Instituto de Computação/Unicamp - Projeto e Análise de Algoritmos II - MC558A

Prof. Lehilton Lelis Chaves Pedrosa

Programa de Ensino

Este é o plano de desenvolvimento da disciplina. Leia-o com atenção e consulte este documento durante todo o semestre. Também, sempre acompanhe os avisos na página da disciplina: https://www.ic.unicamp.br/~lehilton/mc558a/.

Objetivos

Ao final do curso, o aluno deverá ser capaz de:

- modelar, desenvolver e analisar algoritmos para problemas em grafos aplicando subproblemas clássicos: buscas, caminhos mínimos, árvore geradora mínima;
- resolver problemas (atribuição, otimização, etc.) via redução para problemas de fluxo em redes e de programação linear.

Atividades

Aula expositiva: Durante as aulas, os alunos deverão participar levantando dúvidas, sugestões ou resolvendo problemas solicitados.

Exercícios de fixação: Em cada unidade, serão propostas listas de exercícios de fixação (do livro-texto e outras fontes) para serem feitos em casa. O conteúdo das listas é considerado parte integrante do curso e os alunos deverão resolver os exercícios para se prepararem para as avaliações. Recomenda-se tentar fazer as atividades individualmente e, só então, discutir em grupo. As principais dúvidas devem ser levantadas no **início** das aulas, ou com o monitor.

Teste individual (avaliação): Serão realizados 7 testes de unidade **individuais e sem consulta** com duração de 20 a 40 min (especificado em cada teste), no horário das aulas, em datas a serem divulgadas na página da disciplina. A nota de cada teste será de 0 a 10.

Exercícios (avaliação): Serão realizadas 7 aulas de exercícios, em datas a serem divulgadas na página da disciplina. Nessas aulas os alunos deverão resolver os exercícios propostos e deverão submeter a resolução, no prazo de 24 h. A nota do trabalho corresponderá a **um único exercício**, que será sorteado **somente depois** do prazo de entrega e valerá de 0 a 10. A dinâmica será:

1. durante a primeira parte da aula, os alunos devem tentar resolver os exercícios (se preferirem, poderão discutir em duplas ou trios, **desde que não repitam** um/a parceiro/a anterior); na segunda parte da aula, o professor (ou um aluno voluntário com auxílio do professor) apresentará uma ideia/sugestão de resolução para a turma;
2. cada aluno deverá escrever as resoluções de todas as questões (individualmente, sem auxílio de colegas) e entregá-las até 24 h depois do início da aula, **manuscritas** e digitalizadas no formato PDF por meio do sistema SUSY (trabalhos copiados, desorganizados, incompletos, ilegíveis ou rasurados terão nota zerada).

observação: para otimizar o tempo e exercitar a escrita formal, os trabalhos entregues devem ser escritos à mão; utilize um escâner de mesa ou um aplicativo de celular com a função scan para criar um PDF.

Exercícios de programação (avaliação): serão realizados 6 trabalhos de programação, a serem submetidos no SUSY, com prazos de uma semana (incluindo eventuais falhas do sistema) divulgados na página, valendo de 0 a 10. Os critérios de correção serão especificados em cada tarefa.

Avaliação

Média: Serão calculadas as médias aritméticas dos testes individuais (T), dos exercícios (E) e dos exercícios de programação (P). Depois será calculada a média aritmética do semestre (M) entre T, E e P. Será aprovado com nota de aproveitamento A := M o aluno que satisfizer todos os critérios abaixo**:**

1. 75% de frequência;
2. T ≥ 6; E ≥ 6; P ≥ 6.

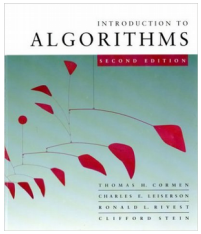
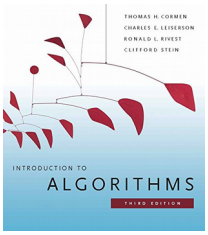
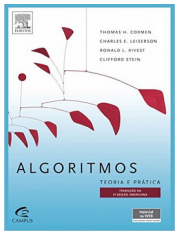
Do contrário, a nota de aproveitamento do semestre será A := *mínimo {4, M}*.

Exame: Poderá realizar exame, com nota E, entre 0 e 10, no dia 11/7/17, o aluno com 75% de frequência e nota A ≥ 2,5. Se o aluno fizer o exame, a nota final será *mínimo {5, (A+E)/2},* senão a nota final será *A*.

Fraude: Em caso de fraude (plágio, uso de bibliotecas não permitidas, cola, etc.), os envolvidos serão reprovados com nota 0.

Bibliografia

Será adotado como livro-texto: “Algoritmos – Teoria e Prática”, de T. Cormen, C. Leiserson, R. Rivest, C. Stein (CLRS). Poderá ser usada a segunda ou a terceira edição, tanto na versão em inglês quanto na versão traduzida. Por causa das diferentes numerações entre as edições, os exercícios solicitados do livro poderão ser transcritos.

	inglês	português
2ª edição	[1] 	[2] 
3ª edição	[3] 	[4] 

[1] T. Cormen, C. Leiserson, R. Rivest, C. Stein. Introduction to Algorithms (2ª edição), 2001.
[2] T. Cormen, C. Leiserson, R. Rivest, C. Stein. Algoritmos – Teoria e Prática (2ª edição), 2002.
[3] T. Cormen, C. Leiserson, R. Rivest, C. Stein. Introduction to Algorithms (3ª edição), 2009.
[4] T. Cormen, C. Leiserson, R. Rivest, C. Stein. Algoritmos – Teoria e Prática (3ª edição), 2012.

Além desses, poderão ser consultados outros livros, disponíveis na biblioteca, como o livro clássico de Donald Knuth, que é considerado o pai da análise de algoritmo.

[5] D. E. Knuth. The Art of Computer Programming, 1974

O livro de Manber é muito usado em cursos de algoritmos.

[6] U. Manber, Algorithms. A Creative Approach, 1989.

Há alguns livros pouco mais recentes, como o livro do Kleinberg e da Tardos.
[7] J. Kleinberg, E. Tardos. Algorithm Design, 2005.

Livros de autores brasileiros importantes também estão disponíveis.
[8] J. L. Szwarcfiter. Grafos e Algoritmos Computacionais, 1984.
[9] N. Ziviani. Projeto de Algoritmos (2ª edição), 2004.

Alguns assuntos, como modelos computacionais, são mais profundamente tratados em capítulos específicos de alguns livros, como sobre **reduções**
[10] M. Sipser. Introduction to the Theory of Computation (3ª edição), 2012.
[11] P. J. de Rezende, J. Stolfi. Fundamentos de geometria computacional, 1994. Disponível em: https://www.ic.unicamp.br/~rezende/rez-sto-94-fgc.pdf.

e sobre **programação linear e fluxo**.
[12] M. Bazaraa, J. Jarvis, H. Sherali. Linear Programming and Network Flows (4ª edição), 2009.
[13] L. A. Wolsey. Integer Programming, Wiley (1998).
[14] R. K. Ahuja, T. L. Magnanti, J. B. Orlin. Network flows: theory, algorithms, and applications, 1993.

Por último, mas não menos importante, na internet há diversos e excelentes recursos que podem servir para pesquisa. Em particular, veja o curso de análise de algoritmos do prof. Paulo Feofiloff e (com cuidado) a Wikipédia.
[15] P. Feofiloff. Análise de Algoritmos. https://www.ime.usp.br/~pf/analise_de_algoritmos/.
[16] Wikipédia. https://en.wikipedia.org/wiki/List_of_algorithms#Graph_algorithms.

Sugestão: não tente ler várias referências de uma vez; na maior parte do conteúdo, atenha-se ao livro-texto ([1]-[4]); para reduções, recomenda-se também ler o primeiro capítulo de [11]; para fluxos e programação linear, o conteúdo não excederá aos dos capítulos introdutórios de [12]-[14], escolha um livro de preferência e/ou leia os verbetes na Wikipédia. Consulte a bibliografia complementar sempre que solicitado, para se aprofundar, ou se tiver dificuldade com algum assunto do livro-texto e quiser uma apresentação diferente.

Rotina de estudo

Importante: Note que não há provas final e de meio de semestre. Portanto é fundamental criar uma rotina de estudos contínua. Planeje-se e separe algumas horas e alguns dias por semana para ler o livro, resolver as listas de exercícios e desenvolver as atividades de programação solicitadas. Os alunos (que preferirem) também são encorajados a se reunir e estudar em grupo (sempre depois de tentar resolver os exercícios individualmente e certificando-se de não copiar qualquer trecho de código nos programas).

Cada aluno tem sua própria maneira de estudar. Não obstante, algumas sugestões são úteis para o bom desenvolvimento da disciplina:

1. Leia o capítulo ou seção do livro correspondente ao conteúdo **antes** da aula correspondente (veja o calendário previsto abaixo); utilize a aula principalmente para tirar dúvidas e confirmar o seu entendimento.
2. Faça ou tente fazer os exercícios correspondentes a uma aula no próximo horário que tiver reservado para estudar a disciplina; **anote as principais dificuldades** e discuta com colegas, monitor ou, persistindo, com o professor no início das próximas aulas.
3. **Veja o resultado das avaliações!** Elas servem para que você identifique os problemas (erro de lógica, incorreção, conceitos incorretos, dificuldade com formalismo, incompletude, erros de português e escrita matemática, etc.). Tente refazer as tarefas e, se não puder identificar o que está errado ou não conseguir corrigir o problema, anote a dúvida e leve-a ao monitor ou ao professor (mas evite pedir aumento de nota, ou comparar notas de colegas).

Material didático

Os slides usados serão disponibilizados. A maioria dos slides e exercícios adicionais foram ou criados e gentilmente cedidos pelo prof. Cid Carvalho de Souza e pela profa. Cândida Nunes da Silva (particularmente com modificações do prof. Orlando Lee); ou criados e gentilmente cedidos pelo prof. Flávio Keidi Miyazawa. Eu adaptei o material disponibilizado e possivelmente introduzi erros, que podem ser reportados a mim.

Organização do curso

O curso tem 7 unidades. As datas abaixo são apenas uma tentativa e servem para que o aluno se planeje e estude para as aulas. Datas, ordem e conteúdo podem variar a depender do andamento. **Sempre consulte também os avisos na página da disciplina!**

i - Grafos
- 2/3/2017: Aula 1 - Conceitos de grafos
- 7/3/2017: Aula 2 - Fatos básicos de grafos
- 9/3/2017: Aula 3 - Representação de grafos
- 14/3/2017: ***Aula de exercícios 1***

ii - Buscas em grafos
- 16/3/2017: Aula 4 - Busca em largura
- 21/3/2017: Aula 5 - Busca em profundidade
- 23/3/2017: Aula 6 - Ordenação topológica
- 23/3/2017: ***Teste de unidade individual 1***
- 25/3/2017: ***Divulgação do exercício de programação 1*** (entrega até 2/4/2017)
- 28/3/2017: Aula 7 - Componentes fortemente conexas
- 30/3/2017: ***Aula de exercícios 2***

iii - Caminhos mínimos
- 4/4/2017: Aula 8 - Caminhos mínimos com uma origem
- 6/4/2017: Aula 9 - Algoritmo de Dijkstra
- 8/4/2017: ***Divulgação do exercício de programação 2*** (entrega até 16/4/2017)
- 11/4/2017: Aula 10 - Algoritmo de Bellman-Ford
- 11/4/2017: ***Teste de unidade individual 2***
- 18/4/2017: Aula 11 - Caminhos mínimos entre todos os pares de vértices
- 20/4/2017: ***Aula de exercícios 3***

iv - Árvore geradora mínima
- 25/4/2017: Aula 12 - Árvore geradora mínima
- 27/4/2017: Aula 13 - Algoritmo de Kruskal e conjuntos disjuntos
- 27/4/2017: ***Teste de unidade individual 3***
- 29/4/2017: ***Divulgação do exercício de programação 3*** (entrega até 7/5/2017)
- 2/5/2017: Aula 14 - Conjuntos disjuntos com florestas disjuntas
- 4/5/2017: ***Aula de exercícios 4***

v - Reduções entre problemas
- 9/5/2017: Aula 15 - Conceitos de redução entre problemas
- 11/5/2017: Aula 16 - Reduções para obtenção de cota inferior
- 11/5/2017: ***Teste de unidade individual 4***
- 13/5/2017: ***Divulgação do exercício de programação 4*** (entrega até 21/5/2017)
- 16/5/2017: Aula 17 - Exemplos de reduções
- 18/5/2017: ***Aula de exercícios 5***

vi - Fluxo em redes
- 23/5/2017: Aula 18 - Fluxos
- 25/5/2017: Aula 19 - Teorema do fluxo máximo e corte mínimo
- 27/5/2017: ***Divulgação do exercício de programação 5*** (entrega até 4/6/2017)
- 30/5/2017: Aula 20 - Algoritmos especializados de fluxo
- 30/5/2017: ***Teste de unidade individual 5***
- 1/6/2017: Aula 21 - Aplicações de fluxo
- 6/6/2017: ***Aula de exercícios 6***

vii - Programação Linear
- 8/6/2017: Aula 22 - Programação Linear
- 10/6/2017: ***Divulgação do exercício de programação 6*** (entrega até 18/6/2017)
- 13/6/2017: Aula 23 - Problema de fluxo de custo mínimo
- 13/6/2017: ***Teste de unidade individual 6***
- 20/6/2017: Aula 24 - Programação linear inteira
- 20/6/2017: ***Teste de unidade individual 7***
- 22/6/2017: ***Aula de exercícios 7***