ALGORITMO GRASP PARA O PROBLEMA DE CORTE BIDIMENSIONAL

Lucas Felix Dantas Rocha
l136625@dac.unicamp.br
Francisco A. M. Gomes
chico@ime.unicamp.br
INSTITUTO DE MATEMÁTICA, ESTÁTISTICA E COMPUTAÇÃO CIENTÍFICA (IMECC)
Órgão Financiador: Sistema de Apoio ao Estudante (SAE)
Palavras-chave: Computação - Corte Bidimensional – BLP - GRASP

Introdução

Muitos setores da indústria que trabalham com chapas necessitam cortá-las em peças menores, desperdiçando o mínimo de material possível. Este problema, denominado Problema de Corte Bidimensional (PCB), é de natureza combinatória, o que torna inviável que seja resolvido de forma exata devido ao tempo que se gastaria para chegar a uma solução ótima.

Objetivos

Neste projeto, buscamos desenvolver um programa que solucione, de forma eficiente, um PCB que seja ponderado, orientado, guilhotinado, com dois estágios e com aparagem.

Metodologia

Implantamos, no ambiente MATLAB, um algoritmo para o PCB baseado no Procedimento de Alocação Inferior Esquerdo (BLP). Como evidenciado no próprio nome, esse algoritmo inicia o posicionamento das peças a serem cortadas no extremo inferior esquerdo da placa. O programa inicia a alocação pela peça de maior altura, e assim que sua demanda for igual a zero, ele passa para a próxima peça de maior altura. Como trabalhamos com o problema de corte em dois estágios, a altura da primeira peça posicionada define a altura da primeira tira. Quando não houver mais espaço nessa tira, ou a peça a ser colocada é maior que o espaço restante, o algoritmo cria outra tira, caso exista espaço na placa. Importante ressaltar que, sempre que o algoritmo passa para a próxima peça, ele verifica se não há espaço nas tiras já utilizadas para sua alocação. Também implantamos a primeira fase da meta-heurística GRASP adaptada ao PCB, denominada Fase de Construção. Esta fase inicia-se estabelecendo todos os elementos (peças) que podem ser incorporados à solução parcial (tira) em construção, sem que se destrua sua factibilidade. Estes elementos são, então, selecionados por uma função gulosa (Greedy) que avalia quais deles fornecerão o menor custo incremental (neste caso, a maior área utilizada). Cria-se, assim, uma Lista Restrita de Candidatos (cujo acrônimo em inglês é RCL). Em seguida, um elemento desta lista é escolhido de forma aleatória para fazer parte da solução parcial e, então, a RCL é atualizada, repetindo-se todos os procedimentos até que se atinja algum critério de parada. Futuramente, também implantaremos um algoritmo que realizará a fase de busca local do GRASP. Quando terminado, os 3 algoritmos serão reunidos em um programa final cuja eficiência será testada com problemas encontrados na literatura e comparado com algoritmos anteriores.

Resultados

Até o momento, foram produzidos programas que reproduzem com sucesso o BLP e o GRASP, representados pelos pseudocódigos abaixo.

```
procedure DesenhoPCB(d,c,a,C,A)
1       Ordenar os parâmetros de entrada por uma ordem decrescente de altura
2       para n = 1:np,
3               enquanto b(n) > 0,
4                       i = 1;
5                       se c(i) <= resto(i)
6                               Adicionar peça n à tira i e reduzir resto(i)
7                       caso contrário
8                               Aumentar i, criando uma nova tira se necessário
9       imprima Matriz S com as peças existentes em cada tira e gráfico com os cortes
fim DesenhoPCB;
```

Figura 1: Pseudocódigo do BLP.

```
procedimento Fase_de_Construção(Seed)
1       Solução = 0
2       Avaliar os custos incrementais dos elementos candidatos;
3       enquanto Solução não for uma solução completa,
4               Construa a lista restrita de candidatos (RCL);
5               Selecione um elemento s da RCL aleatoriamente;
6               Solução = Solução U {s}
7               Reavalie os custos incrementais;
8       fim
9       imprima Solução;
```

Figura 2: Pseudocódigo da fase de construção do GRASP.

As figuras ao lado representam os gráficos com os cortes realizados na placa para o caso do algoritmo BLP e da fase de construção do GRASP. As três figuras possuem peças em proporções e demandas diversas. Porém, somente na terceira figura algumas foram deixadas de lado, dadas as especificidades já elencadas da fase de construção do GRASP.

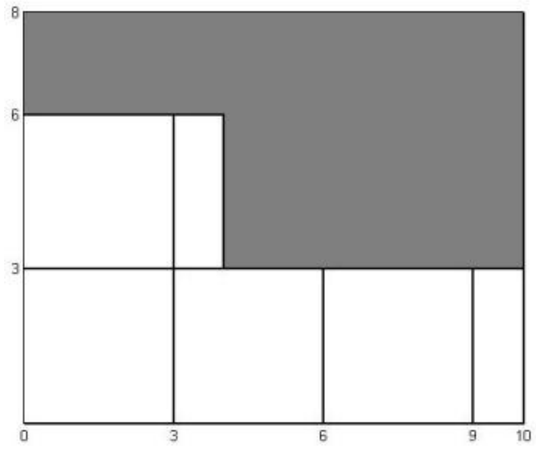

Figura 3: 1° exemplo do PCB

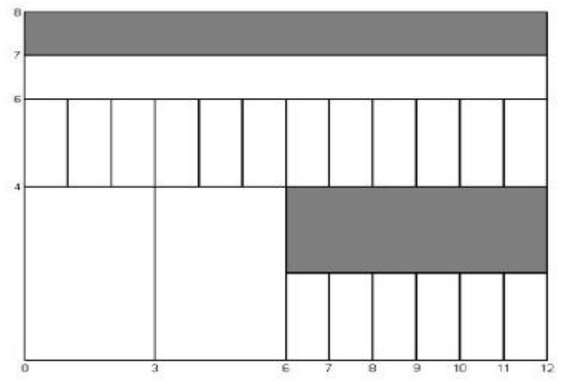

Figura 4: 2° exemplo do PCB

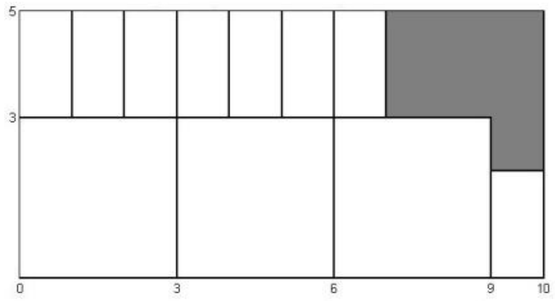

Figura 5: fase de construção do GRASP

Conclusão

Os resultados apresentados pelo BLP e pela fase de construção do GRASP foram satisfatórios. Até o fim do projeto, pretendemos implementar a fase de busca local do GRASP, para podermos aplicar o programa completo à solução de problemas encontrados na literatura e comparar os resultados com os algoritmos já desenvolvidos.