

# Desenvolvimento de uma nova Metaheurística: Roll 'n' Improve

Autor: Leandro Augusto de Araújo Silva  
Orientador: Prof. Dr. Marcone Jamilson Freitas Souza

Universidade Federal de Ouro Preto

Dezembro de 2011



- 1 Introdução
- 2 O Problema de Sequenciamento
- 3 Metodologia
- 4 Resultados
- 5 Conclusões e Trabalhos Futuros



# Sumário

- 1 Introdução
  - Conceitos Iniciais
  - Objetivos
- 2 O Problema de Sequenciamento
- 3 Metodologia
- 4 Resultados
- 5 Conclusões e Trabalhos Futuros



# Problemas NP-Completo:

- Não é possível garantir que a solução ótima seja encontrada em tempo polinomial;
- $\therefore$  não é uma boa alternativa utilizar algoritmos exatos.



# Solução: utilizar heurísticas

- Inspiradas em processos intuitivos;
- procuram uma boa solução a um custo computacional aceitável;
- não garantem otimalidade.



# Melhores Estratégias

Também conhecidas como metaheurísticas ou métodos inteligentes flexíveis:

- 1 os estudos se intensificaram na década de 1980
- 2 procedimentos heurísticos com certa estrutura teórica e caráter mais geral;
- 3 domínios teóricos ainda pouco explorados na literatura;



# Problema!

- Muitas necessitam de uma boa solução inicial para que sejam eficazes e eficientes.



# Solução

- 1 Desenvolver uma nova metaheurística;
- 2 tal que essa seja capaz de excluir a necessidade de outra estratégia para criação de uma boa solução inicial;
- 3 seu nome: “Roll 'n' Improve”.





# PSUMAA

- Problema de Sequenciamento de Tarefas em Uma Máquina com penalidades pelos Adiantamento e Atraso da Produção;
- Filosofia just-in-time;
- tempo de preparação dependente da sequencia de produção;
- Problema NP-Completo.



# Construção Gulosa

- Heurísticas clássicas;
- elementos candidatos são ordenados segundo uma função gulosa;
- função essa que estima o benefício da inserção de cada elementos;
- o melhor, naquele momento, é adicionado à solução.



# GRASP

- Duas fases: construção e refinamento da solução;
- combina elementos de um algoritmo guloso, assim como elementos de aleatoriedade.



# Busca Tabu

- 1 Procedimento de busca local;
- 2 explora o espaço de soluções utilizando uma estrutura de memória;
- 3 move para seu melhor vizinho e armazena a nova solução na memória;
- 4 assim, impede, por um tempo determinado, o retorno a soluções já visitadas.



# Reconexão por Caminhos

- 1 Faz um balanço entre intensificação e diversificação;
- 2 proposta originalmente para ser utilizada em aliança à busca tabu;
- 3 considere um par de soluções, uma base e uma guia;
- 4 o objetivo é partir da solução base e caminhar até à guia através da inserção gradativa de atributos da solução guia na solução base.



# Objetivos

- 1 Mostrar sua adaptabilidade e eficácia para uma extensa classe de problemas de otimização;
- 2 Descrever o funcionamento detalhado da **Roll 'n' Improve**, de forma a permitir sua reprodutibilidade;
- 3 Aplicar a metaheurística, inicialmente, ao PSUMAA;
- 4 Comparar os resultados da **Roll 'n' Improve** nesses problemas com os de outros métodos da literatura;
- 5 Disseminar o uso da nova metaheurística;
- 6 Incluir a metaheurística em conhecidos frameworks;



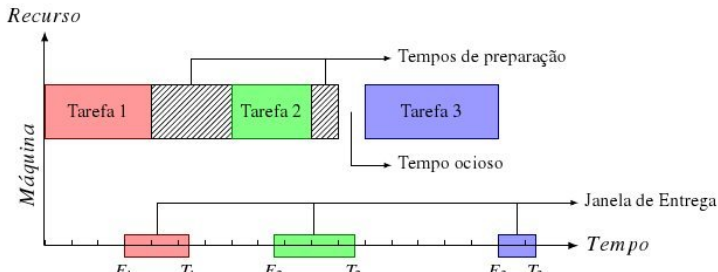
# Sumário

- 1 Introdução
- 2 O Problema de Sequenciamento
  - Descrição do Problema Abordado
- 3 Metodologia
- 4 Resultados
- 5 Conclusões e Trabalhos Futuros



## Problema Abordado - Características

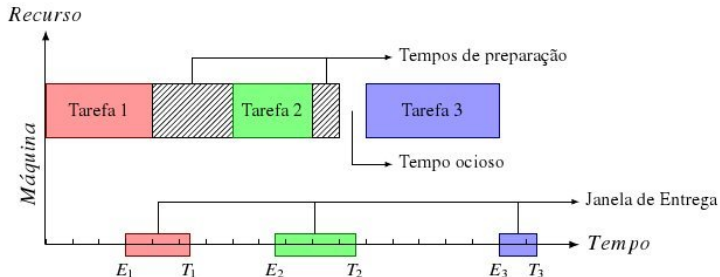
- Problema de sequenciamento em uma máquina;
- a máquina pode executar no máximo uma tarefa por vez e uma vez iniciado o processamento de uma tarefa, não é permitida a sua interrupção;
- penalidades por antecipação e atraso da produção;
- janelas de entrega distintas.





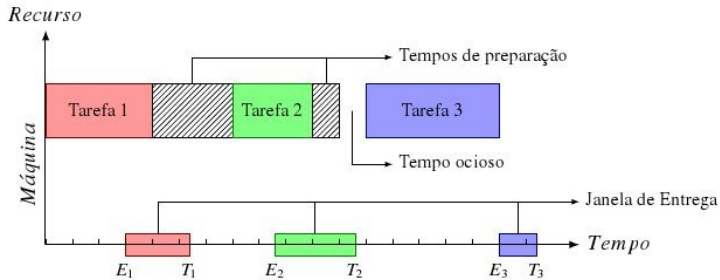
## Problema Abordado - Características

- Tempo de preparação da máquina dependente da sequência de produção;
- assume-se que a máquina não necessita de preparação para processamento da primeira tarefa;
- tempo ocioso entre duas tarefas.



# Problema Abordado - Características

- Todas as tarefas estão disponíveis para processamento na data 0;
- denominado PSUMAA-JP.



# Problema Abordado - Função Objetivo

## Objetivo

Minimizar o somatório dos custos de antecipação e atraso da produção.

$$\sum_{i=1}^n (\alpha_i e_i + \beta_i t_i)$$

# Sumário

- 1 Introdução
- 2 O Problema de Sequenciamento
- 3 Metodologia**
  - Funcionamento
  - Exemplo
  - Algoritmo Proposto
  - Justificativa e Relevância
- 4 Resultados
- 5 Conclusões e Trabalhos Futuros



## De onde veio a idéia

- Partira do fato que, se tem-se uma solução boa, é pouco provável que o elemento de uma posição se encontre em uma distante da sua na solução ótima;



# Roll 'n' Improve

Na prática, a metaheurística funciona da seguinte forma:

- **1** Gere uma solução aleatória;
- **2** Atribua 1 ao denominador;
- **3** Crie a matriz de probabilidades da seguinte forma, através de um único parâmetro (denominador): há uma linha / um vetor para cada elemento; o peso da posição corrente do elemento na solução será 1; seja  $d$  a distância entre a posição avaliada e a corrente, as demais posições possuem peso  $1 / \text{denominador}^d$ ;



## Roll 'n' Improve

- 4 Sorteie um elemento, rode a roleta e o insira na posição que a mesma parar; remova a coluna da referida posição da matriz e continue o procedimento até que a solução seja formada. Se essa solução for melhor que a melhor encontrada até então, retorne a 2;  
Se ela não for de melhora, verifique se o número máximo de soluções sem melhora para um mesmo denominador foi atingido ou se a função objetivo dela é maior  $pMax$  vezes a função objetivo da melhor solução, em caso positivo de quaisquer das duas, incremente o denominador e retorne a 3;



## Roll 'n' Improve

- 5 Se as últimas  $m$  soluções tiverem mesmo valor de função objetivo, inicie a fase de atenuação, da seguinte forma:  $mDen$  = denominador no qual a solução convergiu;
- 6 Atribua 1 ao denominador;
- 7 Crie a matriz de probabilidades tal como em 3, mas, ao invés de passar o denominador como parâmetro da função, passe  $1/denominador$ ;





## Roll 'n' Improve

- 8 Sorteie um elemento, rode a roleta e o insira na posição que a mesma parar; remova a coluna da referida posição da matriz e continue o procedimento até que a solução seja formada. Se essa solução for melhor que a melhor encontrada até então, retorne a 2;  
Se ela não for de melhora, verifique se o número máximo de soluções sem melhora para um mesmo denominador foi atingido ou se a função objetivo dela é maior  $pMax$  vezes a função objetivo da melhor solução, em caso positivo de quaisquer das duas, incremente o denominador e verifique se o mesmo se tornou maior que  $mDen$ , se sim, o resultado do procedimento é a melhor solução encontrada até então, senão retorne a 7;

# Parâmetros

Para um problema de minimização qualquer, considere os seguintes parâmetros para a Nonggis:

- $\alpha$ : valor mínimo do denominador;
- $\beta$ : número máximo de soluções de mesmo valor objetivo consecutivas (critério de convergência);
- $\omega$ : número máximo de iterações com determinado denominador;
- $\kappa$ : valor do incremento do denominador;
- $\theta$ : percentual máximo que uma solução pode ser pior que a melhor corrente para que o denominador não seja incrementado. Este parâmetro evita que muitas soluções potencialmente ruins sejam exploradas.



# Parâmetros

Para um problema de minimização qualquer, considere os seguintes parâmetros para a Nonggis:

- $\alpha$ : valor mínimo do denominador;
- $\beta$ : número máximo de soluções de mesmo valor objetivo consecutivas (critério de convergência);
- $\omega$ : número máximo de iterações com determinado denominador;
- $\kappa$ : valor do incremento do denominador;
- $\theta$ : percentual máximo que uma solução pode ser pior que a melhor corrente para que o denominador não seja incrementado. Este parâmetro evita que muitas soluções potencialmente ruins sejam exploradas.



# Parâmetros

Para um problema de minimização qualquer, considere os seguintes parâmetros para a Nonggis:

- $\alpha$ : valor mínimo do denominador;
- $\beta$ : número máximo de soluções de mesmo valor objetivo consecutivas (critério de convergência);
- $\omega$ : número máximo de iterações com determinado denominador;
- $\kappa$ : valor do incremento do denominador;
- $\theta$ : percentual máximo que uma solução pode ser pior que a melhor corrente para que o denominador não seja incrementado. Este parâmetro evita que muitas soluções potencialmente ruins sejam exploradas.



# Parâmetros

Para um problema de minimização qualquer, considere os seguintes parâmetros para a Nonggis:

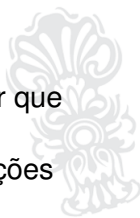
- $\alpha$ : valor mínimo do denominador;
- $\beta$ : número máximo de soluções de mesmo valor objetivo consecutivas (critério de convergência);
- $\omega$ : número máximo de iterações com determinado denominador;
- $\kappa$ : valor do incremento do denominador;
- $\theta$ : percentual máximo que uma solução pode ser pior que a melhor corrente para que o denominador não seja incrementado. Este parâmetro evita que muitas soluções potencialmente ruins sejam exploradas.



## Parâmetros

Para um problema de minimização qualquer, considere os seguintes parâmetros para a Nonggis:

- $\alpha$ : valor mínimo do denominador;
- $\beta$ : número máximo de soluções de mesmo valor objetivo consecutivas (critério de convergência);
- $\omega$ : número máximo de iterações com determinado denominador;
- $\kappa$ : valor do incremento do denominador;
- $\theta$ : percentual máximo que uma solução pode ser pior que a melhor corrente para que o denominador não seja incrementado. Este parâmetro evita que muitas soluções potencialmente ruins sejam exploradas.



# Função Cria Matriz de Probabilidades

---

## Algoritmo 1: CriaMatrizDeProbabilidades

---

Entrada: *sol*, *n*, *denominador*

Saída: *matProb*[][]

```
1 início
2   para i = 1 .. n faça
3     total ← 0
4     para j = 1 .. n faça
5       matProb[sol][i][j] ← (denominadordistancia(i,j))-1
6       // Essa distância varia de acordo com o problema. Por exemplo, no
        // PSUMMA a distância entre o primeiro e o último elemento de uma
        // solução é n, enquanto no PCV, essa distância é 1.
7       total ← total + matProb[sol][i][j]
8     fim
9     //Normalização dos resultados
10    para j = 1 .. n faça
11      matProb[sol][i][j] ←  $\frac{\text{matProb}[\text{sol}][i][j] \times 100}{\text{total}}$ 
12    fim
13  fim
14 fim
```

---



# Procedimento Roll 'n' Improve

---

## Algoritmo 2: Roll 'n' Improve

---

```
Entrada: solAtual  
Saída: solOtimizada  
1 início  
2   exectype  $\leftarrow 1$   
3    $\varphi \leftarrow \infty$   
4   enquanto exectype  $\leq 2$  faça  
5     min  $\leftarrow \text{calculaFO}(\text{solAtual})$   
6     nIterSemMelhora  $\leftarrow 0$   
7     nIterConv  $\leftarrow 0$   
8     denominador  $\leftarrow \alpha$   
9     ultRes  $\leftarrow \infty$   
10    matProb  $\leftarrow \text{CriaMatrizDeProbabilidades}(\text{solAtual}, n, \text{denominador})$   
11    se exectype == 2 então  
12      InverteProbabilidades(matProb)  
13    fim  
14    repita  
15      LCR  $\leftarrow \text{CriaListaDeCandidatosRestritos}()$  // Nesse momento, são adicionados todos os  
16      elementos a tal lista  
17      solAuxiliar  $\leftarrow \emptyset$   
18      enquanto LCR  $\neq \emptyset$  faça  
19        elemenSort  $\leftarrow$  escolha aleatoriamente um elemento de LCR  
        LCR = LCR - {elemenSort}
```





# Procedimento Roll 'n' Improve

```
20      posSort ← sorteie, via roleta não determinística, uma posição através do vetor  
21      matProb[elemenSort] = elemenSort  
22      elimine a coluna posSort de matProb  
23  fim  
24  result ← calculaFO(solAuxiliar)  
25  se result < min então  
26      exectype ← 1  
27      min ← result  
28      solOtimizada ← solAuxiliar  
29      nIterSemMelhora ← 0  
30      denominador ←  $\alpha$   
31      matProb ← CriaMatrizDeProbabilidades(solOtimizada, n, denominador)  
32  fim  
33  senão  
34      nIterSemMelhora ← nIterSemMelhora + 1  
35  fim  
36  se (nIterSemMelhora ==  $\omega$ ) OR (result >  $\frac{\theta \times \min}{100}$ ) então  
37      denominador ← denominador +  $\kappa$   
38      matProb ← CriaMatrizDeProbabilidades(solOtimizada, n, denominador)
```



# Procedimento Roll 'n' Improve

```
39 |         se exectype == 2 então
40 |             | InverteProbabilidades(matProb)
41 |         fim
42 |     fim
43 |     se result ≠ ultRes então
44 |         | ultRes ← result
45 |         | nIterConv ← 0
46 |     fim
47 |     senão
48 |         | nIterConv ← nIterConv + 1
49 |     fim
50 | até (nIterConv >  $\beta$ ) OR (denominador >  $\varphi$ )
51 | se exectype == 1 então
52 |     |  $\varphi$  ← denominador
53 |     | // Isso implica que  $\varphi$  é igual ao denominador no qual a solução convergiu
54 |     | exectype ← exectype + 1
55 | fim
56 | return solOtimizada
57 fim
```



# Roll 'n' Improve

Sejam o vetor  $[2,1,3,5,4]$  a atual solução e o denominador igual a 2.

Uma nova solução ocorrerá da seguinte maneira:



# Matriz de Probabilidades

Solução: [2,1,3,5,4]

Solução gerada: [-,-,-,-,-]

## Pré normalização

		Posições				
		1	2	3	4	5
Elementos	1	50	100	50	25	12,5
	2	100	50	25	12,5	6,75
	3	25	50	100	50	25
	4	6,75	12,5	25	50	100
	5	12,5	25	50	100	50

# Matriz de Probabilidades

Solução corrente: [2,1,3,5,4]

## Pós normalização

		Posições				
		1	2	3	4	5
Elementos	1	21%	42%	21%	10,5%	5,75%
	2	51,5%	25,8%	12,9%	6,5%	3,2%
	3	10%	20%	40%	20%	10%
	4	3,2%	6,5%	12,9%	25,8%	51,5%
	5	5,75%	10,5%	21%	42%	21%

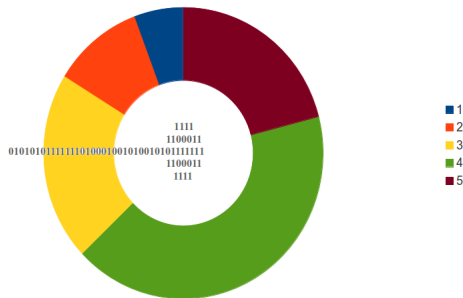
A partir de agora, selecione, em cada rodada, um elemento que ainda não pertença a seleção e rode a roleta para o mesmo.

Solução gerada:  $[-,-,-,-,-]$



## Elemento sorteado: 5

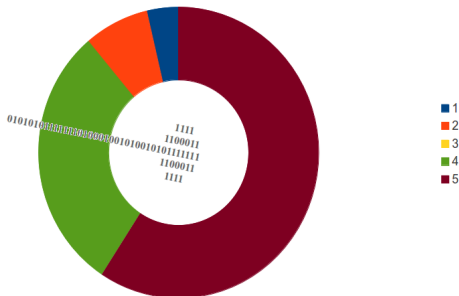
Solução corrente: [2,1,3,5,4]



Solução gerada: [-,-,5,-,-]

## Elemento sorteado: 4

Solução corrente: [2,1,3,5,4]

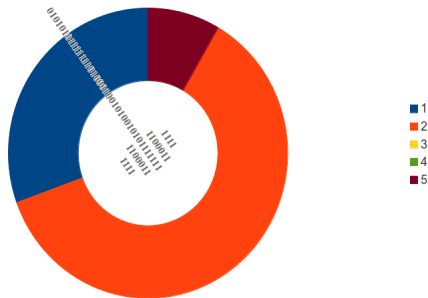


Solução gerada: [-,-,5,4,-]



## Elemento sorteado: 1

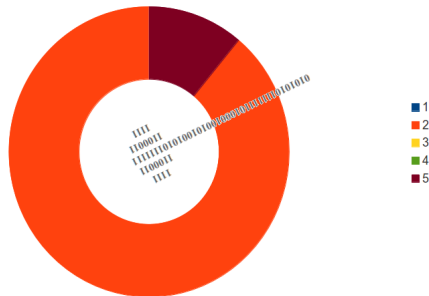
Solução corrente: [2,1,3,5,4]



Solução gerada: [1,-,5,4,-]

## Elemento sorteado: 2

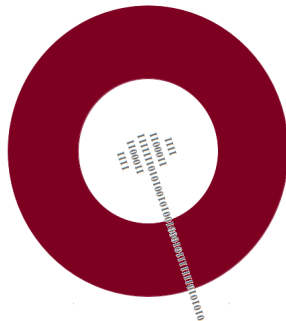
Solução corrente: [2,1,3,5,4]



Solução gerada: [1,2,5,4,-]

## Elemento sorteado: 3

Solução corrente: [2,1,3,5,4]



- 1
- 2
- 3
- 4
- 5

Solução gerada: [1,2,5,4,3]

# Algoritmo Proposto

---

## Algoritmo 3: AlgoritmoProposto

---

**Entrada:**  $n$

**Saída:**  $solOtimizada$

1 **início**

2      $solOtimizada \leftarrow constroiSolucaoAleatoria(n)$

3      $solOtimizada \leftarrow Nonggis(SolOtimizada)$

4      $solOtimizada \leftarrow BuscaTabu(SolOtimizada)$

5      $solOtimizada \leftarrow ReconexaoPorCaminhos(SolOtimizada)$

6     return  $solOtimizada$

7 **fim**

---



# Algoritmo Proposto

- Combinação de Roll 'n' Improve com Busca Tabu e Reconexão por caminhos;
- os demais precisam de boa solução inicial;
- procedimento nomeado ***Rnl-TS-PR***.



# Justificativa e Relevância

- 1 Importância em vários campos da ciência;
- 2 aspecto financeiro relacionado;
- 3 equilíbrio da carga de trabalho;
- 4 muitas metaheurísticas requerem boa solução inicial.



# Sumário

- 1 Introdução
- 2 O Problema de Sequenciamento
- 3 Metodologia
- 4 Resultados**
- 5 Conclusões e Trabalhos Futuros



# Resultados

- Algoritmo desenvolvido com a linguagem C++, utilizando-se o IDE *NetBeans* 7.0.1, o compilador GCC 4.6.1, e o sistema operacional Ubuntu 11.10;





# Resultados: Algoritmos Heurísticos

- Melhora das soluções:

$$imp_i^{best} = \frac{f_i^* - f_i^{RnITSPR^*}}{f_i^*}$$

- Desvio médio das soluções finais:

$$gap_i^{avg} = \frac{\bar{f}_i^{RnITSPR} - f_i^*}{f_i^*}$$



## Resultados: Algoritmos Heurísticos

- Melhora das soluções:

$$imp_i^{best} = \frac{f_i^* - f_i^{RnITSPR^*}}{f_i^*}$$

- Desvio médio das soluções finais:

$$gap_i^{avg} = \frac{\bar{f}_i^{RnITSPR} - f_i^*}{f_i^*}$$



# Resultados

Os resultados estão em mãos, porém as tabelas estão sendo geradas. De modo que até a hora da apresentação as mesmas serão exibidas neste espaço.



# Sumário

- 1 Introdução
- 2 O Problema de Sequenciamento
- 3 Metodologia
- 4 Resultados
- 5 Conclusões e Trabalhos Futuros
  - Cronograma de atividades – BCC391 - Monografia II



# Conclusões

- Uma nova metaheurística foi desenvolvida;
- essa metaheurística se mostra capaz de gerar soluções de qualidade em um curto período de tempo;
- as soluções geradas são melhores que as da estratégia gulosa;
- soluções boas e heterogêneas são geradas no procedimento;
- redução do tempo de execução e soluções pouco piores que o GRASP TS PR;



# Conclusões

- Uma nova metaheurística foi desenvolvida;
- essa metaheurística se mostra capaz de gerar soluções de qualidade em um curto período de tempo;
- as soluções geradas são melhores que as da estratégia gulosa;
- soluções boas e heterogêneas são geradas no procedimento;
- redução do tempo de execução e soluções pouco piores que o GRASP TS PR;



# Conclusões

- Uma nova metaheurística foi desenvolvida;
- essa metaheurística se mostra capaz de gerar soluções de qualidade em um curto período de tempo;
- as soluções geradas são melhores que as da estratégia gulosa;
- soluções boas e heterogêneas são geradas no procedimento;
- redução do tempo de execução e soluções pouco piores que o GRASP TS PR;



# Conclusões

- Uma nova metaheurística foi desenvolvida;
- essa metaheurística se mostra capaz de gerar soluções de qualidade em um curto período de tempo;
- as soluções geradas são melhores que as da estratégia gulosa;
- soluções boas e heterogêneas são geradas no procedimento;
- redução do tempo de execução e soluções pouco piores que o GRASP TS PR;





# Conclusões

- Uma nova metaheurística foi desenvolvida;
- essa metaheurística se mostra capaz de gerar soluções de qualidade em um curto período de tempo;
- as soluções geradas são melhores que as da estratégia gulosa;
- soluções boas e heterogêneas são geradas no procedimento;
- redução do tempo de execução e soluções pouco piores que o GRASP TS PR;



## Trabalhos Futuros

- Experimentar remover a fase Busca Tabu;
- testar em outros problemas (próximo: TSP);
- realizar experimentos com outros parâmetros;
- testá-la como uma opção de otimização de boas soluções;
- paralelizar o algoritmo;
- estudar a inclusão de uma busca local no decorrer ou após o procedimento;
- realizar testes de probabilidade empírica.



## Trabalhos Futuros

- Experimentar remover a fase Busca Tabu;
- testar em outros problemas (próximo: TSP);
- realizar experimentos com outros parâmetros;
- testá-la como uma opção de otimização de boas soluções;
- paralelizar o algoritmo;
- estudar a inclusão de uma busca local no decorrer ou após o procedimento;
- realizar testes de probabilidade empírica.



# Trabalhos Futuros

- Experimentar remover a fase Busca Tabu;
- testar em outros problemas (próximo: TSP);
- realizar experimentos com outros parâmetros;
- testá-la como uma opção de otimização de boas soluções;
- paralelizar o algoritmo;
- estudar a inclusão de uma busca local no decorrer ou após o procedimento;
- realizar testes de probabilidade empírica.



## Trabalhos Futuros

- Experimentar remover a fase Busca Tabu;
- testar em outros problemas (próximo: TSP);
- realizar experimentos com outros parâmetros;
- testá-la como uma opção de otimização de boas soluções;
- paralelizar o algoritmo;
- estudar a inclusão de uma busca local no decorrer ou após o procedimento;
- realizar testes de probabilidade empírica.



## Trabalhos Futuros

- Experimentar remover a fase Busca Tabu;
- testar em outros problemas (próximo: TSP);
- realizar experimentos com outros parâmetros;
- testá-la como uma opção de otimização de boas soluções;
- paralelizar o algoritmo;
- estudar a inclusão de uma busca local no decorrer ou após o procedimento;
- realizar testes de probabilidade empírica.



## Trabalhos Futuros

- Experimentar remover a fase Busca Tabu;
- testar em outros problemas (próximo: TSP);
- realizar experimentos com outros parâmetros;
- testá-la como uma opção de otimização de boas soluções;
- paralelizar o algoritmo;
- estudar a inclusão de uma busca local no decorrer ou após o procedimento;
- realizar testes de probabilidade empírica.



## Trabalhos Futuros

- Experimentar remover a fase Busca Tabu;
- testar em outros problemas (próximo: TSP);
- realizar experimentos com outros parâmetros;
- testá-la como uma opção de otimização de boas soluções;
- paralelizar o algoritmo;
- estudar a inclusão de uma busca local no decorrer ou após o procedimento;
- realizar testes de probabilidade empírica.





## Cronograma de atividades – BCC391 - Monografia II

A Tabela 1 contém o cronograma de atividades proposto para a disciplina BCC391 em 2012/1.

<b>Atividades</b>	<b>Mar</b>	<b>Abr</b>	<b>Mai</b>	<b>Jun</b>	<b>Jul</b>
Revisão de literatura	X	X			
Implementação de formulações	X	X			
Testes	X	X	X		
Paralelização do Nonggis			X	X	
Redigir a Monografia			X	X	X
Apresentação do Trabalho				X	X

**Tabela:** Cronograma de Atividades para a Disciplina BCC391.



# Fim da Apresentação

Obrigado!

