Universidade Federal de Minas Gerais
Programa de Pós-Graduação em Ciência da Computação
Exame de Qualificação 1º Estágio
2º Semestre de 2019

Em 23/08/2019, **09:00 horas**.

Prova individual sem consulta com duração de **3 horas**.

Observações:

1. A prova deve ser resolvida no próprio caderno de questões.
2. As questões desta prova estão nas páginas seguintes, as quais estão numeradas de 1 a 9.
3. Faz parte da prova a interpretação das questões. Caso você ache que falta algum detalhe nos enunciados ou nos esclarecimentos, você deverá fazer as suposições que achar necessárias e escrever essas suposições juntamente com as respostas.
4. **Todas** as respostas devem ser justificadas.
5. Somente serão corrigidas respostas legíveis.
6. Não se esqueça de escrever seu **nome completo em todas as páginas**.

Desejamos a você uma boa prova!

A COPEQ

Atenção: Esta prova contém um total de 6 (seis) questões, das quais você deve fazer 4 (quatro). Marque abaixo as questões que devem ser consideradas para avaliação:

1 2 3 4 5 6 (selecione até quatro)

Nome completo:_____

Assinatura: _____

Questão 1

Em cada caso abaixo, uma função $T(n)$ é apresentada de forma recursiva. Encontre, em cada caso, uma função $h(n)$, apresentada de forma não recursiva, tal que $T(n) \in \Theta(h(n))$. Você precisa justificar sua resposta de algum modo (demonstração por indução, ou citação correta ao Teorema Mestre, ou desenho de árvore e análise de seu custo, etc.). Assuma, em todos os casos, que $T(1)$ é uma constante positiva.

(a) $T(n) = 4T(\lfloor n/2 \rfloor) + n^2$.

(b) $T(n) = 3T(n-1)$.

(c) $T(n) = T(\lfloor n/3 \rfloor) + T(\lceil 2n/3 \rceil) + n$.

Solução:

(a) *De acordo com o Teorema Mestre, se* $T(n) = aT(n/b) + f(n)$ *é tal que* $f(n) \in \Theta(n^{\log_b a})$, *então* $T(n) \in \Theta(n^{\log_b a} \log n)$. *Como* $f(n) = n^2$ *e* $\log_b a = 2$, *tem-se, portanto, neste caso, que* $T(n) \in \Theta(n^2 \log n)$.

(b) $T(n) \in \Theta(3^n)$. *Para tal, seja* $T(1) = c > 0$, *e assuma como hipótese indutiva que* $T(n) = c3^n$. *Temos que* $T(n+1) = 3T(n)$, *por definição, e então, por indução,* $T(n+1) = 3c3^n = c3^{n+1}$. *Logo* $T(n) \in \Theta(3^n)$.

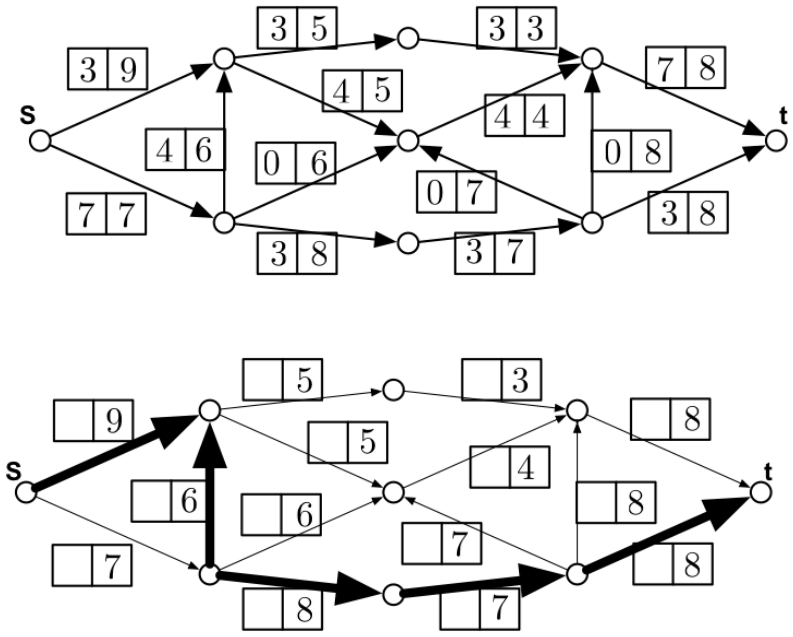
(c) *Neste caso, deve-se esboçar uma árvore de recorrência. Os primeiros* $\log_3 n$ *níveis serão completos de custo constante* $= n$. *Os demais níveis terão custo menor. Há um total de* $\log_{3/2} n$ *níveis. Portanto* $T(n) \in \Omega(n \log_3 n)$ *e* $\in O(n \log_{3/2} n)$. *Portanto* $T(n) \in \Theta(n \log n)$.

Questão 2

O método de Ford-Fulkerson é utilizado para encontrar um fluxo máximo de um vértice s para um vértice t em um grafo dirigido, com capacidades nos arcos. Ele consiste basicamente em encontrar um caminho de s para t e realizar algum procedimento que aumenta, se possível, o fluxo total que flui de s para t.

No grafo abaixo, o par de números em cada arco representa | `fluxo atual` | `capacidade` | de uma instância do método. O algoritmo de busca utilizado encontrou então o caminho em negrito no segundo grafo.

(a) Escreva em **todos os espaços em branco** os fluxos obtidos após uma iteração do método de Ford-Fulkerson que usa o caminho em negrito, baseado no estado abaixo.

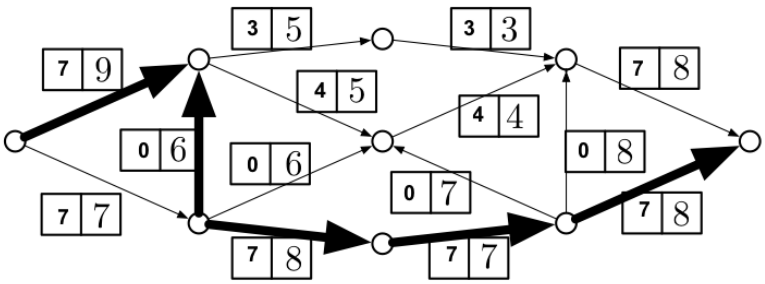


(b) Qual o valor máximo de um st-fluxo neste grafo?

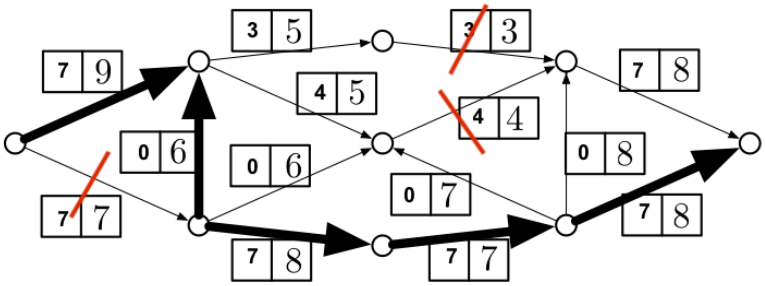
Um conjunto de arcos cuja remoção elimina todos os caminhos dirigidos de s para t é chamado de um corte. Após o término do método, é possível identificar um corte cuja soma das capacidades dos arcos no corte é igual ao fluxo máximo encontrado.

(c) Marque no primeiro grafo os arcos de um corte com mesmo valor do fluxo máximo.

Solução:



Fluxo máximo tem valor 14 (o encontrado já é máximo). O corte está abaixo



Questão 3

O método geral para encontrar uma árvore geradora mínima em um grafo conexo consiste em adicionar arestas de custo minimal desde que elas não criem ciclos. Dentre as várias estratégias possíveis, duas se destacam: o algoritmo de Kruskal e o algoritmo de Prim.

(a) Descreva, em linhas gerais, a diferença entre esses dois algoritmos.

(b) Julgue as afirmativas abaixo como verdadeiro ou falso. Suponha que G é um grafo não-direcionado, conexo, e que todas as arestas tem pesos distintos. Para cada alternativa, escreva uma breve justificativa.

(i) A aresta de peso mínimo está presente em todas as árvores geradores de custo mínimo.

(ii) Se a aresta de custo máximo pertence a uma árvore geradora mínima, então a sua remoção desconectaria o grafo G.

(iii) Para cada aresta cujo custo não é o máximo nem o mínimo, há pelo menos uma árvore geradora mínima que não a contém, e outra que a contém.

(iv) A aresta de custo máximo nunca será escolhida pelo algoritmo de Prim.

(v) Se G possui $|V|$ vértices, é possível implementar tanto o algoritmo de Kruskal como o de Prim com custo $O(|V|^2)$.

Solução:

Em Kruskal, arestas de custo mínimo são adicionadas desde que não criem ciclos, até que $n-1$ *tenham sido adicionadas. Em Prim, arestas de custo minimal são adicionadas de modo que uma árvore cresça a partir de um vértice até que seja geradora do grafo.*

Em ordem, VVFFF.

Questão 4

No problema dos baldes, você deve, utilizando dois baldes, conseguir medir exatamente um determinado volume de água. Na entrada deste problema, são dados três inteiros a, b e n, onde a e b correspondem às capacidades, em litros, de dois baldes, e n é a quantidade de água que se deseja obter.

Para obter a quantidade desejada, pode-se utilizar qualquer combinação das seguintes operações, incluindo repetições:

1. Encher totalmente um balde em uma torneira;
2. Esvaziar totalmente um balde;
3. Entornar o conteúdo de um balde em outro até que o balde de origem esvazie, ou que o balde de destino encha, o que ocorrer primeiro.

Como exemplo, considere a entrada $a = 3$, $b = 5$ e $c = 4$. Neste caso, a quantidade de 4 litros poderia ser obtida da seguinte maneira:

- Encha o balde de capacidade 5 (operação 1);
- Entorne o conteúdo do balde de capacidade 5 no balde de capacidade 3 (operação 3);
- Esvazie o balde de capacidade 3 (operação 2).
- Entorne o conteúdo do balde de capacidade 5 no balde de capacidade 3 (operação 3);
- Encha o balde de capacidade 5 (operação 1);
- Entorne o conteúdo do balde de capacidade 5 no balde de capacidade 3 (operação 3).

Note que, após a última operação, o balde de capacidade 5 terá a quantidade desejada de água.

(a) Escreva um algoritmo para resolver este problema, isto é, dados a, b e n, determine se é possível obter a quantidade n em uma jarra.

(b) Analise a complexidade de pior caso de seu algoritmo, fornecendo um limite assintótico em função de a, b e n.

(c) Dizemos que um problema de decisão pertence a P se existe um algoritmo polinomial que o resolve. É possível afirmar que seu algoritmo é evidência suficiente para mostrar que o problema das jarras pertence a P?

Solução:

(a) O algoritmo mais simples pode ser vista como uma busca em largura em um grafo implícito, com um vértice para cada estado possível. O estado pode ser representado pelo par (x, y), onde x e y correspondem às quantidades de água nos baldes de a e b respectivamente. As arestas correspondem às possíveis aplicações de cada uma das 3 operações a cada um dos baldes. Basta, então, verificar se um estado (n, i) ou (i, n) pode ser alcançado, para algum i.

(b) Como o número máximo de vértices do grafo implícito é ab, e cada vértice possui no máximo 6 arestas saindo, a busca em largura terá um número de passos $O(ab)$.

(c) Não. Neste caso, $O(ab)$ não é um polinômio no tamanho da entrada, pois a entrada é $\Theta(\log a + \log b + \log n)$.

Questão 5

Considere um vetor de $n \geq 2$ posições, cada uma preenchida com um dígito no intervalo 0-9. Queremos decompor este vetor em subvetores, ou seja, em trechos consecutivos. Além disso, todo subvetor deve ser par, isto é, o número formado pelos dígitos do subvetor deve ser par.

Por exemplo, caso o vetor de entrada seja 805438452, este pode ser particionado como "8054, 38, 452", com 3 subvetores, ou como "8, 0, 54384, 52", com 4 subvetores, dentre outras possibilidades. Escreva um algoritmo que determine o número de maneiras diferentes em que um em que o vetor pode ser decomposto. Por exemplo, caso o vetor de entrada seja 1234, o vetor pode ser decomposto de apenas duas maneiras: "12, 34" e "1234". Determine a complexidade de seu algoritmo. Você pode assumir que o último dígito do vetor é par.

Solução: Basta notar que a solução é sempre da forma 2^{k-1}, onde k é o número de dígitos pares do vetor. Isso segue do fato de que todo número par termina com um dígito par e, para cada dígito par (com exceção do último) podemos decidir se ali termina ou não um dos subvetores.

Contar o número k de dígitos pares do vetor pode ser feito em tempo linear, portanto a solução é linear.

Questão 6

No problema 3-SAT, é dada uma fórmula lógica na forma normal conjuntiva e deve-se determinar se é possível atribuir valores às suas variáveis de forma a tornar a fórmula verdadeira. Cada variável pode ocorrer positiva ou negativamente (negada) as cláusulas contêm, cada uma, até 3 literais.

(a) Mostre a NP-completude ou forneça um algoritmo polinomial para a versão do problema 3-SAT onde, na fórmula de entrada, cada variável ocorre sempre positiva ou sempre negativa.

(b) Mostre a NP-completude ou forneça um algoritmo polinomial para a versão do problema 3-SAT onde, na fórmula de entrada, cada literal ocorre no máximo uma vez.

Solução:

(a) Um algoritmo guloso é suficiente: se uma variável só ocorre positivamente, podemos atribuir o valor verdadeiro a ela, caso contrário, atribuímos o valor falso. Todos os literais serão verdadeiros e a fórmula satisfeita.

(b) Se cada literal só ocorre uma vez, uma variável poderá satisfazer no máximo uma cláusula. Resta, então, determinar, para cada cláusula, qual literal irá satisfazê-la. Desta forma, basta construir um grafo bipartido com um vértice para cada cláusula e cada variável, e uma aresta entre dois vértices se e somente se eles correspondem a uma variável x e a uma cláusula C tal que $x \in C$. Basta, então, determinar um emparelhamento máximo neste grafo e verificar se todo vértice correspondente a uma cláusula é incidente a uma aresta do emparelhamento.