GRASP

Aplicado ao problema de aterrissagem de aviões

Bruno Fiss
Kauê Silveira

Instituto de Informática - UFRGS

23 de junho de 2010

1 Definição do problema
- Formalização
- Descrição como programa inteiro
- Representação de uma solução

2 GRASP
- Algoritmo
- Geração de soluções
- Busca local

3 Experimentos
- Resultados
- Análise

Descrição

- O problema de aterrissagem de aviões consiste em definir um momento no tempo para a aterrissagem de cada avião $i \in P$, sendo P o conjunto de aviões. Cada avião possui os seguintes dados:

Descrição

- O problema de aterrissagem de aviões consiste em definir um momento no tempo para a aterrissagem de cada avião $i \in P$, sendo P o conjunto de aviões. Cada avião possui os seguintes dados:
- E_i: Momento mais prematuro em que o avião i pode realizar pouso.

Descrição

- O problema de aterrissagem de aviões consiste em definir um momento no tempo para a aterrissagem de cada avião $i \in P$, sendo P o conjunto de aviões. Cada avião possui os seguintes dados:
- E_i: Momento mais prematuro em que o avião i pode realizar pouso.
- T_i: Momento ideal para pouso do avião i.

Descrição

- O problema de aterrissagem de aviões consiste em definir um momento no tempo para a aterrissagem de cada avião $i \in P$, sendo P o conjunto de aviões. Cada avião possui os seguintes dados:
- E_i: Momento mais prematuro em que o avião i pode realizar pouso.
- T_i: Momento ideal para pouso do avião i.
- L_i: Momento mais tardio em que o avião i pode realizar pouso.

Descrição

- O problema de aterrissagem de aviões consiste em definir um momento no tempo para a aterrissagem de cada avião $i \in P$, sendo P o conjunto de aviões. Cada avião possui os seguintes dados:
- E_i: Momento mais prematuro em que o avião i pode realizar pouso.
- T_i: Momento ideal para pouso do avião i.
- L_i: Momento mais tardio em que o avião i pode realizar pouso.
- g_i: Penalidade por unidade de tempo da diferença do pouso para o tempo ideal se o avião chegar mais cedo do que o ideal.

Descrição

- O problema de aterrissagem de aviões consiste em definir um momento no tempo para a aterrissagem de cada avião $i \in P$, sendo P o conjunto de aviões. Cada avião possui os seguintes dados:
- E_i: Momento mais prematuro em que o avião i pode realizar pouso.
- T_i: Momento ideal para pouso do avião i.
- L_i: Momento mais tardio em que o avião i pode realizar pouso.
- g_i: Penalidade por unidade de tempo da diferença do pouso para o tempo ideal se o avião chegar mais cedo do que o ideal.
- h_i: Penalidade por unidade de tempo da diferença do pouso para o tempo ideal se o avião chegar mais tarde do que o ideal.

Descrição

- O problema de aterrissagem de aviões consiste em definir um momento no tempo para a aterrissagem de cada avião $i \in P$, sendo P o conjunto de aviões. Cada avião possui os seguintes dados:
- E_i: Momento mais prematuro em que o avião i pode realizar pouso.
- T_i: Momento ideal para pouso do avião i.
- L_i: Momento mais tardio em que o avião i pode realizar pouso.
- g_i: Penalidade por unidade de tempo da diferença do pouso para o tempo ideal se o avião chegar mais cedo do que o ideal.
- h_i: Penalidade por unidade de tempo da diferença do pouso para o tempo ideal se o avião chegar mais tarde do que o ideal.
- S_{ij}: Distância de tempo requerida após o pouso do avião i para que o avião j possa pousar.

Descrição

- O objetivo é encontrar uma solução com somatório de todas as penalidades mais baixo possível.
- E_i: Momento mais prematuro em que o avião i pode realizar pouso.
- T_i: Momento ideal para pouso do avião i.
- L_i: Momento mais tardio em que o avião i pode realizar pouso.
- g_i: Penalidade por unidade de tempo da diferença do pouso para o tempo ideal se o avião chegar mais cedo do que o ideal.
- h_i: Penalidade por unidade de tempo da diferença do pouso para o tempo ideal se o avião chegar mais tarde do que o ideal.
- S_{ij}: Distância de tempo requerida após o pouso do avião i para que o avião j possa pousar.

Variáveis

- Variáveis x_i representam o momento de aterrissagem do avião i.

Variáveis

- Variáveis x_i representam o momento de aterrissagem do avião i.
- Variáveis α_i e β_i representam o quão adiantado ou atrasado, respectivamente, o avião i chega em relação ao seu tempo ideal.

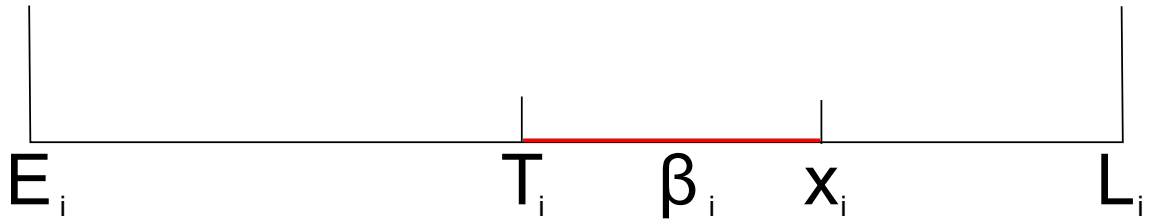

Figura: Exemplo da variável β_i

Variáveis

- Variáveis x_i representam o momento de aterrissagem do avião i.
- Variáveis α_i e β_i representam o quão adiantado ou atrasado, respectivamente, o avião i chega em relação ao seu tempo ideal.

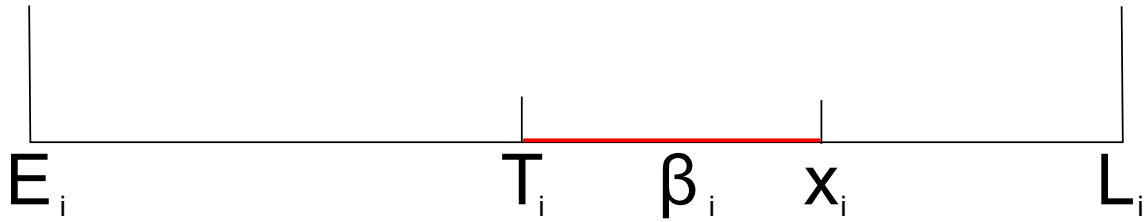

Figura: Exemplo da variável β_i

- Variáveis binárias δ_{ij} indicam se o avião i aterrissa antes do avião j.

Restrições

$$\textbf{minimiza} \quad \sum_{i \in P} g_i \alpha_i + h_i \beta_i$$

$$\begin{aligned}
\textbf{sujeito a} \quad & x_i = -\alpha_i + \beta_i + T_i, && \forall i \in P \\
& E_i \leq x_i \leq L_i, && \forall i \in P \\
& x_j - x_i \geq S_{ij}\delta_{ij} + (E_j - L_i)\delta_{ji}, && \forall i, j \in P \\
& \delta_{ij} + \delta_{ji} = 1, && \forall i, j \in P \\
& x_i \geq 0, x_i \in \mathbb{R}, && \forall i \in P \\
& \alpha_i \geq 0, \alpha_i \in \mathbb{R}, && \forall i \in P \\
& \beta_i \geq 0, \beta_i \in \mathbb{R}, && \forall i \in P \\
& \delta_{ij} \in \mathbb{B}, && \forall i, j \in P
\end{aligned}$$

Restrições

As variáveis δ_{ij} e as restrições $x_j - x_i \geq S_{ij}\delta_{ij} + (E_j - L_i)\delta_{ji}, \forall i, j \in P$ podem ser desnecessárias de acordo com as relações entre os aviões i e j.

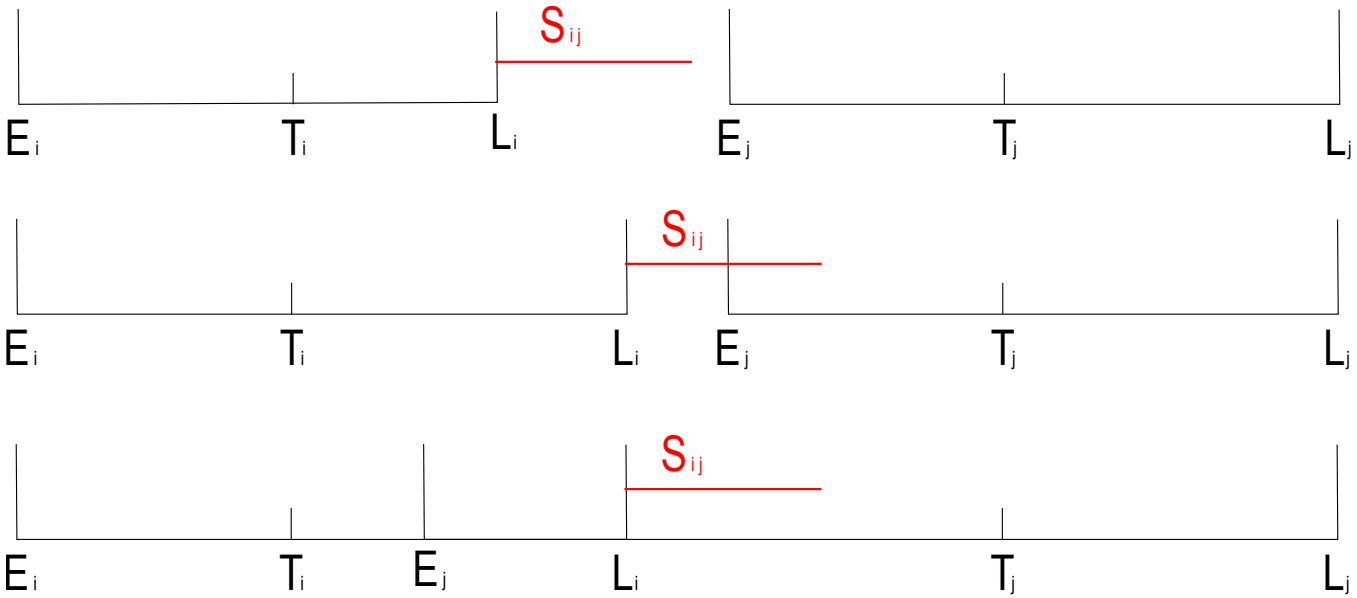

Figura: Diferentes situações entre aviões

Variáveis inteiras

- A parte mais complicada do problema está na definição das variáveis inteiras, δ_{ij}.

Variáveis inteiras

- A parte mais complicada do problema está na definição das variáveis inteiras, δ_{ij}.
- A solução do problema será considerada a instanciação dessas variáveis, pois com essa podemos resolver o problema linear resultante, de forma eficiente, com o método Simplex.

Sequência de aviões

- Para representar a solução, decidimos utilizar uma representação de sequência de aviões, na qual a ordem dos aviões diz a ordem de chegada desses ao aeroporto.

Sequência de aviões

- Para representar a solução, decidimos utilizar uma representação de sequência de aviões, na qual a ordem dos aviões diz a ordem de chegada desses ao aeroporto.

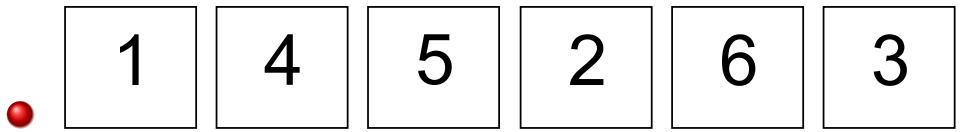

Figura: Uma sequência de aviões

Sequência de aviões

- Para representar a solução, decidimos utilizar uma representação de sequência de aviões, na qual a ordem dos aviões diz a ordem de chegada desses ao aeroporto.

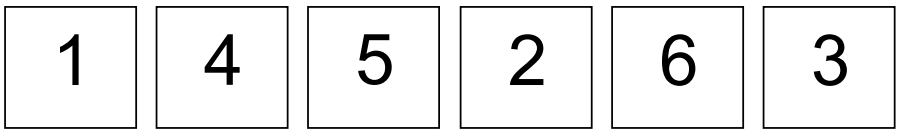

Figura: Uma sequência de aviões

- A partir dessa representação podemos definir as variáveis δ_{ij}, e vice-versa.

Idéia geral

- A idéia principal da meta-heurística GRASP é construir soluções iniciais de uma forma gulosa porém aleatória.

Idéia geral

- A idéia principal da meta-heurística GRASP é construir soluções iniciais de uma forma gulosa porém aleatória.
- Em vez de selecionar-se o melhor elemento a ser inserido numa solução, escolhe-se um entre alguns dos melhores de forma aleatória.

Idéia geral

- A idéia principal da meta-heurística GRASP é construir soluções iniciais de uma forma gulosa porém aleatória.
- Em vez de selecionar-se o melhor elemento a ser inserido numa solução, escolhe-se um entre alguns dos melhores de forma aleatória.
- Após isso se faz uma busca local em torno da solução inicial gerada.

Idéia geral

- A idéia principal da meta-heurística GRASP é construir soluções iniciais de uma forma gulosa porém aleatória.
- Em vez de selecionar-se o melhor elemento a ser inserido numa solução, escolhe-se um entre alguns dos melhores de forma aleatória.
- Após isso se faz uma busca local em torno da solução inicial gerada.
- A melhor solução encontrada durante a execução do algoritmo é retornada como resultado.

Função principal

```
procedimento GRASP(alpha, semente)
    melhorSolucao = INF;
    enquanto nao criterio-de-parada faca
        sol = gerarSolucaoInicial(alpha, semente);
        res = buscaLocal(sol);
        se res < melhorSolucao entao
            melhorSolucao = res;
        fim-se;
    fim-enquanto;
    retornar melhorSolucao;
fim-procedimento;
```

Sequência ideal

- De forma a construir soluções com um algoritmo semi-guloso foi necessário definir o conceito de sequência ideal.

Sequência ideal

- De forma a construir soluções com um algoritmo semi-guloso foi necessário definir o conceito de sequência ideal.
- Essa sequência serve de base para a determinação de quais são os melhores elementos a serem inseridos em um dado passo do algoritmo.

Sequência ideal

- De forma a construir soluções com um algoritmo semi-guloso foi necessário definir o conceito de sequência ideal.
- Essa sequência serve de base para a determinação de quais são os melhores elementos a serem inseridos em um dado passo do algoritmo.
- Essa sequência é incialmente definida como a ordem em que os aviões chegariam caso todos aterrissassem no seu momento ideal.

Sequência ideal

- De forma a construir soluções com um algoritmo semi-guloso foi necessário definir o conceito de sequência ideal.
- Essa sequência serve de base para a determinação de quais são os melhores elementos a serem inseridos em um dado passo do algoritmo.
- Essa sequência é incialmente definida como a ordem em que os aviões chegariam caso todos aterrissassem no seu momento ideal.
- Cada vez que uma solução melhor do que a atual é encontrada, durante a execução do algoritmo, a sequência ideal é atualizada para a sequência que gerou a nova solução.

Parâmetro α

- Parâmetro do método GRASP que define o quão "ruim" um elemento pode ser e ainda assim ser selecionado.

Parâmetro α

- Parâmetro do método GRASP que define o quão "ruim" um elemento pode ser e ainda assim ser selecionado.
- Na nossa implementação, esse parâmetro equivale à máxima distância aceitável entre a posição do avião na sequência ideal e a posição a ser preenchida na solução parcial.

Variação: Reactive GRASP

- A meta-heurística GRASP original mantinha o parâmetro α fixo durante todas as iterações do algoritmo.

Variação: Reactive GRASP

- A meta-heurística GRASP original mantinha o parâmetro α fixo durante todas as iterações do algoritmo.
- Com base no Reactive GRASP, nosso algoritmo escolhe aleatoriamente, a cada iteração, um valor para α entre 0 e α_{max}, um novo parâmetro do nosso método.

Função geradora de soluções

```
procedimento gerarSolucaoInicial(alpha, semente)
    solucao = vazio;
    enquanto naoPronta(solucao) faca
        RCL = gerarRCL(alpha);
        elemento = sortear(RCL,semente);
        adicionarElemento(solucao,elemento);
    fim-enquanto;
    retornar solucao;
fim-procedimento;
```

RCL

- RCL (Resctricted candidate list) é a lista de todos os elementos (nesse caso aviões) que podem ser inseridos em um determinado passo da construção da solução.

RCL

- RCL (Resctricted candidate list) é a lista de todos os elementos (nesse caso aviões) que podem ser inseridos em um determinado passo da construção da solução.
- Todos os elementos dessa lista satisfazem às restrições citadas na descrição do parâmetro α, e também atendem ao critério de que, se forem adicionados à solução, mantê-la-ão viável.

RCL - Exemplo

Nas figuras abaixo podemos ver um exemplo de sequência ideal e de uma solução parcial. Nesse exemplo, se α fosse 2, a RCL seria _.

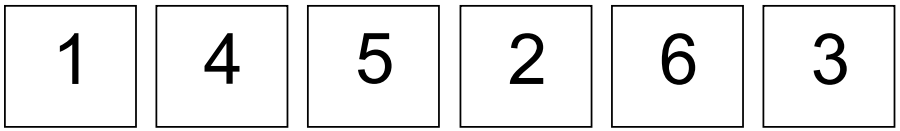

Figura: Exemplo de sequência ideal

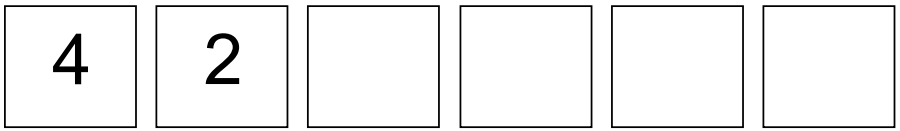

Figura: Exemplo de solução parcial

RCL - Exemplo

Nas figuras abaixo podemos ver um exemplo de sequência ideal e de uma solução parcial. Nesse exemplo, se α fosse 2, a RCL seria $\{1, 5, 6\}$.

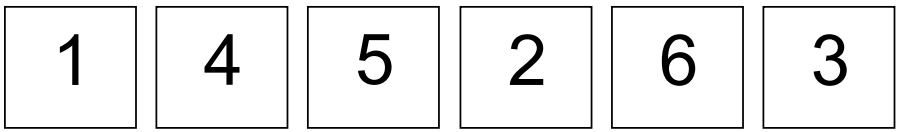

Figura: Exemplo de sequência ideal

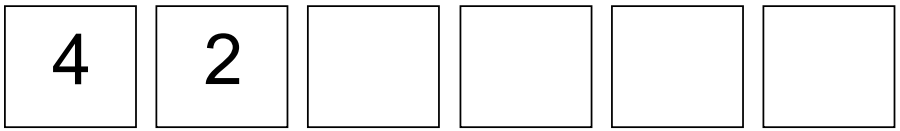

Figura: Exemplo de solução parcial

Vizinhança e busca local

- A segunda parte de cada iteração do método GRASP consiste em realizar uma busca local ao redor da solução gerada gulosa e aleatoriamente.

Vizinhança e busca local

- A segunda parte de cada iteração do método GRASP consiste em realizar uma busca local ao redor da solução gerada gulosa e aleatoriamente.
- A definição de um vizinho de uma sequência é: uma sequência é vizinha de outra se for resultante da troca de posição entre dois aviões adjacentes e de mais nenhuma alteração.

Vizinhança e busca local

- A segunda parte de cada iteração do método GRASP consiste em realizar uma busca local ao redor da solução gerada gulosa e aleatoriamente.
- A definição de um vizinho de uma sequência é: uma sequência é vizinha de outra se for resultante da troca de posição entre dois aviões adjacentes e de mais nenhuma alteração.
- O tipo de busca local escolhido foi o de primeiro incremento, isto é, pesquisa-se na vizinhança por uma solução melhor e, quando encontra-se a primeira solução melhor, passa-se a considerar essa nova solução como a solução atual, reiniciando a pesquisa nos vizinhos dessa.

Resultados I

Testamos, para todos os casos de entrada, o parâmetro α_{max} com valores entre 1 e 5, sempre utilizando sementes aleatórias diferentes.

Tabela: Tabela de resultados dos experimentos

Caso	α_{max}	Semente	Solução	Tempo (s)	Desvio (%)
airland1	1	14766	700	1	0
airland1	2	20024	700	2	0
airland1	3	19673	700	1	0
airland1	4	23915	700	2	0
airland1	5	710	700	2	0

Resultados II

Tabela: Tabela de resultados dos experimentos

Caso	α_{max}	Semente	Solução	Tempo (s)	Desvio (%)
airland2	1	22574	1.480	5	0
airland2	2	26494	1.480	9	0
airland2	3	11040	1.480	8	0
airland2	4	27382	1.480	9	0
airland2	5	11503	1.500	10	1.33

Resultados III

Tabela: Tabela de resultados dos experimentos

Caso	α_{max}	Semente	Solução	Tempo (s)	Desvio (%)
airland3	1	15751	820	18	0
airland3	2	14185	820	21	0
airland3	3	31028	820	21	0
airland3	4	20012	820	21	0
airland3	5	17962	820	21	0

Resultados IV

Tabela: Tabela de resultados dos experimentos

Caso	α_{max}	Semente	Solução	Tempo (s)	Desvio (%)
airland4	1	26543	2.520	18	0
airland4	2	23571	2.520	21	0
airland4	3	21435	2.520	21	0
airland4	4	3896	2.520	21	0
airland4	5	21136	2.520	21	0

Resultados V

Tabela: Tabela de resultados dos experimentos

Caso	α_{max}	Semente	Solução	Tempo (s)	Desvio (%)
airland5	1	20362	3.100	21	0
airland5	2	19016	3.100	21	0
airland5	3	28928	3.100	21	0
airland5	4	13363	3.100	21	0
airland5	5	4087	3.100	21	0

Resultados VI

Tabela: Tabela de resultados dos experimentos

Caso	α_{max}	Semente	Solução	Tempo (s)	Desvio (%)
airland6	1	27207	24.442	1	0
airland6	2	14705	24.442	1	0
airland6	3	31960	24.442	2	0
airland6	4	32355	24.442	1	0
airland6	5	15148	24.442	1	0

Resultados VII

Tabela: Tabela de resultados dos experimentos

Caso	α_{max}	Semente	Solução	Tempo (s)	Desvio (%)
airland7	1	4734	1.550	12	0
airland7	2	11699	1.550	13	0
airland7	3	23048	1.550	13	0
airland7	4	30431	1.550	14	0
airland7	5	10183	1.550	14	0

Resultados VIII

Tabela: Tabela de resultados dos experimentos

Caso	α_{max}	Semente	Solução	Tempo (s)	Desvio (%)
airland8	1	30714	1.950	21	0
airland8	2	13072	1.950	33	0
airland8	3	16746	1.950	40	0
airland8	4	20341	1.950	21	0
airland8	5	21053	1.950	28	0

Análise e conclusões

- A escolha dos parâmetros e funções envolvidas na construção gulosa-aleatória de soluções é fundamental para o sucesso do algoritmo.

Análise e conclusões

- A escolha dos parâmetros e funções envolvidas na construção gulosa-aleatória de soluções é fundamental para o sucesso do algoritmo.
- Aparentemente, a escolha de soluções iniciais próximas de uma sequência ideal pré-definida é eficaz para a solução desse problema.

Análise e conclusões

- A escolha dos parâmetros e funções envolvidas na construção gulosa-aleatória de soluções é fundamental para o sucesso do algoritmo.
- Aparentemente, a escolha de soluções iniciais próximas de uma sequência ideal pré-definida é eficaz para a solução desse problema.
- A variação do parâmetro α, como proposto na meta-heurística Reactive GRASP, também parece ter sido importante para o sucesso da abordagem.

Obrigado!

Perguntas?