutação é responsável em realizar perturbações nas soluções [Linden 2012]. Esse operador faz com que o algoritmo não caia em mínimos locais. Nesse trabalho, o operador de mutação adotado pode ser visto na Figura 9, onde ele realiza a inversão da posição da peças, ou seja, o valor da largura será o novo comprimento e o valor do comprimento será a nova largura. O gene escolhido aleatoriamente é o sexto, então, após a inversão, o Indivíduo 1' será o novo filho para compor a próxima geração.

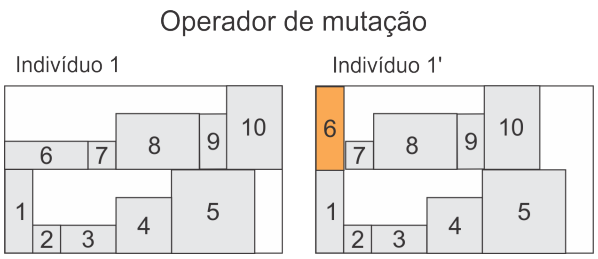


Figura 9. Detalhes sobre o processo do operador de mutação.

4. Metodologia

Após o aprofundamento no problema e das possíveis soluções existentes na literatura, foi desenvolvido um software para que o mesmo fosse capaz de encontrar boas soluções para resolver o Problema de Corte. Para minimizar o tempo computacional, ele foi desenvolvido de forma paralela e distribuída, em conjunto com a API MPI. Foram implementado três GAs, sendo um sequencial e dois paralelos, o GA sequencial é o citado nas seções anteriores e os GA paralelos serão abordados a seguir:

4.1. Algoritmo Genético com o operador de seleção sem migração

Ao analisar um GA com objetivo de torna-lo distribuído, é possível observar que o ponto mais crítico é o operador de seleção. Em um GA tradicional, o operador de seleção precisa ter conhecimento do valor da avaliação de cada indivíduo - isto é, o operador de *crossover* necessita de dois indivíduos, já o de mutação precisa apenas do filho gerado pelo *crossover*.

Diante desse problemática, o primeiro GA citado divide a população pela quantidade de processos, de forma que o operador de seleção tem conhecimento apenas de uma parte da população, a população que se encontra no processo atual. Nesse caso, não há comunicação entre os operadores de seleção. Na Figura 10 é possível identificar que ao iniciar cada processo, a população terá apenas N/P indivíduos, onde N é o tamanho da população total e P a quantidade de computadores distribuídos.

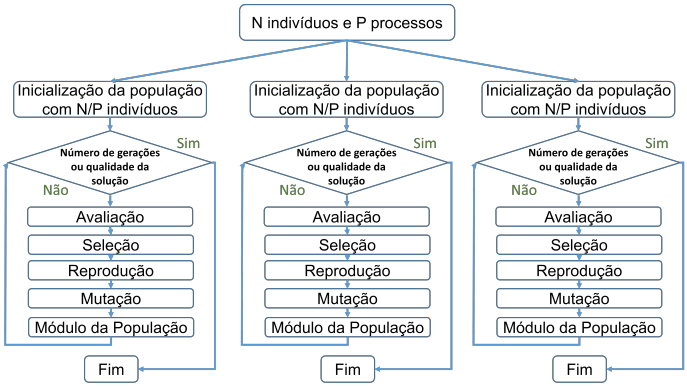


Figura 10. Fluxograma do GA com operador de seleção sem migração

4.2. Algoritmo Genético com o operador de seleção com migração

Nesse segundo GA, a população também é dividida em P partes. No entanto, o operador de seleção de cada computador se comunica, de forma que pode ser feita uma abstração observando-se como uma única população. Cada processo avalia os indivíduos da sua população, em seguida cada processo envia o somatório das avaliações dos indivíduos para todos os outros computadores conectados na rede. Quando cada processo recebe a soma dos demais, é feita a soma global com todos os indivíduos.

Com a soma da avaliação de todos os indivíduos, inclusive dos computadores conectados, cada processo pode selecionar os pais de qualquer processo. Seguindo os passos do método da roleta, então será escolhido um valor de 0 até a soma de todos os indivíduos. Em seguida, será verificado a qual população o indivíduo pertence, isto é,

será verificado qual o computador a população pertence, para que o valor escolhido seja enviado para o processo adequado.

Quando um processo P_i recebe uma mensagem de outro processo P_j, o processo P_i identificará qual o indivíduo correspondente e enviará o cromossomo para o processo P_j. Ao receber todos os indivíduos, o algoritmo segue para o passo da reprodução. Dessa forma, um processo não fica limitado apenas aos seus indivíduos (Conforme a Figura 11).

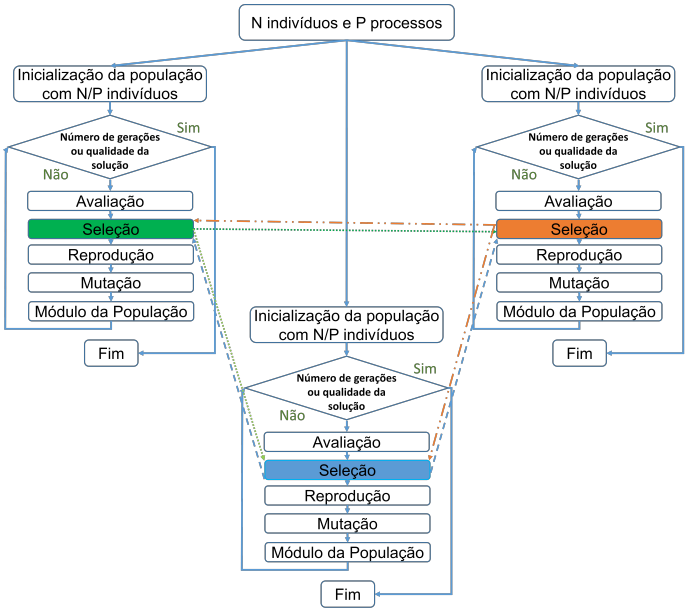
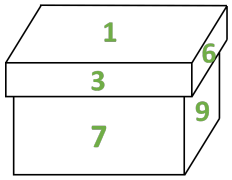


Figura 11. Fluxograma do GA com operador de seleção com migração

4.3. Experimento

Para o executar o algoritmo, é preciso de alguns parâmetros como o tamanho da população, número de gerações, a taxa de mutação e as peças a serem cortadas. Sendo assim, uma entrada com dados reais para o problema é apresentada na Figura 12, onde tem-se uma caixa com um tampa encaixável, que no total tem dimensões 20 cm de comprimento, 15 cm de largura e 11 cm de altura. No lado direito da figura, há uma tabela que corresponde as 10 peças necessárias para que seja montada a caixa. Então, para os testes realizados foram inseridos, como entrada, 10 caixas com essas dimensões, e a placa a ser utilizada para o corte tem 185 cm de comprimento por 275 cm de largura.



ID	Comprimento	Largura
1	14	19
2	15	20
3	5	20
4	5	20
5	5	15
6	5	15
7	10	19
8	10	19
9	10	14
10	10	14

Figura 12. Exemplo de entrada real para o realização dos testes.

Após a implementação dos GAs, foram realizados alguns testes a fim de se verificar o comportamento de cada algoritmo. Para cada algoritmo foram feitas variações

no tamanho da população e na quantidade de gerações. Com 25 gerações cada algoritmo foi executado 30 vezes com o tamanho da população 40, 80, 120 e 160, totalizando 120 execuções para um algoritmo com 25 gerações. No final, foram gerados 4 valores correspondendo a média de cada 30 execuções com mesmos parâmetros.

Ao concluir todos os experimentos, foram gerados quatro gráficos, sendo um parâmetro fixo que pode ser visto na Figura 13, no primeiro com 25 gerações, o segundo com 55 gerações, o terceiro com 85 gerações e o ultimo com 115 gerações. O melhor valor encontrado pelos algoritmos foi 175 cm de saldo na altura. Todos os algoritmos encontraram 175 cm, no entanto, nas várias execuções, o AG-CM (GA com migração) mostrou-se com maior frequência, em seguida o operador AG (Sequencial) e por último o AG-SM (GA sem migração), com a menor frequência.

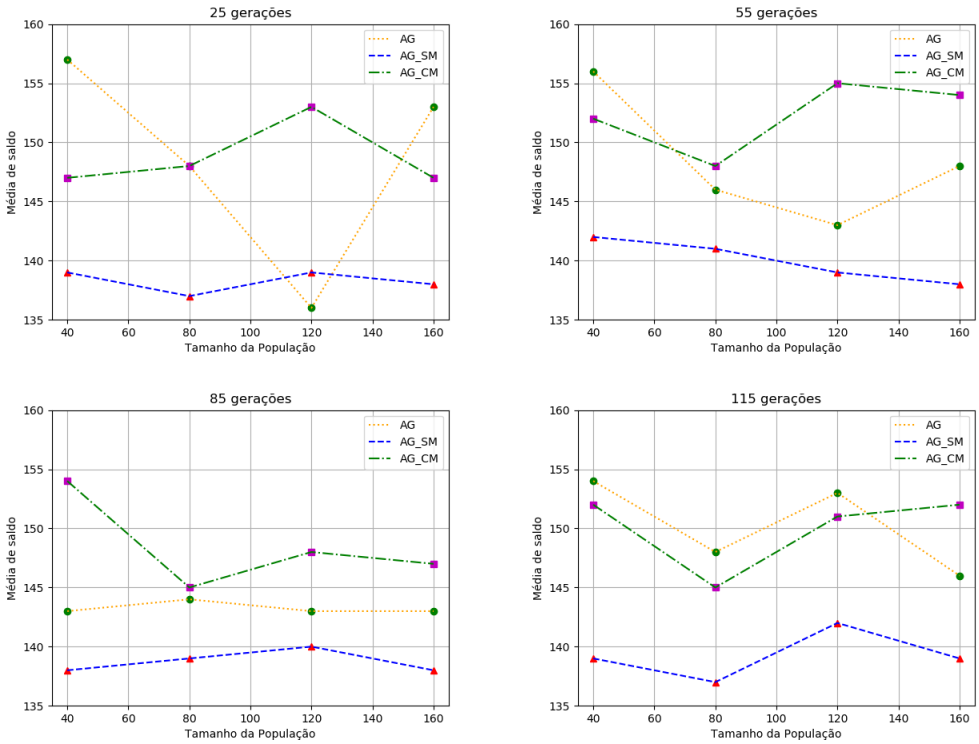


Figura 13. Resultados do experimento

Com base no gráfico, é possível notar que o AG-CM obteve melhores resultados com 55, 85 e 115 gerações. Na Figura 14, é exibido a comparação em relação ao tempo médio de execução de cada algoritmo, onde foi alterado o valor do tamanho da população e mantido os demais parâmetros. Ao analisar os gráficos, é possível observar que o operador AG Sequencial, na maioria dos casos, ficou com o desempenho médio entre os operadores paralelos, mas foi o que gastou mais tempo. Já o AG-SM não teve um bom desempenho na qualidade da solução mesmo superando os outros em relação ao tempo. Esse comportamento era esperado visto que o em todas as comparações ele estava com a população reduzida e isolada, o que dificulta o processo de percorrer o espaço de busca. Já o AG-CM apresentou um bom desempenho na qualidade da solução com um tempo menor que o AG Sequencial e maior que o AG-SM, visto que, essa diferença entre eles é referente ao custo pela transmissão das mensagens.

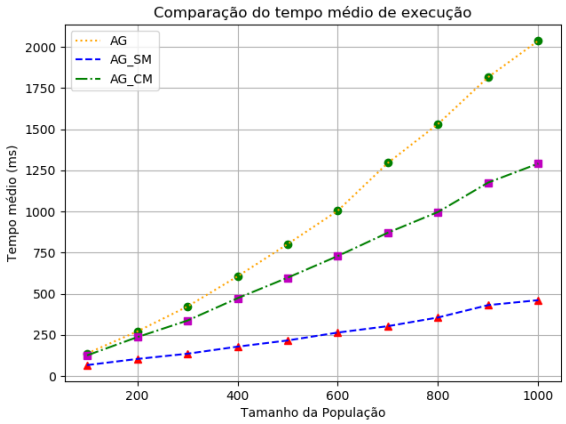


Figura 14. Comparação em relação ao tempo médio de execução dos algoritmos

5. Conclusão

Diante do trabalho realizado, foi possível identificar o bom desempenho do GA paralelo como o operador de seleção da roleta com migração. No entanto, é interessante fazer uma análise do problema para verificar se realmente se faz necessário o uso de GA paralelo. Trata-se de uma boa dica para problemas com o tamanho de população alto, pois cada processo fica com uma parte da população, sem que se perca a comunicação entre os processos. Diminui-se, dessa forma, o tempo de processamento total, ocorrendo ainda um aumento na memória devido ao alto tamanho da população.

Projetando trabalhos futuros, pode-se encontrar uma modelagem matemática que aceite peças irregulares, como o círculo, entre outras, integrando-a a um módulo gráfico que facilite para os usuários finais, como marceneiros ou até mesmo clientes que desejam verificar quanto de material será gasto na fabricação de seus produtos.

Referências

Jorge, A. R., Mundim, L. R., Cherri, L. H., and Andretta, M. (2015). O problema de corte de itens irregulares: aplicaçao na industria de aventais e forros de luva. *Simpósio Brasileiro de Pesquisa Operacional, Porto de Galinhas, Pernambuco-PE*.

Linden, R. (2012). *Algoritmos genéticos (3a ediçao)*. Brasport.

Parmar, K. B., Prajapati, H. B., and Dabhi, V. K. (2014). Cutting stock problem: A survey of evolutionary computing based solution. In *International Conference on Green Computing Communication and Electrical Engineering (ICGCCEE)*, pages 1–6. IEEE.

Rangel, S. and Figueiredo, A. G. d. (2008). O problema de corte de estoque em indústrias de móveis de pequeno e médio portes. *Pesquisa Operacional*, 28(3):451–472.

Simon, D. (2013). *Evolutionary optimization algorithms*. John Wiley & Sons.

Souza, G. P. K. d. (2014). Otimização de funções reais multidimensionais utilizando algoritmo genético contínuo.

Wäscher, G., Haußner, H., and Schumann, H. (2007). An improved typology of cutting and packing problems. *European journal of operational research*, 183(3):1109–1130.