Otimização Combinatória: Trabalho Prático 2025/1

Prof. Henrique Becker

10 de Março de 2025

1 Instruções gerais

1. O trabalho prático consiste em 4 entregáveis e uma apresentação.
2. Os entregáveis e a nota associada aos mesmos segue:
   (a) meta-heurística e relatório preliminar (1.2pt),
   (b) formulação inteira (0.3pt),
   (c) relatório final (0.5pt),
   (d) e material da apresentação (0.5pt em conjunto c/ a apresentação).
3. Os seguintes prazos devem ser observados:
   (a) a 'meta-heurística e relatório preliminar' deve ser entregue até às 23:59 de 2025-05-04 (domingo);
   (b) os demais *entregáveis* até às 23:59 de 2025-06-15 (domingo);
   (c) a apresentação ocorrerá dia 2025-06-17 (terça-feira) ou no dia 2025-06-24 (terça-feira).
4. Todas datas podem ser prorrogadas caso os alunos sejam notificados pelo fórum de avisos (e/ou e-mail) com pelo menos 72h de antecedência.
5. As aulas de apresentação terão chamada e contam presença.
6. Os alunos não saberão a ordem das apresentações de antemão; esta será divulgada grupo a grupo durante os dias de apresentação.

7. O relatório final pode incluir correções e/ou melhoramentos aplicados a meta-heurística (nesse caso o novo código deve ser disponibilizado na segunda fase de entrega), esses melhoramentos *podem* incrementar a nota do primeiro entregável em, no *máximo*, 3 décimos (caso *todos* pontos negativos indicado na correção seja resolvidos).

8. O trabalho será realizado idealmente por trios.

   (a) A formação dos trios fica a critério e responsabilidade dos alunos.

   (b) É responsabilidade de cada aluno formar um trio com outros alunos que irão contribuir com o trabalho tanto quanto ele próprio.

   (c) A mesma exata nota será compartilhada pelos 3 membros do trio, salvo circunstâncias excepcionais.

      i. Circunstâncias excepcionais incluem, mas não estão limitadas a, existir evidência esmagadora de que um aluno não contribuiu com absolutamente nada no trabalho.

   (d) Caso a quantidade de alunos da turma seja perfeitamente divisível por três, os últimos 3 alunos fora de quaisquer grupos serão automáticamente considerados um trio.

   (e) Caso a quantidade de alunos da turma seja tal que `num_alunos % 3 == 2`, os últimos 2 alunos fora de qualquer grupo serão automáticamente considerados um par.

   (f) Caso a quantidade de alunos da turma seja tal que `num_alunos % 3 == 1`, o último pedido de trio possível não será aceito, os quatro alunos restantes devem formar duas duplas.

9. O tópico do trabalho é aplicar uma meta-heurística sobre um problema.

   (a) A lista dos problemas pré-aprovados será disponibilizada em outro documento e somente após as aulas iniciais de meta-heurísticas.

   (b) A escolha do tópico é responsabilidade dos trios.

   (c) Não pode ser escolhido o mesmo tópico já escolhido por outro trio.

   (d) Escolher meta-heurística ou problema não pré-aprovada exige uma justificativa por parte to grupo e aprovação por parte do professor.

   (e) Assim que um trio decidir por um tópico (i.e., par de problema/meta-heurística), este deve ser imediatamente informado por e-mail para

o professor (hbecker@inf.ufrgs.br) com assunto '[OC TF] Escolha' e o número do cartão da UFRGS dos três envolvidos.

(f) Este e-mail será respondido pelo professor confirmando ou não a possibilidade (após verificar outros e-mails e atualizar a planilha).

10. O entregável **meta-heurística e relatório preliminar** consiste em:

(a) Um código-fonte e um relatório em formato PDF.

(b) O código-fonte pode estar em qualquer linguagem mas deve possuir instruções claras para compilação/execução em Linux.

(c) O 'programa' se refere a execução do código-fonte interpretado ou a execução do binário compilado a partir do código-fonte.

(d) O programa deve receber a sua entrada por meio de **parâmetros posicionais da linha de comando**[1] na *seguinte ordem*: (i) o caminho até o arquivo de entrada (incluíndo seu nome); (ii) um valor que controla o critério de parada escolhido (e.g., iterações por elemento do arquivo de entrada); (iii) a semente de aleatóriedade (se a heurística é não-deterministica) ou o parâmetro escolhido para variação (se a heurística é deterministica). A partir da quarta posição, ou por meio de *flags*/opções, os alunos podem passar quaisquer outros parâmetros *opcionais* que eles queiram.

(e) Após a geração da solução inicial, e toda a vez que o programa encontra uma solução melhor que a melhor conhecida até o momento, esta deve escrever na saída padrão: (i) a quantidade de segundos (com duas casas após a vírgula) desde o começo da execução; (ii) o valor dessa solução; (iii) uma representação da mesma que seja possível de ser compreendida por um ser humano. Além disso, a implementação também deve escrever na saída padrão quaisquer outras informções usadas para montagem das tabelas do relatório preliminar.

(f) O programa deve executar de forma determinística. Ou seja, dadas as exatas mesmas entradas, ele precisa retornar os mesmos resultados, independente da máquina utilizada. Dessa forma, o parâmetro indicando o critério de parada **NÃO PODE SER**

[1] Em C/C++ isso é feito por meio de `argc` e `argv`, em Python por `sys.argv`, em Julia por `ARGS`.

TEMPO. Os experimentos exigem rodar o código por "cerca de" 5s e 300s porém o recomendado é que o aluno codifique um controle que escreva no terminal o número da primeira iteração após 5s de execução (e o mesmo para 300s), e então adote esses números como parâmetos de critério de parada para os experimentos do trabalho.

(g) O código deve implementar a meta-heurística escolhida. Cada meta-heurística tem lacunas que variam com o problema e decisões de implementação que ficam a critério dos alunos, mas também tem partes que caracterizam ela como aquela meta-heurística (estas *precisam* estar presentes no código).

(h) São critérios de avaliação da implementação da meta-heurística:

   i. a sua corretude (e.g., não retorna soluções inválidas),
   ii. a sua legibilidade (e.g., organização, nomes, e comentários),
   iii. a adequação a meta-heurística escolhida (como explicado acima),
   iv. e a sua performance (especialmente no que tange a escolha de representação da solução e como esta é alterada, a exagerada criação novos objetos de solução ao invés de sua alteração, e o recálculo do valor da função objetivo do a partir zero ao invés de por delta).

(i) A versão do código-fonte entregue deve ser exatamente a mesma utilizada para a escrita do relatório preliminar entregue.

(j) O relatório preliminar deve possuir no máximo 4 páginas e no mínimo fonte de tamanho 12.

(k) O relatório preliminar deve possuir uma descrição clara e completa da implementação:

   i. Escolha de representação do problema.
   ii. Construção da solução inicial.
   iii. Principais estruturas de dados.
   iv. Vizinhança e a estratégia de escolha de vizinhos (quando aplicável).
   v. Processo de recombinação e factibilização (quando aplicável).
   vi. Parâmetros do método (valores usados nos experimentos).
   vii. Critério de parada (NÃO pode ser *diretamente* tempo).

(l) **Comparações preliminares:** Com a intenção de apresentar a versão da sua meta-heurística com os melhores resultados possíveis, é esperado que os alunos variem e testem diferentes valores para os diferentes parâmetros. Essa experimentação pode ser exploratória e não envolver um método sistemático mas, ao invés disso, uma breve justificativa deve ser dada para porque se escolheu os valores para cada parâmetro da meta-heurística. No relatório final, os parâmetros terão valores fixados baseado no que foi considerado melhor durante essa análise preliminar (com exceção dos métodos completamente determinísticos que ainda variarão o parâmetro considerado de maior impacto por esses experimentos dentro de um intervalo também justificado por esses experimentos).

(m) **Execuções mínimas:** Embora os experimentos sejam exploratórios, é necessário que o aluno apresente uma tabela com todos os parâmetros de entrada e o valor da função objetivo retornado para, pelo menos, uma run para cada instância (para permitir ao professor aferir que o código realmente funciona como esperado).

11. O entregável **formulação inteira** consiste em:

    (a) O código da formulação inteira segue as mesmas restrições dadas nos itens 10b até 10d exceto que: (i) o parâmetro do critério de parada é diretamente tempo[2]; e (ii) a semente de aleatoriedade é sempre utilizada e deve ser passada para o resolvedor[3].

    (b) O HiGHS é o solucionador recomendado mas outros solucionadores podem ser utilizados caso seja a preferência dos alunos.

    (c) Uma execução deve escrever na saída padrão as informações usadas para montar a tabela do relatório (melhor solução encontrada, tempo, etc...).

    (d) São critérios de avaliação da implementação da formulação inteira: a sua corretude, a sua legibilidade, o emprego do que aprendido na disciplina, e decisões que levam a formulação a ter mais ou menos variáveis e ser mais apertada ou mais frouxa.

[2]No caso de Julia/JuMP/HiGHS, o tempo limite do solucionador pode ser definida com `set_time_limit_sec(model, parse(Int, seconds))`

[3]No caso de Julia/JuMP/HiGHS, a semente de aleatoriedade do solucionador pode ser definida com `set_attribute(model, "random_seed", seed_parametro)`

(e) Essa formulação deve ser exatamente a mesma utilizada para escrita do entregável **relatório final**.

12. O entregável **relatório final** consiste em:

    (a) O texto *original* do relatório preliminar *e, se aplicável,* o que foi modificado em seção separada.

    (b) Descrição clara e completa da formulação inteira empregada.

    (c) **Execuções da formulação:** Para cada instância do problema, são necessárias pelo menos 10 execuções variando a semente de aleatoriedade de 1 até 10. Os valores apresentados podem ser médias dos valores soluções encontradas (e médias dos tempos de execução). Estes valores podem ser apresentados direto na tabela de comparação do item 12d ou em tabela própia (mas o grupo precisa comparar a performance e qualidade de formulação entre formulação e meta-heurística no texto). Usar limite de tempo de 300s (com critério de parada tempo mesmo, diferente da metaheurística).

    (d) **Execuções finais da meta-heurística** Os experimentos descritos nesse item devem ser repetidos para dois tempos de execução alvo (controlados por critério de parada determinístico): 5 segundos e 300 segundos (i.e., 5 minutos). Caso a meta-heurística seja determinística, escolher um parâmetro da meta-heurística para variar em 10 valores diferentes e comparar os resultados. Caso a meta-heurística não seja determinística, variar a semente de aleatoriedade (de 1 até 10 como com a formulação inteira). Logo, se são 2 limites de 'tempo', e 10 varições por instância, e cerca de 10 instâncias, tem de se apresentar os resultados de 200 execuções. Os aunos podem usar uma tabela resumo com médias para discussão porém é necessário (possivelmente em apêndice) uma tabela com as informações de cada execução de forma que o experimento seja reprodutível. Segue uma lista das colunas esperadas:

        i. Nome da instância (ou fazer uma tabela por instância).
        ii. Valor do parâmetro escolhido ou da semente de aleatoriedade.
        iii. Tempo de execução da meta-heurística (segundos): *H T (s).*
        iv. Valor da solução inicial da meta-heurística: S_i.
        v. Valor da melhor solução encontrada pela meta-heurística: S_h.

vi. Valor médio da solução encontrada pela formulação: S_f.

vii. Caso termine por limite de tempo, o limite superior[4]: U_f.

viii. Tempo médio de execução da formulação (segundos): *F T (s)*.

(e) Análise dos resultados – por exemplo: resultados variam muito com a semente de aleatoriedade? por que se acha que o valor encontrado teve melhores resultados? em que situações a formulação ganha da meta-heurística? (e vice-versa)

(f) Conclusões e bibliografia utilizada.

13. O entregável **apresentação** consiste em:

(a) O material da apresentação consiste em nos slides (se utilizados) e/ou transcrição e esboço do que será desenhado/escrito/falado.

(b) O material da apresentação entregue na segunda fase deve ser exatamente o mesmo utilizado na apresentação em si.

(c) Uma apresentação para turma de *no máximo* 4 minutos **por integrante**.

(d) A apresentação será interrompida se ultrapassar o limite de tempo.

(e) O objetivo da apresentação é apresentar as implementações e os resultados.

(f) Após o término da apresentação, o professor e os alunos tem 3 minutos para perguntas.

(g) Cada membro do grupo deve participar e comentar o que fez.

14. Os entregáveis devem ser enviados em pacotes respeitando as seguintes imposições:

(a) A entrega será via formulários no Moodle da disciplina.

(b) O pacote deve estar em um dos seguintes formatos: tar, tar.gz, tar.bz2, zip, rar, ou 7z.

(c) O nome do pacote da primeira entrega deve ser:
`inf05010_2024-2_<letra da turma>_TP_Grupo-<número>_FASE-1.<extensão>`

(d) O nome do pacote da segunda entrega deve ser:
`inf05010_2024-2_<letra da turma>_TP_Grupo-<número>_FASE-2.<extensão>`

[4]Em Julia/JuMP o limite superior é dado pela função `objective_bound`.

(e) O relatório e a apresentação de slides (exatamente a mesma a ser usada no dia da apresentação) devem estar em formato PDF, e terem nome, respectivamente: `relatorio.pdf` e `apresentacao.pdf`.

(f) O código da formulação inteira deve estar numa pasta chamada `formulation` que possui um arquivo README detalhando o processo de compilação/execução, o mesmo vale para o código da meta-heurística porém o nome da pasta deve ser `metaheuristic`.

(g) Caso, por qualquer motivo, o grupo **NÃO consiga submeter o seu trabalho pelo Moodle**, este deve imediatamente submeter o trabalho por e-mail para `hbecker@inf.ufrgs.br` e, caso este método também falhe (possivelmente por tamanho de arquivo), o aluno deve enviar o arquivo para uma plataforma online de armazenamento de dados (i.e., *cloud*) de sua escolha, que possua timestamp pública e confiável da data/horário de submissão/edição do arquivo, e então compartilhar um link para o arquivo (com as permissões de acesso adequadas) com o professor na primeira oportunidade possível (este arquivo não deve ser editado após a data de entrega). Ou seja, falhas na submissão pelo Moodle, ou por e-mail, não são justificativa para o trio não possuir um pacote com todo o trabalho prático que comprove que o mesmo foi produzido antes da data final de entrega do trabalho. As timestamps de arquivos dentro de um tar, zip, ou qualquer outro formato de compressão, são manipuláveis e não garantem que os mesmos realmente foram produzidos dentro do prazo.

(h) Caso, por qualquer motivo, o grupo **NÃO consiga finalizar um entregável até o prazo de entrega** É recomendado aos alunos que minimizem a porção do trabalho atrasada, isto é, procurem finalizar completamente um entregável antes de iniciar o próximo e, entreguem tudo que foi desenvolvido dentro do prazo antes do final do prazo indicando claramente o que está finalizado e o que ficou incompleto. Não é garantido que o que for entregue após o prazo de entrega será considerado e, caso seja, terá um fator de redução aplicado a nota. Entregar o que for possível dentro do prazo antes de considerar entregar uma versão mais completa após o prazo garante que *o que foi entregue dentro do prazo não será afetado pelo fator de redução da nota.*

Algumas dicas:

- Acerca da escolha de vizinhanças: uma boa vizinhança é aquela que permite alcançar qualquer solução *dadas suficientes iterações* (a vizinhança imediata de uma determinada solução não deve conter todas as possíveis soluções).
- No caso de vizinhos com mesmo valor de solução, utilizar escolhas aleatórias como critério de desempate podem trazer bons resultados.

2 Meta-heurísticas

As metaheurísticas pré-aprovadas disponibilizadas para os grupos seguem abaixo. Devido a similaridade, algoritmos genéticos e meméticos contam como a mesma opção (e, consequentemente, um grupo não pode pegar genético para um problema e outro grupo pegar memético para o mesmo problema). Os slides da disciplina também são uma referência válida para o desenvolvimento do trabalho, porém, a fim de fornecer uma base mais sólida para cada meta-heurística, é fornecida também um texto introdutório sobre cada meta-heurística (vide abaixo).

1. Late Acceptance Hill Climbing: 'BURKE, Edmund K.; BYKOV, Yuri. The late acceptance Hill-Climbing heuristic. European Journal of Operational Research, v. 258, n. 1, p. 70–78, 2017.' (`https://doi.org/10.1016/j.ejor.2016.07.012`)
2. Tabu Search: 'Tabu Search: A Tutorial, by Fred Glover (1990), Interfaces.' (`https://www.ida.liu.se/~zebpe83/heuristic/papers/TS_tutorial.pdf`).
3. Iterated Local Search: 'Helena Ramalhinho Lourenço; Olivier Martin; Thomas Stützle. A beginner's introduction to iterated local search. In: Proceeding of the 4th Metaheuristics International Conference. Porto, Portugal: [s.n.], 2001.' (`http://84.89.132.1/~ramalhin/PDFfiles/2001_MIC_FILS.pdf`)
4. GRASP: 'T.A. Feo, M.G.C. Resende, and S.H. Smith, A greedy randomized adaptive search procedure for maximum independent set, Operations Research, vol. 42, pp. 860-878, 1994' (`http://mauricio.resende.info/doc/gmis.pdf`).

5. Simulated Annealing: 'Optimization by simulated annealing: An experimental evaluation; part I, graph partitioning, by D.S. Johnson, C.R. Aragon, L.A. McGeoch, C. Schevon, Operations research 37 (6), 865-892, 1989.' (https://pubsonline.informs.org/doi/epdf/10.1287/opre.37.6.865)

6. VNS: 'A Tutorial on Variable Neighborhood Search, by Pierre Hansen (GERAD and HEC Montreal) and Nenad Mladenovic (GERAD and Mathematical Institute, SANU, Belgrade), 2003.' (http://www.cs.uleth.ca/~benkoczi/OR/read/vns-tutorial.pdf)

7. Genetic Algorithm: 'A genetic algorithm tutorial, by D. Whitley, Statistics and computing 4 (2), 65-85.' (http://link.springer.com/content/pdf/10.1007%252FBF00175354.pdf)