INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGICA DE SANTA CATARINA
CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE CONSTRUÇÃO CIVIL
CURSO SUPERIOR DE ENGENHARIA CIVIL

RUAN ANDREY DE OLIVEIRA TELES

APLICAÇÃO DE FERRAMENTAS DE OTIMIZAÇÃO PARA O CORTE DE BARRAS DE AÇO PARA CONCRETO ARMADO

FLORIANÓPOLIS, ABRIL DE 2021.

Teles, Ruan Andrey de Oliveira
Otimização de cortes de barra para concreto armado/ Ruan Andrey de Oliveira Teles; Louis Augusto Gonçalves – Florianópolis, SC, 2020.
31 p.

Projeto de Trabalho de Conclusão de Curso (graduação) – Instituto Federal de Educação de Santa Catarina. Curso de Engenharia Civil.

Inclui referências

1. Otimização Combinatória. 2. Algoritmos genéticos 3. Pesquisa operacional I. Gonçalves, Louis Augusto. II. Instituto Federal de Educação de Santa Catarina. Curso Superior em Engenharia Civil III. Aplicação de Ferramentas de otimização para o corte de barras de aço para concreto armado.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGICA DE SANTA CATARINA
CÂMPUS FLORIANÓPOLIS
DEPARTAMENTO ACADÊMICO DE CONSTRUÇÃO CIVIL
CURSO SUPERIOR DE ENGENHARIA CIVIL

RUAN ANDREY DE OLIVEIRATELES

APLICAÇÃO DE FERRAMENTA DE OTIMIZAÇÃO PARA O CORTE CORTES DE BARRAS DE AÇO PARA CONCRETO ARMADO

Projeto de Trabalho de Conclusão de Curso submetido ao Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina como parte dos requisitos para obtenção do título de Engenheiro Civil.

Professor Orientador: Louis Augusto Gonçalves, Dr.

FLORIANÓPOLIS, ABRIL DE 2021.

APLICAÇÃO DE FERRAMENTA DE OTIMIZAÇÃO PARA O CORTE CORTES DE BARRAS DE AÇO PARA CONCRETO ARMADO

RUAN ANDREY DE OLIVEIRA TELES

Este trabalho foi julgado adequado para obtenção do título de Bacharelado em Engenharia Civil e aprovado na sua forma final pela banca examinadora do Curso Superior de Tecnologia em Construção de Edifícios do Instituto Federal de Educação, Ciência e Tecnologia de Santa Catarina.

Florianópolis, 27 de abril de 2021.

Banca Examinadora:

Louis Augusto Gonçalves

Louis Augusto Gonçalves, D.Sc.

Adriano Vitor, D.Sc.

Fabiano Cardos Cidral

Fabiano Cidral, D.Sc.

Priscila Maria Machado Dal Moro

Priscila Maria Machado Dal Moro, Engª Civil, Docteur.

RESUMO

No Brasil, resíduos da construção são produzidos diariamente e representam boa parte do volume de resíduos sólidos municipais. O aço utilizado para a produção de armadura entra nesse volume, sendo que a geração dessas sobras é em parte oriunda da ineficiência do processo de produção das armaduras, e podem ser mitigadas com planejamento adequado. A quantidade de cortes de aço, a partir de uma barra inicial de grande tamanho, enquadra-se num caso clássico de pesquisa operacional, chamado problema de corte e estoque (cut-stock problem), em sua variante unidimensional. A resolução exata desses problemas, em muitos casos, possui tempo computacional inviável, de forma que se faz necessário a utilização de métodos heurísticos. Neste trabalho apresenta-se um estudo do problema de corte e estoque na construção civil, trazendo resultados exatos em casos onde há viabilidade e resultados heurísticos sob uma formulação com algoritmo genético quando o tempo de execução é proibitivo. Verificou-se que os algoritmos genéticos apresentam bons resultados na resolução desses problemas e que, com adaptações, podem ser aplicados em situações práticas concretas.

Palavras-chave: Otimização Combinatória. Algoritmos genéticos. Pesquisa Operacional. Problemas de Corte e Estoque.

ABSTRACT

In Brazil, construction waste are daily produced and represents a large part of the volume of municipal solid waste. The steel used for the production of reinforced concrete contributes to this volume, and the generation of these leftovers is partly due to the inefficiency of the steel bars cutting process, which can be mitigated with proper planning. The number of steel cuts, from a large initial bar, fits in a classic case of operational research, called cut-stock problem, in its one-dimensional variant. The exact resolution of these problems, in many cases takes a long time to find, even using high-end computers, so it is necessary to use heuristic methods. This work presents a study of the problem of cutting and stock in civil construction, bringing exact results in cases where there is viability and heuristic results under a formulation with a genetic algorithm when the execution time is prohibitive. It was found that genetic algorithms have good results in solving these problems and that, with adaptations, they can be applied in concrete practical situations.

Palavras-chave: Combinatorial Optimization. Genetic Algorithms. Operational Research. Cut-Stock Problems.

AGRADECIMENTOS

A Deus pelar força e sabedoria para superar as dificuldades.

Ao IFSC, junto ao DACC pelo esforço e dedicação para manter bons cursos gratuitos ao acesso de todos.

Aos professores Adriano Vitor, Fabiano Cidral e Priscila Maria Machado Dal Moro pela cordialidade oferecida como membros da banca.

Aos meus pais e tios por sempre estarem presentes e me ajudando, independentemente da situação.

A todos meus amigos, que direta ou indiretamente, contribuíram para a conclusão deste trabalho.

Por último e mais importante, ao meu orientador Louis Augusto. Não só pela paciência para ensinar, mas pela motivação que forneceu durante a conclusão deste trabalho.

“*But I am a warrior of the sun! Spot my summon signature easily by its brilliant aura*.”

Solaire of Astora

LISTA DE TABELAS

LISTA DE FIGURAS

SUMÁRIO

1 INTRODUÇÃO

No Brasil, os resíduos na construção civil são produzidos diariamente e representam boa parte do volume de resíduos sólidos nos grandes centros. Em 2010, A construção civil do estado de Santa Catarina produziu um total de 90825,15 toneladas de resíduos sólidos, entre obras públicas e privadas (SNIS, Brasil. 2010). No município de Florianópolis, dos resíduos produzidos: 37% são concreto/argamassa, 15% são solos (areia), 12% são cerâmicos e 34% englobam os restantes dos materiais (IPEA, 2012).

O aço utilizado na construção está englobado nesses 34%, e mesmo sendo um material reciclável, contribui substancialmente para a criação de resíduos de construção. “A principal causa de geração de resíduos de aço é o corte das barras para a produção”, como afirmado por SHAHIN E SALEM (2004). Eles ainda informam que as principais causas para isso são:

- Ao dividir os cortes, o resíduo acaba sendo gerado pela existência de poucas formas de poder executar esses cortes;
- Utilização de formas ineficientes na hora do corte das barras, criando resíduos que podem ser eliminados no processo produtivo;
- Mesmo na hipótese de corte na forma mais eficiente há ainda geração de resíduo, pois não existe garantia de aproveitamento total de todas as barras.

Assim, há produção de resíduo de aço mesmo considerando as melhores práticas possíveis de produção, o que justifica o interesse num planejamento que traga em seu bojo estas melhores práticas, agregando maior eficácia ao trabalho. Existe um duplo benefício com o uso eficiente dos materiais, diminuição da quantidade de insumo necessário à construção e redução da quantidade de resíduo produzido. Ocorre, portanto, uma melhora no desempenho financeiro da empresa por duas razões, a primeira por comprar a quantidade de material próxima da mínima necessária e precisar de menos mão de obra para alcançar a produção exigida, e a segunda por reduzir despesas com reciclagem ou descarte.

1.1 JUSTIFICATIVA

Na construção civil, principal material usado para a construção de estruturas é o concreto armado, que consiste da mistura entre concreto (mistura homogênea entre cimento, agregado graúdo e agregado miúdo) e aço (armaduras), trabalhando de forma uniforme para manter a estabilidade da estrutura.

Mesmo sendo de extrema importância na execução das estruturas, o modo de corte e montagem das armaduras mais empregado pelas construtoras é de montá-las na obra. É comum utilizar-se da experiência e do bom senso dos profissionais envolvidos, muitas vezes com pessoal contratado ou terceirizado para executar essas tarefas, sem apoiar-se em cálculos ou sequer num método padronizado.

É importante salientar que um planejamento que goze de uma solução prévia para o problema de cortes agrega eficácia ao controle orçamentário da obra, permitindo verificar desvios entre a estimativa inicial de custos e os resultados efetivamente obtidos a posteriori. Trazendo-se à luz quantidade mínima de aço necessária para produzir as armaduras, gera-se a possibilidade de prever como a matéria-prima pode ser melhor utilizada e de estimar com certa segurança o quanto irá sobrar de material, tornando viáveis o controle de geração de sucata e de erros de manufatura *in loco*. Sob este prisma, torna-se bem-vinda uma formulação matemática que descreva o problema e possibilite obter uma solução eficiente, se não ótima, pelo menos próxima da ótima.

1.2 PROBLEMA DE PESQUISA

Atualmente na construção civil, via de regra, não é utilizado um modelo, ou uma planilha, para os cortes das armaduras (aço) de concreto armado. Como anteriormente mencionado, o trabalho neste campo é baseado comumente em métodos

empíricos e na experiência de trabalho do profissional responsável pela produção das armaduras.

A pesquisa consiste no estudo de métodos que consigam obter resultados satisfatórios para o problema de corte. Há uma vasta gama de aplicações conhecidas, não limitadas à somente ao corte de barras de aço para a construção civil. STADTLER (1990) utilizou-se do problema para gerar cortes eficientes de alumínio para a produção de janelas, SULIMAN (2001) cita aplicações na indústria madeireira, têxtil, de produção de vidros, couro, papel e chapas metálicas; e CHENG (1994) em vários outros casos.

Como visto, o problema goza de bastantes aplicações e é de interesse para diversos setores, sendo, portanto, um importante problema para a Engenharia em geral, de forma que merece destaque e um estudo aprofundado.

1.3 OBJETIVO GERAL

Sob a óptica teórica, analisar uma metodologia de estudo que permita implementar uma formulação matemática que modele a forma como os cortes de vergalhões pode ser feita para a produção das armaduras de aço utilizadas na construção.

Sob o ponto de vista operacional, utilizar uma metodologia que vise reduzir custos e desperdícios em obras de concreto armado, evitando que cortes para manufatura das armaduras sejam feitos sem prévio estudo.

1.4 OBJETIVOS ESPECÍFICOS

Os objetivos específicos deste projeto são:

a) Descrever o ambiente em que o problema se insere;

b) Definir uma formulação matemática que modele o problema;

c) Verificar a viabilidade de implantação do recurso computacional;

d) Implementar um algoritmo de otimização exato e heurístico;

e) Analisar e discutir os resultados.

2 REVISÃO DA LITERATURA

Visando definir conceitos, base teórica e situar o pesquisador quanto a outros trabalhos publicados na área, coletar dados e verificar os estágios em que estão os conhecimentos a respeito do item investigado faz-se necessário uma revisão bibliográfica.

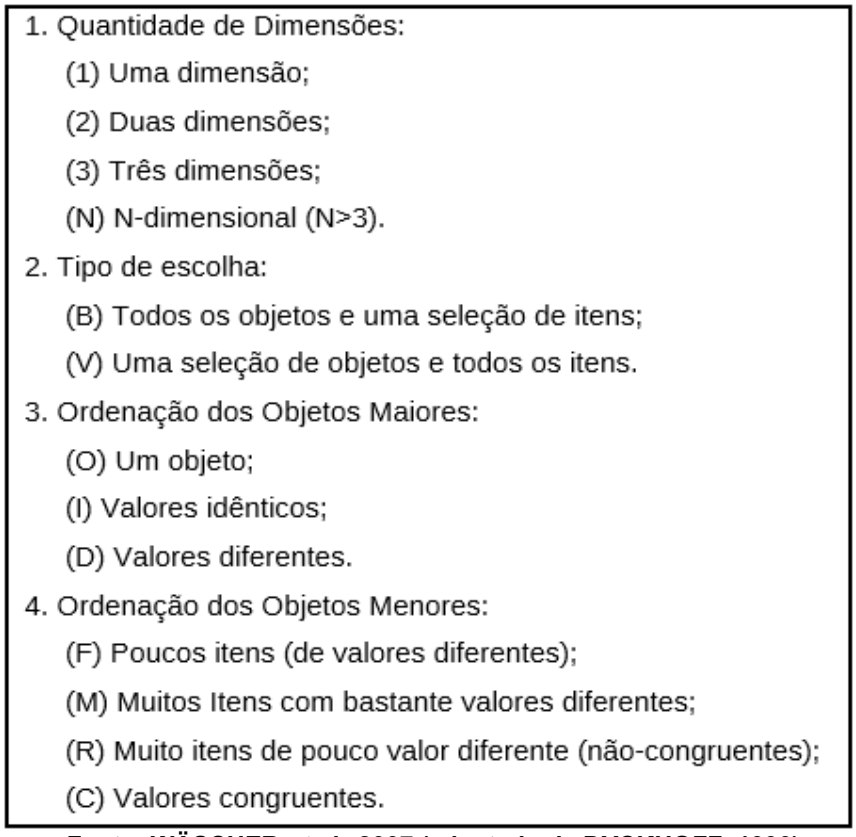
2.1 Problema de corte e estoque (*Cutting Stock Problem – C&P*)

Conforme HAESSLER (1991) a primeira formulação do problema de corte e estoque foi realizada pelo economista russo Kantorivich em 1939 e primeiramente resolvida por GILMORE (1961) utilizando programação linear com um método de inclusão de colunas determinado por um problema da mochila. A partir de então, dada a existência de computadores capazes de realizar cálculos em tempo viável, os estudos se aprofundaram até que no início dos anos 90, HAESSLER (1991) cita que haviam sido publicados mais de 500 *papers* sobre o assunto envolvendo tanto o problema diretamente como aplicações.

Problemas de corte de estoque consistem em cortar peças maiores (objetos) disponíveis em estoque com a finalidade de produzir peças menores (itens) para atender uma dada demanda, otimizando uma determinada função objetivo que pode ser, por exemplo, minimizar a perda de material, ou o custo dos objetos cortados. Os problemas de corte e estoque ocorrem de variadas formas, e suas especificações, restrições e objetivos são definidos pelas exigências práticas nos ambientes em que eles aparecem.

Com objetivo de classificar os problemas de corte e estoque presentes na literatura, DYCKHOFF (1990) desenvolveu uma tipologia de forma a abranger os problemas abordados que é mostrada na figura 1:

Figura 1: Tipologia de Classificação proposta por Dyckhoff

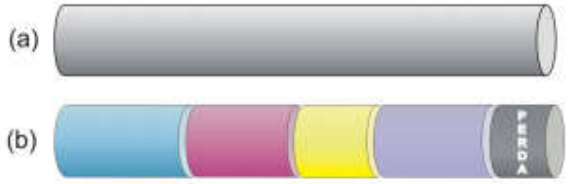


Fonte: WÄSCHER *et al.*, 2007 (adaptado de DYCKHOFF, 1990)

De forma resumida, os problemas de corte e estoque podem ser representados como sendo de uma dimensão, duas dimensões e três dimensões. A seguir é demonstrado como cada classificação funciona:

Problemas de corte unidimensional: nesse modelo, apenas uma das dimensões do objeto é considerada relevante para o processo de corte (Figura 2). Estes problemas ocorrem em processos de corte de barras de aço de mesma seção transversal, corte de bobinas de papel, tubos para produção de treliças, etc. (CHERRI, 2009).

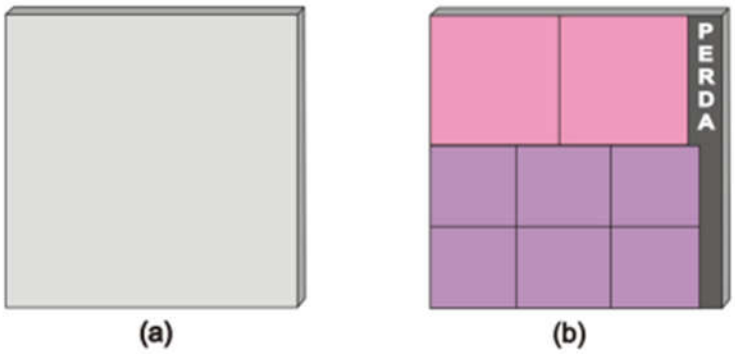
Figura 2: (a) objeto a ser cortado; (b) padrão de corte produzindo 4 itens e uma sobra



Fonte: CHERRI, 2009

Problemas de corte bidimensional: nessa problemática se enquadram as situações onde as duas dimensões do objeto são relevantes (já que a espessura do objeto já é previamente conhecida e não pode ser alterada). A figura 3 exemplificar o padrão de corte para uma chapa de material aleatório. Este tipo de problema ocorre em industrias de placas de vidro, alumínio ou madeira (CHERRI, 2009).

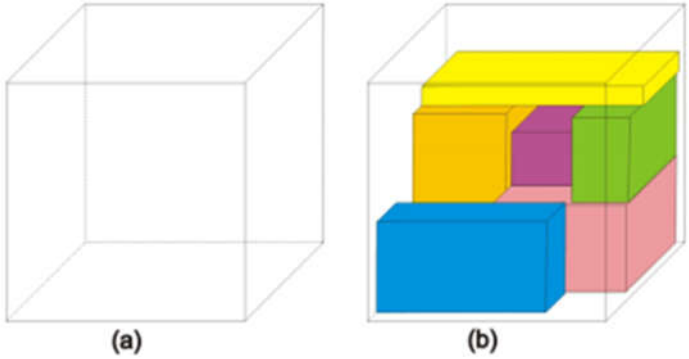
Figura 3: (a): Placa a ser cortada (b): Padrão de corte produzindo 8 itens e uma perda.



Fonte: CHERRI, 2009

Problemas de corte tridimensional: agora, as três dimensões do objeto são relevantes no processo de corte. Com essa problemática pode-se fazer uma analogia com o problema de empacotamento: cortar itens desse objeto pode ser comparado com organizar itens dentro do objeto (Figura 4). As aplicações típicas são dimensionamento de contêineres, transportes de volumes por transportadoras, etc. (CHERRI, 2009)

Figura 4: (a): Contêiner; (b): 6 caixas empacotadas no contêiner.



Fonte: CHERRI, 2009

2.2 Modelagem matemática (Método Exato)

Com a finalidade de dar uma maior compreensão do problema será apresentado o problema proposto em CHVATAL(1983), adaptado ao problema de corte de barras.

Suponha que barras sejam produzidas em uma bitola uniforme, com comprimento de 100 cm e que os pedidos possam ser feitos para cortes de 14, 31, 36 e 45 cm. A empresa recebeu as seguintes encomendas:

Figura 5: Lista de pedidos solicitados

Comprimento	Quantidade Pedida
14	211
31	395
36	610
45	97

Fonte: CHVATAL, 1983

Uma única barra de 100 cm pode ser cortada em uma ou mais das larguras do pedido. Por exemplo, uma barra pode ser cortada em duas partes de 45 cm, o que

gera um comprimento residual de 10 cm da barra, que forma sucata. De outro modo a barra poderia ser cortada em partes de 45, 31 e 14 cm, sem formação de sucata.

Cada uma dessas combinações possíveis é chamada de padrão, mas nem todos os padrões encontrados são viáveis. Considerando o exemplo anterior, uma barra de 100 cm pode ser cortada em 7 pedaços de 14 cm e ter uma sobra de 2 cm, ou 3 pedaços de 31cm com uma sobra de 7 cm, ou 2 pedaços de 36 com sobra de 28 cm ou 2 pedaços 45 cm com sobra de 10cm.

Utilizando o princípio fundamental da contagem, a quantidade de padrões cortes possíveis para o exemplo é 7x3x2x2 (7 possibilidades do corte de 14 cm, 3 possibilidades do corte de 31 cm, 2 possibilidades do corte de 36 e 2 possibilidades do corte de 45 cm) totalizando 84 padrões totais, mas apenas 37 desses padrões são viáveis para otimização como mostrado na figura 6, pois a barra não pode ser cortada em pedaços que somem mais do que 100 cm.

Figura 6: Padrões de Corte problema original

```
Ordem dos gabaritos: 14, 31, 36, 45.
Quantidades possiveis de cortes para cada gabarito: 7 3 2 2
37 padroes encontrados:
Padrao 0 (0 0 0 1 ) resto = 55
Padrao 1 (0 0 0 2 ) resto = 10
Padrao 2 (0 0 1 0 ) resto = 64
Padrao 3 (0 0 1 1 ) resto = 19
Padrao 4 (0 0 2 0 ) resto = 28
Padrao 5 (0 1 0 0 ) resto = 69
Padrao 6 (0 1 0 1 ) resto = 24
Padrao 7 (0 1 1 0 ) resto = 33
Padrao 8 (0 2 0 0 ) resto = 38
Padrao 9 (0 2 1 0 ) resto = 2
Padrao 10 (0 3 0 0 ) resto = 7
Padrao 11 (1 0 0 0 ) resto = 86
Padrao 12 (1 0 0 1 ) resto = 41
Padrao 13 (1 0 1 0 ) resto = 50
Padrao 14 (1 0 1 1 ) resto = 5
Padrao 15 (1 0 2 0 ) resto = 14
Padrao 16 (1 1 0 0 ) resto = 55
Padrao 17 (1 1 0 1 ) resto = 10
Padrao 18 (1 1 1 0 ) resto = 19
Padrao 19 (1 2 0 0 ) resto = 24
Padrao 20 (2 0 0 0 ) resto = 72
Padrao 21 (2 0 0 1 ) resto = 27
Padrao 22 (2 0 1 0 ) resto = 36
Padrao 23 (2 0 2 0 ) resto = 0
Padrao 24 (2 1 0 0 ) resto = 41
Padrao 25 (2 1 1 0 ) resto = 5
Padrao 26 (2 2 0 0 ) resto = 10
Padrao 27 (3 0 0 0 ) resto = 58
Padrao 28 (3 0 0 1 ) resto = 13
Padrao 29 (3 0 1 0 ) resto = 22
Padrao 30 (3 1 0 0 ) resto = 27
Padrao 31 (4 0 0 0 ) resto = 44
Padrao 32 (4 0 1 0 ) resto = 8
Padrao 33 (4 1 0 0 ) resto = 13
Padrao 34 (5 0 0 0 ) resto = 30
Padrao 35 (6 0 0 0 ) resto = 16
Padrao 36 (7 0 0 0 ) resto = 2
```

Fonte: Autor, 2021

Analisando a Figura 6: O padrão 21 significa que a barra de 100 cm sofreu dois cortes de 14 cm e um corte de 45 cm, resultando num aproveitamento de 73 cm e uma perda de 27 cm. Já no padrão 25 são feitos dois cortes de 14 cm, um corte de 31 cm e um corte de 36 cm.

Analisar qual a melhor forma de se executar todos esses cortes pode ser inviável manualmente. Para se ter uma reposta mais rápida é interessante transformar esse problema em um problema de otimização, modelado pela equação 1 e usando como restrições a equação 2:

Conjuntos:

I = conjunto de comprimento pedidos;

J = conjuntos das barras padrões.

Parâmetros:

a_{ij} = quantidades de barras com comprimento i cortado no padrão j;

b_i = demanda do pedido de comprimento i.

Variáveis de decisão:

x_j = quantidade de rolos cortados usando o padrão j.

O objetivo do problema de corte e estoque e minimizar o número de barras cortadas de forma a atender o pedido realizado pelo cliente. Com essa notação a formulação é assim definida:

Minimizar: $\sum_{j \in J} x_j$ (1)

Sujeito a: $\sum_{j \in J} a_{ij} x_j \geq b_j, \forall i \in I$ (2)

A equação 1 é a função objetivo do problema e visa minimizar a quantidade de barras necessárias para atender o pedido. A equação 2 garante que toda a demanda do pedido tem que ser atendida.

Na literatura podem ser encontradas alterações na função objetivo que visam ora minimizar a quantidade de barras necessárias para satisfazer a demanda, nesse sentido HAESSLER (1991), ora minimizar a quantidade de resíduo gerado, como em UMETANI (2003); ou simultaneamente reduzir a quantidade de barras e a quantidade de resíduos, aplicando neste caso uma penalidade com pesos para cada padrão escolhido, como se vê em GOULIMIS (1990), SULIMAN (2001) e CHENG (1994).

Para o problema proposto por CHVATAL (1983), a resposta encontrada neste trabalho não levou em consideração o resíduo gerado e obteve 453 barras como solução, e os padrões utilizados são mostrados na tabela 1.

Tabela 1: Solução problema original

Padrões Utilizados	Quantidade Pedida
1	49
4	100
9	198
23	106

Fonte: Autor, 2021

Para a solução encontrada em Tabela 1, temos a demanda satisfeita com poucas sobras, como se pode ver na Tabela 2. A solução foi obtida utilizando o pacote de otimização GLPK em linguagem C++.

Tabela 2: Quantidade total de cortes

Comprimento	Demanda	Produção
14	211	212
31	395	396
36	610	610
45	97	98

Fonte: Autor, 2021

A verificação da solução é feita calculando a combinação linear dos padrões selecionados ponderados pela quantidade obtida para o padrão respectivo. Para esta solução foram encontrados, com dados na Figura 6, os padrões 1 em quantidade 49, 4 em quantidade 100, 9 em quantidade 198 e 23 em quantidade 106, o que gera a seguinte situação:

(0 0 0 2).49 + (0 0 2 0).100 + (0 2 1 0) .198 + (2 0 2 0).106 = (212, 396, 610, 98)

O vetor de demanda é dado por (211, 395, 610, 97). Verifica-se assim que sobrará um corte de 14 cm, um corte de 31 cm e um corte de 97cm.

O principal entrave dessa formulação é que as colunas da matriz-solução para resolver o problema são formadas pelos padrões de corte viáveis. A simplicidade do problema abordado não apresenta problemas para a sua resolução, mas à medida que a quantidade de padrões aumenta o tamanho da matriz também aumenta, necessitando de mais recursos computacionais e de tempo para encontrar uma solução. Um exemplo real de cortes é dado na Tabela 3.

Se usássemos como exemplo a estrutura dimensionada na Tabela 3, a quantidade de padrões possíveis para a barra de Φ 5mm é de $5{,}61193*10^{46}$, impossibilitando sua resolução em tempo hábil.

Vale salientar que a obra em questão, em termos de projeto estrutural e arquitetônico, não é de grande complexidade e já é proibitiva nos termos da formulação para a otimização dos cortes. Logo, obras maiores e mais complexas terão quantidades de padrões bem maiores e necessitaram de uma grande quantidade de tempo para retornar uma solução, tornando a formulação ineficiente para a resolução de problemas mais complexos.

Caso seja necessário o uso desse método para a otimização dos cortes é interessante fazer uma limitação da quantidade de padrões possíveis para serem utilizados. Como muitos dos padrões criados acabam sendo espúrios, é útil não utilizar todos os padrões viáveis e sim priorizar “bons” padrões, isto é, padrões que sejam relevantes para o desenvolvimento eficiente do algoritmo e para obtenção de uma solução de baixo custo.

A primeira abordagem conhecida para reduzir a quantidade de padrões necessários para se obter a solução exata foi realizada por GILMORE (1961), que utilizou o problema da mochila para selecionar bons padrões sem prejudicar a qualidade da solução. Muitos outros autores buscaram métodos para o mesmo fim, como VAHRENKAMP (1995) que propõe uma pesquisa aleatória de padrões, SULIMAN (2001) que propõe um procedimento para gerar padrões para incluir na matriz de restrições, HAESSLER (1991) que propõe alterações em padrões para melhorar a solução e UMETANI (2003) que analisa a frequência de padrões na solução. Em todos os casos

o objetivo é de limitar a quantidade de padrões para tornar viável a execução do problema.

2.2.1 Considerações sobre Problemas de Otimização Combinatória

A Otimização Combinatória, resumidamente, atua em situações onde a quantidade de recursos para a execução das tarefas é limitado, criando soluções onde tais recursos sejam utilizados da melhor forma possível. Assim, sob pressão competitiva imposta pelo mercado, as empresas se esforçam para melhorar a alocação de seus recursos, permitindo redução de custos e preços praticados mais vantajosos, propiciando sua consolidação e longevidade no mercado.

Os Problemas de Otimização Combinatória (POC), em suma, visam encontrar dentro de um conjunto as soluções possíveis, a de melhor valor. A principal dificuldade nesses problemas é encontrar a solução que atenda todas as restrições necessárias simultaneamente e que resulte numa solução que seja bem avaliada. A solução que atende a todos esses requisitos é denominada ótima.

2.2.2 Métodos heurísticos

Em princípio, todo POC linear pode ser resolvido pelos métodos exatos, mas isso não significa que sua resolução seja em tempo razoável (STEFANELLO,2011). A medida que POC´s de larga escala precisaram ser resolvidos, o tempo para a obtenção de uma resposta ótima foi ficando muito grande, a ponto de ultrapassar a expectativa de vida humana. Então os pesquisadores se depararam com o seguinte dilema: encontrar uma resposta em tempo hábil, e que essa resposta forneça resultados satisfatórios.

Diante da necessidade de repostas mais rápidas, o esforço dos pesquisadores foi em aperfeiçoar estratégias aproximativas e eficientes para a resolução desses problemas (GOLDBARG, ET AL. 2016). Assim, os métodos heurísticos começam a ganhar mais espaço no meio acadêmico.

> “Heurísticas construtivas são procedimentos que constroem uma solução a partir de uma ou mais regras específicas para um dado problema de otimiza-

> ção. Os métodos construtivos, geralmente, são rápidos e os resultados obtidos através deles podem ser utilizados como ponto de partida para algoritmos de melhoria e/ou meta-heurísticas“ (ARROYO, 2002).

A intenção no uso de heurísticas para a resolução de problemas é ser um algoritmo que dará uma solução aceitável para o problema, dado uma série de parâmetros definidos pelo criador do algoritmo. Abaixo é mostrada uma lista com os principais fatores para a utilização de heurísticas na resolução de problemas:

- Quando não existe um método exato para a resolução deste problema ou o mesmo requer um tempo muito alto de processamento. Neste caso, oferecer uma solução boa é melhor do que não ter nenhuma solução;
- Quando não é necessária a solução ótima, pois as soluções obtidas já são razoáveis;
- Quando os dados são pouco confiáveis. Neste caso, a busca pela solução ótima não tem sentido, pois a mesma será uma aproximação da realidade;
- Quando limitações de tempo e/ou dinheiro obriguem a utilização de métodos de resposta rápida;
- Como passos intermediários de outros algoritmos, potencialmente exatos ou heurísticos.

Algoritmos Genéticos (AG)

Antes de entendermos o conceito de algoritmos genéticos, faz-se necessário uma explicação sobre a teoria evolutiva.

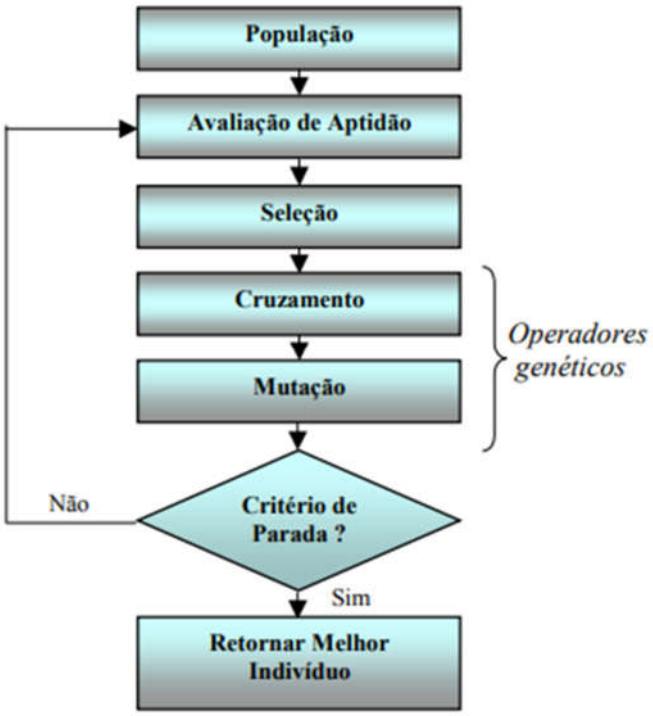
O conceito de evolução, na biologia, afirma que além da capacidade de um ser de sobreviver a um determinado ambiente ele também deva ser capaz de se reproduzir, gerando crias adaptadas ao mesmo meio, mas não igual à sua geração ascendente. Essa descendência de seres acaba “lutando” entre si para continuar sobrevivendo nesse meio, mas nem todos acabam chegando à idade adulta ou velhice. Assim os seres sobreviventes são mais “eficientemente” adaptados ao meio do que os que pereceram.

A necessidade de criar soluções mais eficientes para problemas de engenharia, fez com que John Holland com seus alunos e colegas na Universidade de Michigan, entre as décadas de 60 e 70, criassem os algoritmos genéticos. O intuito de

Holland e seus companheiros não era apenas de encontrar o melhor resultado para um dado problema, mas tentar aplicar o fenômeno biológico da evolução de forma a criar mecanismos “biológicos” que pudessem ser aplicados na computação.

O algoritmo genético, de forma simplificada, trata a solução do problema como um “indivíduo” que vai ser tornando mais “adaptável” (*fitness*) a cada iteração do algoritmo (geração). (POZO et. al., 1998). A figura 7 apresenta um fluxograma sobre o funcionamento de um algoritmo genético.

Figura 7: Funcionamento Algoritmo Genético



Fonte: POZO et. al. (1988)

- Inicialmente cria-se aleatoriamente uma população inicial de indivíduos;
- Essa população é avaliada conforme algum parâmetro, isso vai definir o quão “ideais” os indivíduos são (função de aptidão ou fitness);
- Utilizando o operador de seleção, os indivíduos mais fitness (definidos pela função de aptidão) são usados como base para novas soluções;

- Utilizando os operadores genéticos essa população tem seus genes misturados através de mutação e crossover e mutação;
- Novamente é utilizado o operador de seleção para a escolha dos indivíduos mais “aptos”;
- Esses passos são repetidos até que a solução não consiga mais ser melhorada.

População

É o conjunto de indivíduos que podem ser utilizados como solução e/ou serão utilizados para a criação de novas populações para análise. Esse é o parâmetro inicial da criação do algoritmo e possui um fator determinante na eficiência do algoritmo: populações muito pequenas podem não ter muita diversidade e acabar por encontrar uma solução “prematura” e ineficiente, por outro lado, populações muito grandes exigem muito tempo de análise e maior esforço computacional. Há de se criar um equilíbrio entre o tempo de procura da solução e o tamanho da população.

Aos indivíduos da população são definidos atributos, os quais precisam ser codificados para que o algoritmo trabalhe de forma mais precisa.

> “A codificação usando o próprio alfabeto do atributo que se quer representar (letras, códigos, números reais, etc.) para representar um indivíduo também é muito utilizada. Alguns exemplos podem ser encontrados em (Meyer 1992; Kitano 1994) ”. (POZO, et. al., 1998)

Avaliação de Aptidão

É uma função que vai calcular o quão “apto” o indivíduo é dentro da população. Esse valor norteia a seleção dos indivíduos na população, já que vai definir os aptos e inaptos a serem soluções. Essa é a componente mais importante nos algoritmos genéticos.

> “É essencial que esta função seja muito representativa e diferencie na proporção correta as más soluções das boas. Se houver pouca precisão na avaliação, uma ótima solução pode ser posta de lado durante a execução do algoritmo, além de gastar mais tempo explorando soluções pouco promissoras”. (POZO, et. al., 1998)

Seleção

Define quais indivíduos serão utilizados para a geração de novas populações através dos operadores genéticos, sua escolha é baseada na avaliação de aptidão do indivíduo. Das várias formas de seleção existentes, são muito presentes na literatura a seleção por roleta, torneio e aleatória.

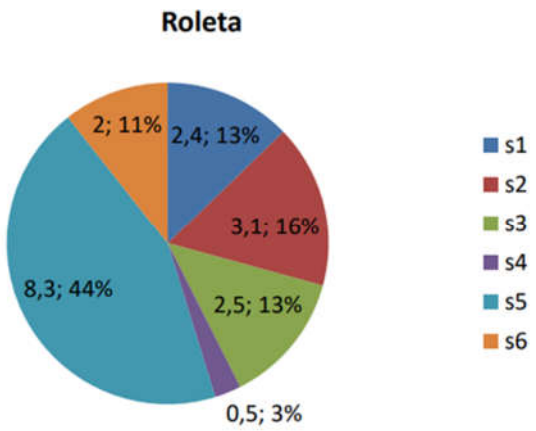
A seleção por roleta é feita através de um gráfico de setores que contém o nível de aptidão de todos os indivíduos (Figura 8). A proporção dos indivíduos no gráfico é proporcional ao nível de aptidão, assim o indivíduo S5 é o que tem maior probabilidade de ser escolhido e o indivíduo S4 a menor probabilidade.

O método do torneio seleciona os indivíduos vitoriosos de um torneiro entre todos os indivíduos da população. O indivíduo com melhor fitness é o vencedor do torneiro e ele é automaticamente inserido na população para o cruzamento, esse processo é repetido até todos os individuo vencedores do torneio preencherem a população para o cruzamento.

A seleção aleatória seleciona os indivíduos para procriação ao acaso, se usar nenhum paramento lógico.

Figura 8: Exemplo de seleção por roleta

Indivíduo	Fitness	Norm.
S1	2,4	13%
S2	3,1	16%
S3	2,5	13%
S4	0,5	3%
S5	8,3	44%
S6	2,0	11%



Fonte: TACLA, 2015

Operadores

São responsáveis por criar as gerações descendentes da população inicial, sempre mantendo as características de aptidão que seus progenitores tinham. Os

operadores: cruzamento (*crossover*) e mutação garantem diversidade de indivíduos na população e também a manutenção de características dos seus progenitores.

Cruzamento: cria novos indivíduos através da troca de material genético entre dois indivíduos, assim ele recombina os melhores genes garantindo que os indivíduos criados terão melhores características. Os tipos de cruzamento mais utilizados são: em 1 ponto ou em 2 pontos.

Cruzamento em 1 ponto: escolhe aleatoriamente um ponto de corte nos progenitores.

Cruzamento em 2 pontos ou mais pontos: um dos descendes fica com a parte central e o outro fica com as partes extremas.

Mutação: aleatoriamente substitui um valor no indivíduo por outro que seja válido. A mutação garante que a prole tenha informações que não apareceram nos seus progenitores e também oferece uma maior probabilidade de utilizar mais pontos no espaço de busca.

Algoritmos Genéticos na construção Civil

DEDE *et al* (2019), no trabalho intitulado "*Usage of optmization techniques in Civil Engineering during the last two decades"* fazem um levantamento de trabalhos mais conhecidos na área de otimização e sua aplicação na construção civil. Eles abordam em torno 20 algoritmos de otimização conhecidos que servem tanto para a otimização de objetivo único como para otimizações multi-objetivo.

O trabalho é concluído afirmando que não há um único "bom" algoritmo, cada algoritmo apresenta vantagens e desvantagens que o tornam muito eficientes numa área, mas completamente ineficientes em outras. E baseado nessas conclusões ele encoraja os pesquisadores a além de melhorar os algoritmos já existentes, criar novos métodos mais eficientes de otimização.

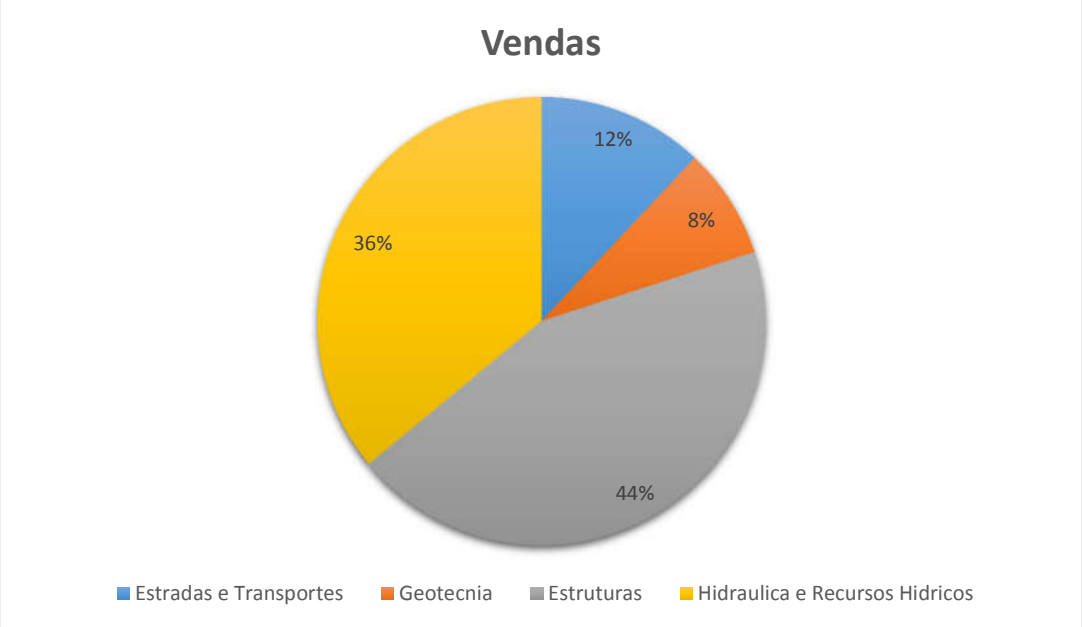
OLIVEIRA (2017) realizou uma pesquisa sobre o uso de algoritmos genéticos nas maiores áreas da construção e chegou a seguinte conclusão: A área de estruturas por resolver os problemas em regime não linear acaba criando situações de muita complexidade, assim os algoritmos genéticos acabam entrando como um bom método para obtenção de soluções interessantes aos problemas.

A segunda área que mais utiliza algoritmos genéticos é a área da hidráulica e recursos hídricos, visto que lidam com fatores naturais, suas modelagens podem se tornar extremamente complexas dadas a possibilidade dos eventos, fazendo com que soluções utilizando modelos tradicionais sejam inviáveis.

Finalmente as áreas de geotecnia e estradas e transportes são as que menos utilizam AG em suas rotinas, mesmo sendo áreas que se beneficiariam muito do uso da computação evolutiva para a formulação de soluções. A figura 9 mostra a distribuição dos trabalhos pesquisados de acordo com a área de pesquisa.

Analisando-se os trabalhos apresentados, conclui-se que os algoritmos genéticos têm uma boa aplicabilidade nos problemas de engenharia, sendo bastante eficientes e eficazes na resolução desses problemas. Além disso, vale mencionar que a evolução das pesquisas e aplicações resultam em maior economia e segurança.

Figura 9: Uso dos AG na Construção Civil



Fonte: OLIVEIRA, 2017

A seguir são citados alguns trabalhos realizados na área de engenharia civil utilizando algoritmos genéticos:

- COSTA et al (2010), faz um comparativo entre um algoritmo genético simples e um algoritmo genético híbrido para encontrar soluções para operação de um sistema de abastecimento de água.

- ALEXANDRE, L.J. Utiliza algoritmos genéticos para otimizar seções retangulares de concreto armado, de forma a atender os parâmetros da NBR 6118:2014 (Projeto de Estruturas Concreto - Procedimento). Isso faz com que a solução encontrada pelo algoritmo precise de pouca, ou nenhuma, adaptação para serem aplicados em projetos reais.

- BOTAN, et al. otimiza o traçado de rodovias utilizando algoritmos baseado em uma nova técnica para a definição do traçado da rodovia.

3 MÉTODO

3.1.1 Exemplo inicial

Para exemplificar os métodos anteriormente apresentados para a resolução do problema de corte, será usado o exemplo, já mencionado, do livro Linear Progamming, de Vasek Chvatal, Tabela 4, com uma pequena modificação que visa melhorar a visualização do método usado, inserindo mais cortes, tornando a obtenção da solução mais difícil.

Tabela 3: Problema Modificado (1ª modificação)

Comprimento	Quantidade Pedida
14	211
31	395
36	610
45	97
41	87
35	280

Fonte: Autor, 2021

3.1.2 Método exato

A solução exata é obtida utilizando-se de algum pacote de otimização. Para este trabalho escolheu-se o GNU Linear Programming Kit (GLPK), solver de código aberto, escrito em linguagem C, e que pode ser chamado via uma biblioteca em um programa escrito em C ou C++ a partir de uma série de funções próprias para inserir os dados da modelagem.

GLPK é um solver robusto que, de acordo com os seus criadores, tem o objetivo de resolver problemas lineares de larga escala, programação inteira e/ou problemas correlatos. Este solver é uma biblioteca com várias rotinas que podem ser chamadas pelo usuário para a resolução de problemas lineares.

A solução encontrada pelo GLPK é utilizando o algoritmo *branch-and bound*. Este método se baseia na construção de uma árvore com soluções baseadas em possíveis bons candidatos (*branch*), e a qualidade da solução é medida através de restrições (*bound*) colocadas pelo usuário e comparada. Essa árvore vai crescendo até que o melhor valor seja encontrado dentro do universo das soluções.

Para o exemplo da Tabela 4 foi necessário a criação de uma matriz com 70 colunas (número de padrões possíveis) que correspondem à quantidade de variáveis do problema e 6 linhas (comprimentos de corte das barras) que corresponde as restrições exigidas.

O armazenamento dos padrões foi feito através de um *array* com tamanho 6 [0 0 0 0 0 0] e cada posição dele corresponde aos comprimentos 14, 31, 36, 45, 41 e 35, da esquerda para a direita respectivamente. No Anexo 1 encontram-se todos os padrões viáveis para o problema. Analisando o padrão 39 [1 1 0 0 0 1] por exemplo, significa que a rolo padrão de 100 cm foi cortada em três partes: uma com 14 cm, outra com 31 cm e uma terceira de 35 cm, deixando um resto de 20 cm de sobra. Já o padrão 50 [2 0 1 0 0 1] gera dois cortes com 14 cm, um com 36 cm e um com 35 cm, deixando um resto de 1 cm, que se tornará sucata. O padrão 67, (4 1 0 0 0 0) corresponde a 4 cortes de 14 cm e um corte de 31 cm, deixando resto de 13 cm.

A solução exata do problema usou os padrões 2, 4, 7, 9, 14, 22, 29, e 50, nas quantidades 97,1, 1, 48, 164, 198, 43 e 84 respectivamente, totalizando 636 barras. Todas os comprimentos demandados foram atendidos, apenas o corte de 31 cm teve excedente de 1 unidade, conforme a tabela 5.

Tabela 4: Quantidade total de cortes (1ª modificação)

Comprimento	Demanda	Produção	Excedente
14	211	212	0
31	395	396	1
36	610	610	0
45	97	98	0
41	87	87	0
35	280	280	0

Fonte: Autor, 2021

Por se tratar de um problema pequeno o método exato não encontrou dificuldades para retornar uma solução. O problema apresentado na tabela 4 demandou um tempo de aproximadamente 3 centésimos de segundo para encontrar a solução usando um notebook com processador Intel I5.

Tabela 5: Segunda modificação do problema

Comprimento	Quantidade Pedida
10	352
13	92
17	69
27	218
38	521
41	213
43	324

45	229
46	236
51	104
59	1170
68	489
69	414

Fonte: Autor, 2021

O problema apresentando na Tabela 5 demorou 0,0015 segundo e no problema da tabela 6, com mais quantidade de padrões, o resultado é obtido em 0,51 segundo, resultando num total de 2520 barras.

Para avaliar a escalabilidade do problema quanto ao comprimento inicial da barra, variou-se o comprimento da barra para 110, 120, 130 cm, o que exigiu um tempo de 4,04 segundos e resultando numa necessidade de 1803 barras. Mais resultados são apresentados na Tabela 7.

Tabela 6: Tempo de solução para outros comprimentos de barra

Comprimento da barra (cm)	Tempo de execução (s)	Padrões Viáveis de Corte	Quantidade de Barras
100	0,512	568	2520
110	1,37	896	2073
120	4,08	1384	1803
130	7,84	2100	1647
140	43,7	3140	1530
150	51,3	4621	1444
160	108	6698	1342

Fonte: Autor, 2021

As soluções obtidas na Tabela 7 foram geradas utilizando um computador com processador Intel I5 com clock de 3,9 GHz e 16GB de memória RAM.

Problemas NP-Hard na prática

Problemas do tipo NP-Hard, dado sua complexidade, podem ter tempos de resolução razoavelmente imprevisíveis. Entre outros fatores, isso quer dizer que problemas semelhantes podem ter tempos de resolução completamente diferentes.

O problema de Cutting Stock é *Np-Hard* SULIMAN (2001) e MCDIARMID (1999). No caso em questão neste trabalho, padrões ineficientes podem dificultar o desenvolvimento do algoritmo, de forma que uma seleção ruim de padrões, além de não gerar uma solução de qualidade, ainda exige um tempo computacional maior do que um conjunto eficiente de padrões.

Para o problema de corte há uma grande dificuldade, pois, uma quantidade de padrões excessiva inviabiliza o desenvolvimento do algoritmo e uma quantidade pequena, porém mal escolhida de padrões, dificulta os cálculos.

A tentativa de lidar com algoritmos genéticos visa selecionar padrões que satisfaçam a exigência de se encontrar uma boa solução sem a explosão combinatória dada pela quantidade de padrões.

3.1.3 Algoritmos Genéticos

Conforme mencionado, a grande dificuldade de se obter uma solução via método exato está na quantidade de padrões gerados no problema. Ao observar soluções exatas do problema percebe-se que poucos dos padrões possíveis são utilizados, então deve existir um método que permita encontrar um pequeno número de padrões que gerem bons resultados. O algoritmo genético proposta busca trabalhar neste sentido, selecionar um número pequeno, mas eficiente, de padrões.

Diferentemente do método proposto por ONWOBULU (2003) e KHALIFA (2006) em que o cromossomo carrega em cada gene a informação sobre o padrão utilizado e em que quantidade, neste trabalho propõe-se compor cada gene do cromossomo somente com um padrão. Como exemplo, usamos o problema da Tabela 4, que possui 70 padrões viáveis, e compomos um cromossomo de tamanho 20 preenchido de forma aleatória, como mostra a Figura 10. Assim cada cromossomo será uma solução para a função objetivo.

Figura 10: Modelo Cromossomo

1	4	7	10	19	20	21	30	32	33	38	39	40	43	45	50	58	64	65	67

Fonte: Autor, 2021

Se no cromossomo da Figura 10 houver todos os padrões que gerem a solução ótima, não serão necessários os outros 50 padrões de que dispõe o problema para uso. Então esta é a racionalidade do algoritmo genético proposto, encontrar um número pequeno de padrões que gere uma solução de boa qualidade, desprezando os cortes ineficientes.

No escopo deste estudo, utilizou-se como função de *fitness* do cromossomo a quantidade de barras utilizadas, logo quanto menos barras obtidas de melhor será o cromossomo.

Assim, ao preencher o cromossomo com os padrões que geram viabilidade de solução, há um controle de quantos padrões serão usados para obter a solução ótima, podendo variar de acordo com a quantidade de cortes necessária. Se dois indivíduos (sequências de padrões de corte) possuírem bons genes, estes genes tendem a se manter na geração seguinte, e com o auxílio de mutações mantém-se alguma diversidade genética, necessária para trazer a possibilidade de algoritmo se afastar de um mínimo local da função objetivo.

Indivíduos

Os indivíduos possuem seus genes selecionados aleatoriamente dentre os padrões de corte disponíveis. A única premissa é que exista uma quantidade mínima de padrões que permita satisfazer a demanda, para isto, colocando os padrões escolhidos um sobre o outro em linha, não pode haver colunas nulas.

Assim o conjunto de padrões (28 0 0 0 0 35), (14 0 0 45 0 35), (28 31 0 0 41 0) e (0 0 36 45 0 0) formam uma quantidade mínima viável para gerar uma solução, pois todos os cortes individuais estão satisfeitos. No entanto a coleção (14 0 0 45 0 35), (28 31 0 0 0 35), (14 0 36 0 0 0), (14 0 36 0 0 35), (14 31 36 0 0 0), (14 0 0 45 0 35) e (14 0 36 45 0 0) não pode gerar solução viável porque o penúltimo item é nulo em todos os padrões, em outras palavras não haverá nenhum padrão com corte de 41 cm.

A quantidade de genes do indivíduo é arbitrado pelo operador, devendo apenas minimamente atender a todos os cortes necessários. Desta forma, neste trabalho, faz-se uma seleção aleatória de n padrões de corte (n maior do que a quantidade de cortes pedida), e por fim verifica-se se todos os cortes individuais estão satisfeitos na seleção. Em caso afirmativo é garantido que o problema possuir solução e considera-se que se trata de um indivíduo viável.

Deve-se salientar que uma quantidade pequena de genes acelera a obtenção do fitness do indivíduo, no entanto gera uma convergência lenta do algoritmo genético, pois mais indivíduos, ou mais interações entre indivíduos, são necessários.

Ao contrário uma quantidade grande de genes acelera a convergência do algoritmo, com o revés de dificultar a obtenção do valor da função fitness. Então deve existir um balanço subjetivo para que o algoritmo seja executado de forma eficiente. A quantidade de padrões não pode ser muito grande de forma a tornar o tempo de processamento do algoritmo exato proibitivo, nem pequeno demais para gerar soluções para o problema que não sejam eficientes.

Função *Fitness*

Conforme exposto anteriormente, a função *fitness* é obtida do resultado da função objetivo do problema exato, usando os padrões de corte determinados pelo indivíduo envolvido. Se o objetivo for minimizar a quantidade de barras utilizada, a função *fitness* retorna a soma de barras usadas em cada padrão de corte. Outra escolha pode ser minimizar a quantidade de sobras ou ainda um equilíbrio entre a quantidade de barras necessárias e as sobras.

População inicial

A população inicial é arbitrada pelo operador. Não pode ser um número grande demais, pois assim a quantidade de pares para geração de descendentes precisa ser muito grande tornando o tempo de execução inviável, nem pequeno demais de forma a convergir prematuramente e limitar a busca por soluções num extremante local.

Não há uma regra fixa, esses valores deveram ser arbitrados pelo operador até ser encontrado uma solução satisfatória. A população em todas as etapas do algoritmo precisa estar ordenada crescentemente pelo fitness dos indivíduos, de

forma que quando um novo elemento passa a integrar a população, substituindo outro, esse ocupa o lugar determinado pelo seu fitness, ou seja, entra na população sem desordenar os elementos.

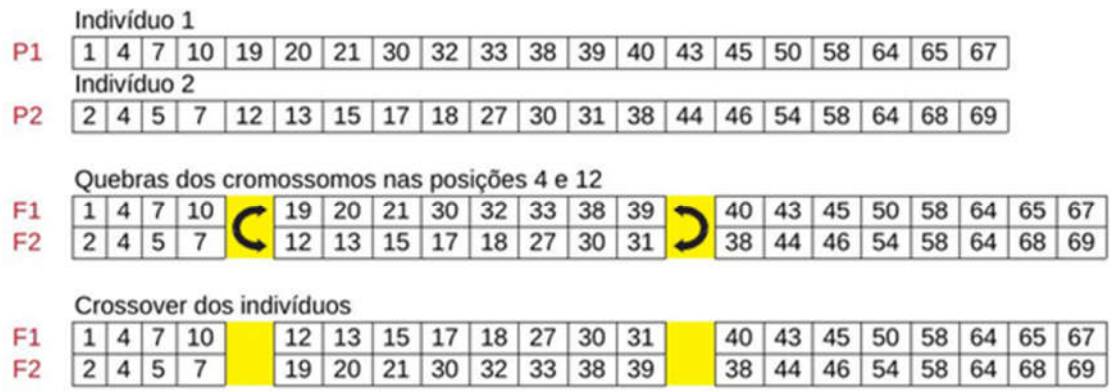
Seleção

Para a aplicação deste trabalho a seleção dos indivíduos foi feita utilizando seleção aleatória. Esta escolha se justificou porque populações pequenas já demonstraram bons resultados, então decidiu-se por não utilizar um processo de seleção para escolha de pares, para assim evitar uma pressão evolutiva desnecessária.

Crossover

O cruzamento entre os indivíduos selecionados foi através de crossover de 2 pontos que eram escolhidos aleatoriamente entre eles. A figura 11 ilustra como o método funciona.

Figura 11: *Crossover* multiponto



Fonte: Autor, 2021

Os pais são divididos na mesma posição de genes, no caso 4 e 12, gerando três alelos. Os alelos centrais são trocados nos pais e a seguir os alelos são reconectados, gerando os genes que os filhos herdam.

Por último os genes são ordenados, como mostrado na Figura 12, de forma crescente para facilitar a identificação do indivíduo e identificar genes repetidos, que devem ser eliminados.

Figura 12: Genes ordenados

Ordenação dos indivíduos

F1	1	4	7	10	12	13	15	17	18	27	30	31	40	43	45	50	58	64	65	67
F2	2	4	5	7	19	20	21	30	32	33	38	38	39	44	46	54	58	64	68	69

Fonte: Autor, 2021

Mutação

A mutação ocorre quando há genes repetidos nos descendentes depois do cruzamento dos cromossomos paternos. Neste caso ocorre a escolha aleatória de um gene (ou seja, um padrão de corte) estranho ao indivíduo, que é então inserido no cromossomo. Se não houver nenhum gene repetido não ocorre mutação.

Figura 13: Individuo mutante

Indivíduo mutante: F2. Valor aleatório de entrada: 52

2	4	5	7	19	20	21	30	32	33	38	39	44	46	52	54	58	64	68	69

Fonte: Autor, 2021

Observando as Figuras 12 e 13, verifica-se que após cruzamento o gene 38 aparece repetido no indivíduo F2, e que um destes genes repetidos é trocado pelo gene 52 (escolhido aleatoriamente entre genes disponíveis ausentes no cromossomo). Em seguida a ordem dos genes é ordenada de modo que o gene 52 ocupe uma posição entre genes vizinhos.

Atualização da População

A nova geração de indivíduos é feita considerando os quatro indivíduos à disposição (2 pais e 2 filhos), sendo que os de menores valores de *fitness* vão compor a população. Assim se os dois pais tiverem *fitness* melhores que os dois filhos gerados, os pais permanecem na população, que fica inalterada.

Se um dos filhos tiver resultado melhor do que um dos pais, o filho de melhor resultado entra na população e o pai menos adaptado sai. Se os dois filhos forem mais adaptados que os dois pais, os dois filhos entram na população e os dois pais saem.

Vale reforçar que qualquer indivíduo que fizer parte da população deve nela estar ordenado de acordo com o *fitness*, desta maneira sempre que uma inserção é feita, deve ser ordenando o novo elemento na posição determinada por seu *fitness*. Como exemplo, se um elemento criado tiver o menor *fitness* que os outros elementos da população, este entra na população na primeira posição.

Critério de parada

O critério usado foi o número de cruzamentos, que deve ser arbitrado pelo operador. Quanto maior o número de cruzamentos mais soluções com *fitness* baixo são encontradas. Na maioria das vezes, com número de cruzamentos suficientemente grande, toda a população se apresenta com o mesmo resultado de fitness, sendo este uma excelente característica do algoritmo genético para o problema, visto que o operador recebe uma gama, do tamanho da população, de opções de uso, todas possivelmente eficientes.

Todavia, o método deixa livre, a cargo do usuário, para outro critério de parada, como convergência persistente dentro da população. Neste caso, se após um certo tempo não houver mudança no valor de fitness dos primeiros termos da população, o algoritmo pode ser interrompido e a princípio se acredita estar em posse de um bom resultado.

Pode-se ainda utilizar mais de uma população e comparar os resultados, se ambas as populações estiverem com resultados coincidentes ou, pelo menos, próximos isto pode sinalizar que o algoritmo atinge seu melhor resultado.

Em suma, não há uma regra absoluta de critério de parada, qualquer método que indique que o algoritmo está estagnado serve como critério de parada.

É notório que ao aumentar a quantidade de cruzamentos o valor encontrado pelo AG se aproxima ainda mais do valor exato como mostra o exemplo da Figura 14 (mas isso obviamente aumenta o tempo de resolução do problema).

Assim o algoritmo genético é tão eficiente quanto o método exato para encontrar a solução problema, seu único contraponto é que alguns parâmetros são definidos pelo usuário e isso influencia na resposta encontrada, necessitando de vários ajustes até a resposta ideal ser encontrada.

Figura 14: Variação do resultado em função da quantidade de cruzamentos

Comprimento da Barra:		130 cm
População:		30
Quantidade de Padrões Utilizados:		50
Resultado Ótimo:		1647
Numero de Cruzamentos	Solução	Variação(%)
500	1719	4,37
1000	1667	1,21
5000	1649	0,12

Fonte: Autor, 2021

3.1.4 Problemas reais na construção civil

Um exemplo de problema real é explicitado pela Tabela 8, para uma barra de tamanho 1100 cm, disponível no mercado.

Tabela 7: Dados reais e números de padrões de corte

Comprimento	Quantidade	Comprimento	Quantidade
21	52	78	14
24	46	82	13
30	37	84	13
34	32	88	13
36	31	90	12
39	28	96	11
40	28	100	11
41	27	108	10
42	26	120	9
44	28	126	9
45	24	128	9
47	23	138	8
49	22	140	8
51	22	158	7
56	20	164	7
58	19	217	5
60	18	228	5
67	16	281	4
74	15	419	3
76	14	545	2
77	14		

Fonte: Autor, 2021

Quando o método é aplicado a um problema real a primeira coisa a ser notada é que a relação entre o comprimento da barra e os cortes solicitados é muito grande, e quando isso acontece ocorre uma explosão combinatória muito rápida do número de padrões disponíveis.

O total de padrões possíveis é proibitivo para qualquer computador, independentemente de se usar o algoritmo exato ou aproximado sem limitar a quantidade de padrões. O que se propõe é dividir o problema em subproblemas e então resolver usando barras com diferentes tamanhos, de forma a tornar o problema viável.

Para exemplificar: usando somente os cortes de 120cm a 545cm há um total de 30.481.920 padrões possíveis, inserindo o corte de 108 cm alcança-se a marca de 304.819.200 padrões possíveis, multiplicado por 13, que são os cortes demandados, chegamos total de 3.962.649.600 posições de memória, o que torna inviável o cálculo para a maioria dos computadores pessoais disponíveis. Inserindo cortes

menores esta explosão combinatória do número de padrões possíveis inviabiliza o cálculo para qualquer computador disponível no mercado. O que se propõe, portanto é resolver o problema para cortes grandes usando o algoritmo, e resolver manualmente a quantidade de barras para cortes pequenos.

No exemplo explicitado pela Tabela 8, considerando os cortes e demandas da Tabela 8, precisamos de apenas de 10 barras para satisfazer a demanda. As barras menores têm suas quantidades satisfeitas manualmente.

Tabela 8: Demanda de cortes exemplo real

Comprimento	Quantidade Pedida
120	42
126	2
128	2
138	2
140	2
158	2
164	2
217	2
228	2
281	2
419	2
545	2

Fonte: Autor, 2021

Para este caso foram encontrados 12951 padrões viáveis na solução encontrada e os padrões utilizados como solução são mostrados a seguir:

Padrão 1 (0 0 0 0 0 0 1 0 1 1 1 0) resto = 8.

Padrão 2 (0 0 0 0 0 1 1 0 1 0 0 1) resto = 5.

Padrão 3 (1 0 0 0 0 0 0 2 0 0 0 1) resto = 1.

Padrão 4 (2 0 0 0 0 1 0 0 0 1 1 0) resto = 2.

Padrão 5 (6 2 1 0 0 0 0 0 0 0 0 0) resto = 0.

Padrão 6 (7 1 1 0 0 0 0 0 0 0 0 0) resto = 6.

Padrão 7 (8 0 0 0 1 0 0 0 0 0 0 0) resto = 0.

Padrão 8 (8 0 0 1 0 0 0 0 0 0 0 0) resto = 2.

O resultado encontrado exigiu 10 barras, 6 barras utilizando os padrões 1, 2, 3, 4, 5, 6, 2 barras utilizando o padrão 7 e 2 barras utilizando o padrão 8

A solução manual do problema para cortes pequenos não é complexa e é mostrada na Tabela 9. Essa solução teve a seguinte formulação: inicialmente foi utilizada uma barra de 1100 cm para atender toda a demanda do corte de 21cm, e teve uma sobra de 428 cm.

Tabela 9: Solução manual

Comprimento (cm)	Quantidade	Comprimento Total	Quantidade máxima de cortes na barra	Quantidade barras	Sobra
21	32	672	52	1	428
24	231	5544	45	5	384
30	10	300	36	0	84
34	10	340	32	1	760
36	40	1440	30	1	420
39	40	1560	28	2	1060
40	24	960	27	0	100
41	31	1271	26	2	1029
42	24	1008	26	0	21
44	24	1056	25	1	65
45	63	2835	24	3	530
47	24	1128	23	1	502
49	37	1813	22	2	889
49	116	5684	22	5	705
51	12	612	21	0	93
56	48	2688	19	3	705
58	18	1044	18	1	761
60	18	1080	18	1	781
67	2	134	16	0	647
74	43	3182	14	3	765
76	89	6764	14	6	601
76	2	152	14	0	449
76	13	988	14	1	561
77	2	154	14	0	407
78	46	3588	14	3	119
78	113	8814	14	8	105

82	33	2706	13	3	699
84	33	2772	13	2	127
88	2	176	12	1	1051
88	16	1408	12	1	743
90	726	65340	12	59	303
96	356	34176	11	31	227
100	10	1000	11	1	327
100	88	8800	11	8	327
108	8	864	10	1	563

Fonte: Autor, 2021

A sobra de 428 cm foi subtraída do comprimento total demandado do corte de 24 cm, ficando este com um comprimento total a ser atendido de 5116 cm que é atendido utilizando 5 barras e tendo uma sobra de 384 cm, que é subtraído do próximo corte na tabela e verificado a quantidade de barras necessária para atender o novo comprimento total.

Esse procedimento é feito até toda a demanda ser atendida, sempre seguindo a ordem dos cortes como feita na tabela. Assim tivemos um total de 157 barras, com um comprimento total de sobras de 17338 cm.

4 CONCLUSÃO

Apesar de o método exato apresentar boa escalabilidade, para problemas grandes o algoritmo genético se mostra uma alternativa viável para lidar com a explosão combinatória dos padrões gerados. Para os problemas estudados, a limitação do método exato não está no tempo de execução do método simplex nem da obtenção da solução inteira do problema posterior, mas na seleção da quantidade de padrões gerados.

Para o gestor de uma obra, todavia, interessa o mínimo custo para a execução do trabalho. O uso dos algoritmos genéticos é conveniente, pois cede várias alternativas de baixo custo, e dentre estas pode-se escolher a que possui menor tempo de processamento na obra, *in loco*, para realizar os cortes, por exemplo. Com efeito, há trabalhos nesse sentido na literatura, como ARAUJO (2014), que considera os tempos de *setup* no desenvolvimento do algoritmo genético.

Sob outro prisma, a metodologia, conforme visto, pode ser adaptada para situações concretas, impactando diretamente no custo da obra, todavia deve ser aplicada com os cortes de difícil solução intuitiva, pois a complexidade aumenta de forma muito rápida, inviabilizando o uso do algoritmo para cortes pequenos quando comparados com o comprimento total da barra.

5 TRABALHOS FUTUROS

Este trabalho não conseguiu solucionar o problema dos dados reais da obra utilizando todos os cortes requisitados no projeto em sua plenitude. Para isto pensa-se numa geração de padrões especializada para o uso no algoritmo genético desenvolvido, aperfeiçoando de forma rápida os novos padrões que vão sendo gerados e inseridos na população. Há estudos na literatura, como YANASSE (2006) e UMETAMI (2003) e MCDIARMID (1999), para selecionar padrões eficientes, mas sem ter como finalidade refinar os resultados para um algoritmo genético. Desta forma há bastante ainda o que se pesquisar para o assunto e pode-se trazer ainda à luz resultados mais satisfatórios.

6 REFERÊNCIAS

ALEXANDRE, J. L. ***Otimização do pré projeto de de vigas de concreto armado utilizando algoritmos*** genéticos. Disertação (Mestrado em Engenharia Civil). Instituto Alberto Luiz Coimbra de Pós-Graduação e Pesquisa em Engenharia, Universidade Federal do Rio de Janeiro. 2014.

ARAUJO, Silvio A; POLDI, Kelly Cristina; SMITH, Jim. ***A genetic algorithm for the one-dimensional cutting stock problem with setups***. Pesquisa Operacional, v. 34, n. 2, p. 165-187, 2014.

ARAUJO, Silvio A.; CONSTANTINO, Ademir A.; POLDI, Kelly C. ***An evolutionary algorithm for the one-dimensional cutting stock problem***. *International Transactions in Operational Research*, v. 18, n. 1, p. 115-127, 2011.

ARENALES, M. N. *et al.* **Pesquisa Operacional: para cursos de engenharia**. Rio de Janeiro: Elsevier: ABREPRO,2011

ARENALES, M. N., POLDI, K. C., ***Heurística para o problema de corte e estoque unidimensional inteiro***. Pesquisa Operacional vol 26, número 3. Rio de Janeiro. 2006

ARROYO, J. E. C., ***Heurísticas e metaheuristicas para otimização combinação multiobjetivo***. Tese (Doutorado em Engenharia Elétrica). Faculdade de Engenharia Elétrica e Computação, Universidade Estadual de Campinas, p 23. 2002.

BOTAN A.A.; MAWDESLEY, M. J.; STACE, R. ***A genetic algorithm tp highway alignment development***. *The University of Nottingham*, UK, 2010.

BRASIL. Ministério do Desenvolvimento Regional. Secretaria Nacional de Saneamento. Sistema Nacional de Informações sobre o Saneamento (SNIS), ***Diagnóstico do manejo de Resíduos Sólidos Urbanos***. Brasilia – 2010

CHEN, Y. H., ***A genetic Algorithm approach for the multiple length cutting stock problem***. *1st Global Conference on Life Science and Technologies*. 2019

CHENG, C. H.; FEIRING, B. R.; CHENG, T. C. E. ***The cutting stock problem – a survey.*** *International Journal of Production Economics*, v. 36, n. 3, p. 291-305, 1994.

CHERRI, A. C., ***Algumas** extensões **do problema de corte e estoque com material reaproveitáveis**.* Tese (Doutorado em Ciências da Computação). Instituto de Ciências Matemáticas e Computação, Universidade de São Paulo, p 5-7. 2009.

CHVÁTAL, V. ***Linear** Programming*, 1ed, *New York*. *W.H. Freeman Co*. 1983.

COSTA, Luis Henrique *Magalhães*; CASTRO, Marco Aurélio Holanda de; RAMOS, Helena. ***Utilização de um algoritmo genético híbrido para operação ótima de sistemas de abastecimento de água**.* Eng. Sanit. Ambient., Rio de Janeiro , v. 15, n. 2, p. 187-196, June 2010 . Available from <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S141341522010000200011&lng=en&nrm=iso>. access on 17 Mar. 2021

DE CARVALHO, JM Valério. *Exact **solution of cutting stock problems using column generation and branch-and-bound***. *International Transactions in Operational Research*, v. 5, n. 1, p. 35-44, 1998.

DEDE, T. KRIPKA, M. TOGAN, V. ***YEPES***, V. RAO, R. V. ***Usage of optmization techniques in Civil Engineering during the last two decades***. *Current Trends in Civil & Structural Engineering*, 2019.

DEICHMANN, A. ***Projeto de estrutural em concreto armado de uma residência de 2 pavimentos***. Trabalho de Conclusão Curso (Engenharia Civil). Centro Tecnologico, Universidade Federal de Santa Catarina, Apendice E. 2016.

DYCKHOFF, H., WÄRSHCER, G., ***Cutting and Packing.*** *European Journal of Operational Research*, v.44, n.2, *special issue*, 1990.

FOERSTER, Hildegard; WÄSCHER, Gerhard. ***Pattern reduction in one-dimensional cutting stock problems**. International Journal of Production Research*, 38:7, 1999.

GAREY, M. R., JOHNSON, D. S. ***Computers and Intractability: A guide to the theory of NP-Completeness***, 1ed, *New York*. *W.H. Freeman Co*. 1990.

GOLDBARG, E. G., *et al.*, ***Otimização Combinatória e Meta-Heurísticas: Algoritmos e Aplicação***: 1 ed, Rio de Janeiro. Elsevier, 2016.

GILMORE, Paul C.; ***GOMORY***, Ralph E. ***A linear programming approach to the cutting-stock problem***. *Operations research*, v. 9, n. 6, p. 849-859, 1961.

GOLDBERG, D. E., ***Genetic Algorithms in search, optimization and machine learning***: 1 ed. *Addinson-Wesley Publishing Company*. USA/CA. 1989.

HAESSLER, Robert W.; SWEENEY, Paul E. ***Cutting stock problems and solution procedures.*** *European Journal of Operational Research*, v. 54, n. 2, p. 141-150, 1991.

INSTITUTO DE PESQUISA ECONÔMICA APLICADA (IPEA), ***Diagnóstico dos Resíduos Sólidos da Construção Civil: Relatório de Pesquisa***, p 29. Brasília, 2012.

KHALIFA, Yaser; SALEM, O.; SHAHIN, Adham. ***Cutting stock waste reduction using genetic algorithms.*** *In: Proceedings of the 8th annual conference on Genetic and evolutionary computation*. 2006. p. 1675-1680.

KANTOROVICH, L. V. ***Mathematical Methods of Organising and Planing Production****, Management Science* 6(4): 363-422. 1960. (Tradução de uma pubicação em russo, datada de 1939)

MCDIARMID, Colin. ***Pattern minimisation in cutting stock problems. Discrete applied mathematics***, v. 98, n. 1-2, p. 121-130, 1999.

OLIVEIRA, N. C. P, ***Aplicabilidade dos Algoritmos Genéticos na Construção Civil***. Trabalho de Conclusão de Curso (Graduação em Engenharia Civil). Departamento de Engenharia Civil e Ambiental, Universidade Federal da Paraíba. 2017

ONWUBOLU, Godfrey C.; MUTINGI, Michael. ***A genetic algorithm approach for the cutting stock problem.*** *Journal of Intelligent Manufacturing*, v. 14, n. 2, p. 209-218, 2003.

POZO, A. *et al*, **Computação Evolutiva**. Disponível em: http://www.inf.ufpr.br/aurora/tutoriais/Ceapostila.pdf. 1998

SHAHIM, A. A., SALEM, O.M. ***Using genetic algortihms in solving the one-dimensional cutting stock problem in the construction industry***. NRC *research Press Web, Canadá.* P 321-331. 2003.

STADTLER, H. A ***one-dimensional cutting stock problem in the aluminium industry and its solution.*** *European Journal of Operational Research*, v. 44, n. 2, p. 209-223, 1990.

STEFANELLO, F., ***Hibridização*** **de métodos exatos e heurísticos para a resolução de problemas de otimização combinatória**. Dissertação (Mestrado em Ciência da Computação). Centro de Tecnologia, Universidade Federal de Santa Maria, p 14. 2011.

SULIMAN, Saad MA. ***Pattern generating procedure for the cutting stock problem.*** *International Journal of Production Economics*, v. 74, n. 1-3, p. 293-301, 2001.

TACLA, C. A. *et al*, **Notas de aula sobre algoritmo genético**. Disponível em: https://pessoal.dainf.ct.utfpr.edu.br/tacla/IA/016a-AlgGeneticos.pdf. 2015.

UMETANI, Shunji; YAGIURA, Mutsunori; IBARAKI, Toshihide. ***One-dimensional cutting stock problem to minimize the number of different patterns.*** *European Journal of Operational Research*, v. 146, n. 2, p. 388-402, 2003.

VAHRENKAMP, Richard. ***Random search in the one-dimensional cutting stock problem.*** *European Journal of Operational Research*, v. 95, n. 1, p. 191-200, 1996.

WÄRSHCER, G., HAUβNER, H., SCHUMANN, H., ***An improved typology cutting and packing problems***. *European Journal of Operation Research*, 183: 1109-1130, 2007.

YANASSE, Horacio Hideki; LIMEIRA, Marcelo Saraiva. ***A hybrid heuristic to reduce the number of different patterns in cutting stock problems***. *Computers & Operations Research*, v. 33, n. 9, p. 2744-2756, 2006.

Anexos

Anexo A1. Lista de padrões possíveis da tabela 4

Padrão 1 (0 0 0 0 0 35) resto = 65

Padrão 2 (0 0 0 0 0 70) resto = 30

Padrão 3 (0 0 0 0 41 0) resto = 59

Padrão 4 (0 0 0 0 41 35) resto = 24

Padrão 5 (0 0 0 0 82 0) resto = 18

Padrão 6 (0 0 0 45 0 0) resto = 55

Padrão 7 (0 0 0 45 0 35) resto = 20

Padrão 8 (0 0 0 45 41 0) resto = 14

Padrão 9 (0 0 0 90 0 0) resto = 10

Padrão 10 (0 0 36 0 0 0) resto = 64

Padrão 11 (0 0 36 0 0 35) resto = 29

Padrão 12 (0 0 36 0 41 0) resto = 23

Padrão 13 (0 0 36 45 0 0) resto = 19

Padrão 14 (0 0 72 0 0 0) resto = 28

Padrão 15 (0 31 0 0 0 0) resto = 69

Padrão 16 (0 31 0 0 0 35) resto = 34

Padrão 17 (0 31 0 0 41 0) resto = 28

Padrão 18 (0 31 0 45 0 0) resto = 24

Padrão 19 (0 31 36 0 0 0) resto = 33

Padrão 20 (0 62 0 0 0 0) resto = 38

Padrão 21 (0 62 0 0 0 35) resto = 3

Padrão 22 (0 62 36 0 0 0) resto = 2

Padrão 23 (0 93 0 0 0 0) resto = 7

Padrão 24 (14 0 0 0 0 0) resto = 86

Padrão 25 (14 0 0 0 0 35) resto = 51

Padrão 26 (14 0 0 0 0 70) resto = 16

Padrão 27 (14 0 0 0 41 0) resto = 45

Padrão 28 (14 0 0 0 41 35) resto = 10

Padrão 29 (14 0 0 0 82 0) resto = 4

Padrão 30 (14 0 0 45 0 0) resto = 41

Padrão 31 (14 0 0 45 0 35) resto = 6

Padrão 32 (14 0 0 45 41 0) resto = 0

Padrão 33 (14 0 36 0 0 0) resto = 50

Padrão 34 (14 0 36 0 0 35) resto = 15

Padrão 35 (14 0 36 0 41 0) resto = 9

Padrão 36 (14 0 36 45 0 0) resto = 5

Padrão 37 (14 0 72 0 0 0) resto = 14

Padrão 38 (14 31 0 0 0 0) resto = 55

Padrão 39 (14 31 0 0 0 35) resto = 20

Padrão 40 (14 31 0 0 41 0) resto = 14

Padrão 41 (14 31 0 45 0 0) resto = 10

Padrão 42 (14 31 36 0 0 0) resto = 19

Padrão 43 (14 62 0 0 0 0) resto = 24

Padrão 44 (28 0 0 0 0 0) resto = 72

Padrão 45 (28 0 0 0 0 35) resto = 37

Padrão 46 (28 0 0 0 0 70) resto = 2

Padrão 47 (28 0 0 0 41 0) resto = 31

Padrão 48 (28 0 0 45 0 0) resto = 27

Padrão 49 (28 0 36 0 0 0) resto = 36

Padrão 50 (28 0 36 0 0 35) resto = 1

Padrão 51 (28 0 72 0 0 0) resto = 0

Padrão 52 (28 31 0 0 0 0) resto = 41

Padrão 53 (28 31 0 0 0 35) resto = 6

Padrão 54 (28 31 0 0 41 0) resto = 0

Padrão 55 (28 31 36 0 0 0) resto = 5

Padrão 56 (28 62 0 0 0 0) resto = 10

Padrão 57 (42 0 0 0 0 0) resto = 58

Padrão 58 (42 0 0 0 0 35) resto = 23

Padrão 59 (42 0 0 0 41 0) resto = 17

Padrão 60 (42 0 0 45 0 0) resto = 13

Padrão 61 (42 0 36 0 0 0) resto = 22

Padrão 62 (42 31 0 0 0 0) resto = 27

Padrão 63 (56 0 0 0 0 0) resto = 44

Padrão 64 (56 0 0 0 0 35) resto = 9

Padrão 65 (56 0 0 0 41 0) resto = 3

Padrão 66 (56 0 36 0 0 0) resto = 8

Padrão 67 (56 31 0 0 0 0) resto = 13

Padrão 68 (70 0 0 0 0 0) resto = 30

Padrão 69 (84 0 0 0 0 0) resto = 16

Padrão 70 (98 0 0 0 0 0) resto = 2