2ª Lista de Exercícios de Teoria dos Grafos

André Couto

03/11/18

1 Considere um grafo bipartido $G = (X, Y, E)$. Prove que a cardinalidade máxima de um matching em G é igual à cardinalidade mínima de uma cobertura de vértices de G.

Começamos definindo um grafo bipartido $G(X, Y, E)$ como um grafo cujos vértices podem ser divididos em dois conjuntos disjuntos X e Y tais que toda aresta conecta um vértice em X a um vértice em Y; ou seja, X e Y são conjuntos independentes. A cardinalidade máxima de um matching desse grafo bipartido pode ser encontrando considerando para cada aresta um par de vértices. Desta forma, a cada aresta que vamos considerando, eliminamos uma outra aresta que usa qualquer um dos vértices desta aresta que acabamos de incluir no nosso matching, no final temos todos os vértices que possuem ligação separado em pares de tal forma que caso há um numero impar de vértices, o ultimo vértice não pertencera ao nosso matching. Neste caso vamos ter um matching máximo para o nosso $G(X, Y, E)$.

Desta forma, a cardinalidade do nosso matching máximo vai ser o numero de arestas que incluímos no nosso matching. Agora podemos partir para a questão da cobertura minima. Partimos a partir do nosso matching que temos um conjunto de arestas que ligam vértices de dois conjuntos diferentes. A cobertura nada mais é que um conjunto de vértices em que temos uma ligação com pelo menos uma extremidade de todo o conjunto de arestas do grafo. Como garantimos que temos um matching máximo e cada aresta está ligada a todos os vértices do conjunto do matching, então temos que podemos eliminar uma extremidade de cada aresta. Logo a cardinalidade minima de cobertura vai ser o numero de vértices que está contido no conjunto da cobertura que vemos que é o mesmo valor que o numero de arestas contido no nosso matching pois para cada par de vértices temos uma aresta e eliminamos um vértice de cada par.

Sabemos também que mesmo que exista uma aresta amais ligando um vértice que não entrou no conjunto de matching ele também sera representado na cobertura pois uma de suas extremidades está presente no matching.

2 Denote por $\chi(G)$ o número cromático de um grafo G. Prove que todo grafo G com m arestas satisfaz: $\chi(G) \leq \frac{1}{2}\sqrt{2m + \frac{1}{4}}$

Queremos provar que: $\chi(G) \leq \frac{1}{2}\sqrt{2m + \frac{1}{4}}$

Começamos analisando que o nosso $\chi(G) = k$ onde k define o menor numero de cores com qual conseguimos colorir os vértices do grafo G de tal forma que a coloração seja própria. Uma coloração é própria se todos os vértices que são adjacentes possuem cores diferentes. Queremos primeiro encontrar uma descrição para o numero de arestas em um grafo colorido G. Notamos que todo par de vértices possuem 2 cores e são ligados por uma aresta, ou seja, em cada extremidade da aresta temos um vértice com uma cor. Logo estabelecemos uma relação que para cada par de cores, vamos ter uma aresta que os liga. Notamos que um mesmo vértice com certa cor pode ser ligado a mais de uma aresta. Desta forma, vamos ter $\binom{k}{2}$ arestas.

Esse $\binom{k}{2}$ é uma combinação sem repetição, ou seja, vamos parear as cores sem que haja repetição de pares adjacentes já existentes.

Podemos usar a formula para uma combinação do tipo $\binom{n}{k}$ para calcular o valor minimo de arestas que existem no grafo. Temos que

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

Aplicando esta formula no nosso problema temos que

$$\binom{k}{2} = \frac{k!}{2!(n-2)!} = \frac{k(k-1)(k-2)...(1)}{2!(k-2)...(1)}$$

Podemos eliminar os termos em comum para finalizar que

$$\binom{k}{2} = \frac{k(k-1)}{2}$$

Notamos que isso é o meu valor minimo de arestas, ele é o valor de arestas apenas para estabelecer um par entre cada cor, porém sabemos que posso ter muito mais arestas (no caso em que temos mais de um vértice com mesma cor). Logo temos que

$$m \geq \frac{k(k-1)}{2}$$

$$\Rightarrow 2m \geq k^2 - k$$

$$\Rightarrow 2m + \frac{1}{4} \geq k^2 - k + \frac{1}{4}$$

$$\Rightarrow 2m + \frac{1}{4} \geq \left(k - \frac{1}{2}\right)^2$$

$$\Rightarrow \sqrt{2m + \frac{1}{4}} \geq k - \frac{1}{2}$$

$$\Rightarrow \sqrt{2m + \frac{1}{4}} + \frac{1}{2} \geq k$$

Voltamos substituindo $\chi(G) = k$ e provamos que

$$\chi(G) \leq \frac{1}{2}\sqrt{2m + \frac{1}{4}}$$

3 Sejam G e H dois grafos e f uma bijeção de V_G em V_H tal que $d_G(v) = d_H(f(v))$ para todo v em V_G. É verdade que $G \equiv H$? Justifique.

A afirmação é verdadeira pois estamos fazendo uma analise vértice a vértice. Temos que, se para todo o conjunto de vértices, o grau de cada v_G é igual ao grau de $_H$ onde $w_H = f(v_G)$ então estamos garantindo que para cada vértices, todos eles possuem as mesmas ligações. Caso o grafo não fosse isomorfo, em algum ponto que aplicamos a função bijetora sob o vértice v_H encontraríamos um grau diferente mostrando que existiria um vertice ou aresta a mais, isto é um absurdo, o que nos permitiria concluir que os grafos não são isomorfos. Portanto podemos concluir que existe pelo menos um mapeamento de vértices que torna o grafo G isomorfo a H.

4 Certo ou errado? Para mostrar que dois grafos G e H com mesmo número de vértices não são isomorfos basta exibir uma bijeção f de V_G em V_H e um par de vértices u e v em V_G tal que $(1)(u,v) \in E_G$ mas $(f(u), f(v)) \ni E_H$ ou $(2)(u,v) \ni E_G$ mas $f(u), f(v) \in E_H$. Justifique sua resposta.

Afirmativa errada. Mostrar que os grafos não são isomorfos por usar um função bijetora não nos garante que eles não são isomorfos, a única coisa que podemos afirmar com isto é que a nossa função bijetora utiliza um mapeamento errado para o grafo. Por exemplo na figura 2 temos que a nossa função bijetora é dada por

$$G_A = H_F$$

$$G_B = H_E$$

$$G_C = H_D$$

Caso usamos uma função bijetora errada dada por

$$G_A = H_E$$

$$G_B = H_D$$

$$G_C = H_F$$

Temos que aresta (B, C) existe em G porém (D, F) não existe logo temos que uma aresta que existe em um grafo não existe em outro, e fora que os graus do vértices são diferentes, logo, está função bijetora está errada, porém notamos que com a função bijetora correta, os grafos não deixam de ser isomorfos.

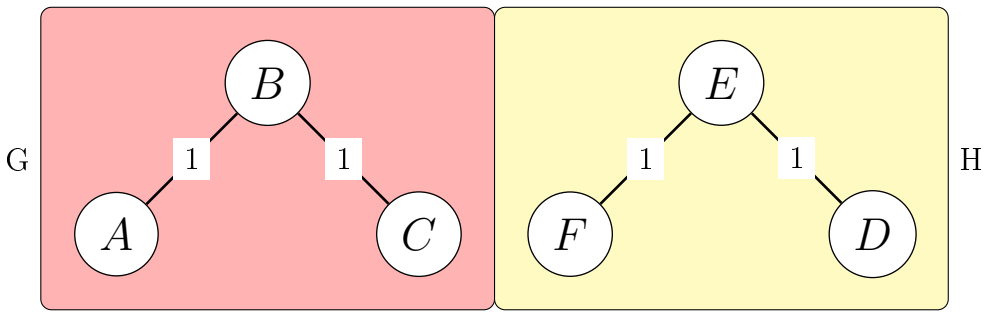


Figura 1: Grafo usado na questão 4

5 São dadas máquinas $1, ..., n$ e intervalos de tempo $I_1, ..., I_n$. Para cada i, um operador deve cuidar da máquina i durante o intervalo I_i. Se $I_i \cap I_j 6 \neq \phi$ um mesmo operador não pode cuidar de i e j. Qual o número mínimo de operadores suficiente para operar as máquinas? Apresente um exemplo com $n \geq 10$. Para o exemplo, mostre o grafo que modela o problema.

Podemos descrever este problema como um problema de coloração de grafos. Temos que é possível representar o conjunto de vértices com o conjunto de maquinas. Logo o nosso grafo vai possuir n vértices. Nós também vamos ter n intervalos de tempo. Queremos então mapear as maquinas que são executadas no mesmo intervalo de tempo. Podemos fazer isso, ligando cada vértice (máquina I_i) com todos os outros vértices(I_j) que ocorrem no mesmo intervalo de tempo (I_i. Se $I_i \cap I_j 6 \neq \phi$). Desta forma, vai existir arestas ligando todos os vértices que ocorrem no mesmo intervalo de tempo, e quando tentarmos colorir esse grafo, buscamos uma coloração com menor numero de cores possíveis.

Notamos então que cada cor equivale a um operário. Fica muito mais intuitivo mostrar essa relação através de um grafo. Vamos representar um conjunto de Máquina

$$M_0, M_1, M_2, M_3, M_4, M_5, M_6, M_7, M_8, M_9$$

onde as maquinas que devem ser operadas no mesmo intervalo de tempo são dadas por:

$$\begin{array}{c|c} M_9 \Rightarrow M_1, M_2 & M_1 \Rightarrow M_0, M_2, M_3 \\ M_2 \Rightarrow M_0, M_1 & M_3 \Rightarrow M_1 \\ M_4 \Rightarrow M_5 & M_5 \Rightarrow M_4 \\ M_6 \Rightarrow M_7, M_9 & M_7 \Rightarrow M_6, M_9 \\ M_8 \Rightarrow \phi & M_9 \Rightarrow M_6, M_7 \end{array}$$

Analisando o grafo da figura 3 vemos então que temos que aplicar um algoritmo de coloração. Por questão se simplicidade vamos simular uma coloração gulosa simples. Ele funciona da seguinte forma: - Inicia em um V0; $V_0 \leftarrow Cor1$ - Analise dos vizinhos de V0 - Para cada vértice vizinho tentar atribuir a menor cor possível sendo que nenhum dos vizinhos do vizinho tenha mesma cor. Se todos os vizinhos do vizinho já usam um cor, é necessário criar uma nova cor.

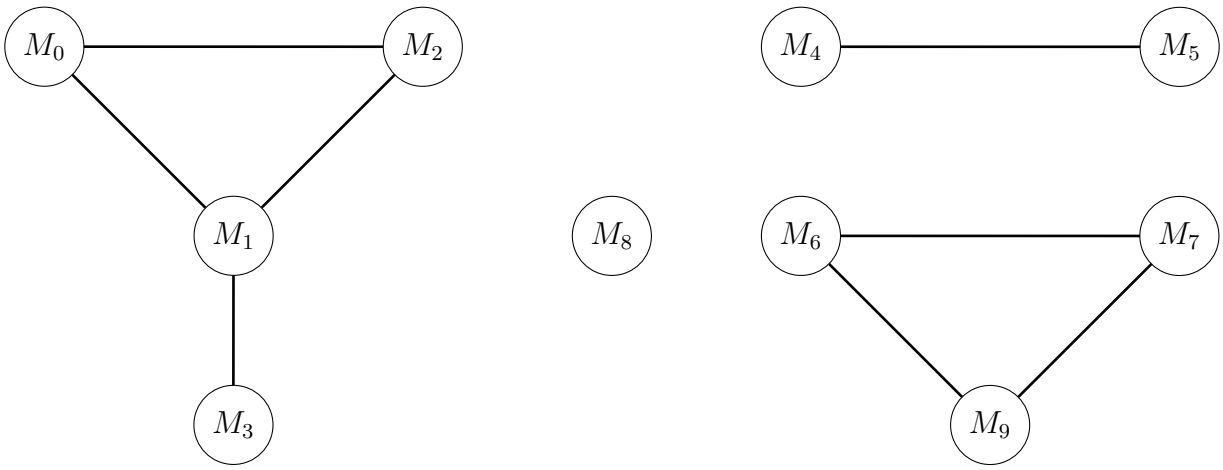


Figura 2: Conjunto de maquinas representados por vértices onde cada maquina que possui um mesmo intervalo de tempo com a outra é ligada por um par de aresta

Desta forma, usando esse algoritmo e iniciando em $M = 0$ temos que para um conjunto de cores $C = [1.Branco, 2.Roxo, 3.Azul]$ iniciando apenas com uma cor branca. Logo a solução do nosso grafo fica da seguinte forma.

$$\begin{array}{r|l} CorM_0 \Rightarrow 1 & CorM_1 - CriarCor \Rightarrow 0 \\ CorM_2 - CriaCor \Rightarrow 2 & CorM_3 \Rightarrow 1 \\ CorM_4 \Rightarrow 1 & CorM_5 \Rightarrow 2 \\ CorM_6 \Rightarrow 1 & CorM_7 \Rightarrow 2 \\ CorM_8 \Rightarrow 1 & CorM_9 \Rightarrow 3 \end{array}$$

Na figura vemos o resultado da coloração.

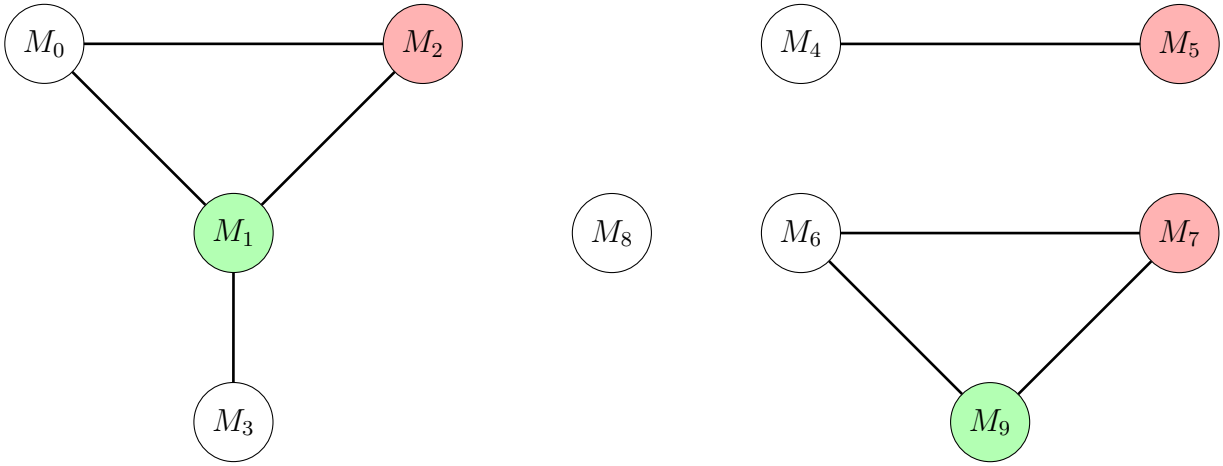


Figura 3: Resultado da coloração usando um algoritmo guloso simples, no exercício 5

6 Prove que toda floresta tem no máximo um matching perfeito.

Sabemos que uma floresta é dividida em arvores que são componentes conexas sem ciclos. Analisando cada arvore individualmente provaremos então que existe apenas uma possibilidade de permutação de matching perfeito. Sabemos que para um matching ser perfeito ele deve representar todos os vértices existentes na arvore e o numero de vértices deve ser par (caso haja quantidade impar de vértices não conseguimos dividir ele em pares pois não possuímos ciclos e para existir uma permutação deste matching perfeito, iriamos sempre precisar de no minimo uma aresta a mais na nossa arvore). Considerando que para cada arvore encontraremos no máximo um matching perfeito, pois não há como rearranjar os pares de vértices de maneira que representaríamos todos(falta arestas), então se expandirmos para o grafo como um todo

temos que só existirá matching perfeito se existir uma possibilidade de matching perfeito para cada arvore dentro da floresta e esse matching perfeito é a união de todos os outros matchings referente a cada arvore da floresta.

7 O grafo completo bipartido $K_{m,n}$ é hamiltoniano? Explique.

Um grafo G é hamiltoniano se tem um ciclo hamiltoniano. Um ciclo hamiltoniano em um grafo G é um caminho(ou circuito) que contém todos vértice em G.

Um grafo bipartido $K_{m,n}$ não é necessariamente um grafo hamiltoniano. No caso em que o grafo bipartido tem $m \neq n$, cada conjunto só pode ter ligação com o outro. Então, não podemos retornar ao vértice inicial sem repetir um dos vértices do meio do caminho. Mostramos isto com um contra-exemplo.

Na figura 1 temos um grafo bipartido K com $n = 2$ e $m = 3$. Começamos nosso caminho a partir do conjunto $C1$ pelo caminho (A, C, B, E) ou (A, C, B, D) e vemos que não tem como ir ao vértice que sobra em $C2$, isso porque não há mais vértices em $C1$ que já não estejam no caminho.

Se começamos em $C2$ temos um caminho (C, A, D, B, E) que é um caminho hamiltoniano, porém, não temos como fechar o ciclo pois o vértice que sobra pertence ao mesmo conjunto do ultimo vértice do caminho, e não podemos ligar eles com uma aresta, pois não teríamos mais um grafo bipartido.

Portanto, concluímos que um grafo bipartido $K_{m,n}$ completo não é necessariamente hamiltoniano.

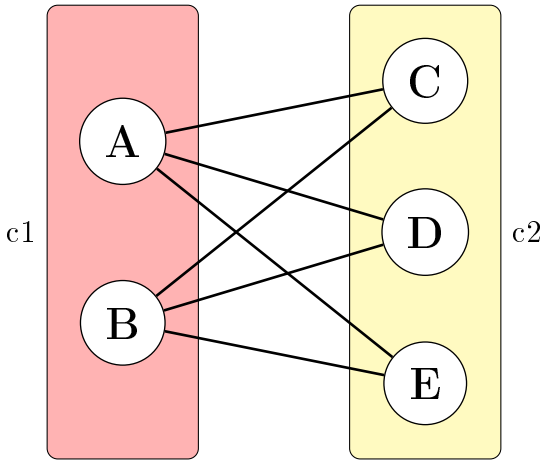


Figura 4: Grafo bipartido $K_{m,n}$ completo utilizado como contra exemplo na questão 7

8 Desenhe um grafo que seja de Euler, mas não hamiltoniano. Explique.

Na questão 1 definimos um grafo hamiltoniano. Um grafo de Euler é definido por um grafo que contém uma tour de Euler. Um tour de Euler é um tour no grafo G que contém cada aresta de G exatamente uma vez. Vemos um exemplo na figura 2.

Definimos o grafo como de Euler pois podemos realizar o tour $(A, C), (C, B), (B, C), (C, A)$.

Esse tour é um tour euleriano e portanto o grafo é euleriano. O Grafo não é Hamiltoniano pois não há maneira alguma que eu retorne para o vértice inicial sem antes repetir algum vértice no tour.

Notamos que essa característica é mantida para qualquer ponto de inicio. Se eu começo em B, resta ir apenas para C, e de C eu tenho que ir para A, caso contrario vou entrar em um laço e indo para A eu tenho que voltar para meu ponto inicial, porém só tem como fazer isso passando por C. Começando em C eu tenho que escolher ir para A ou B, porém, escolhendo um desses, para incluir o último vértice no caminho, eu tenho que passar por C, pois A não é ligada diretamente a B e vice versa.

Assim, concluímos que o grafo é de Euler e não hamiltoniano.

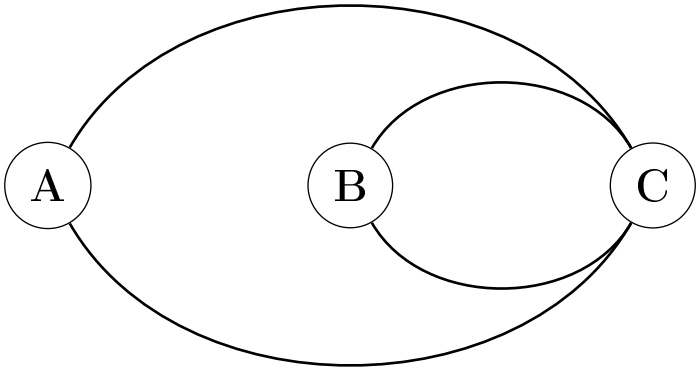


Figura 5: Grafo referente a questão 8

9 Prove ou mostre um contra-exemplo: Todo grafo com número de vértices $n \geq 3$ e grau mínimo $\delta(G) \geq 2$ possui um ciclo hamiltoniano.

Um ciclo hamiltoniano é em G é um ciclo que contém todo vértice em G. Aqui vamos mostrar um contra-exemplo que mostra que nem todo grafo com número de vértices $n \geq 3$ e grau mínimo $\delta(G) \geq 2$ possui um ciclo hamiltoniano.

Na figura 3 vemos um grafo com $N = 5$, portanto $N \geq 3$, e com $\delta(G) \geq 2$.

Esse grafo não possui nenhum ciclo hamiltoniano.

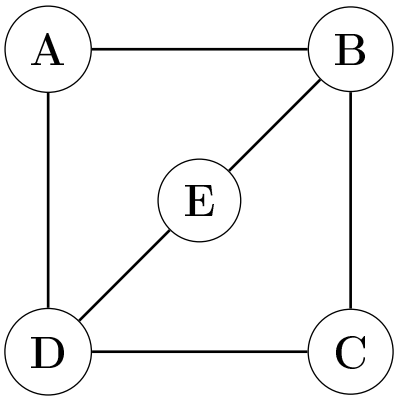


Figura 6: Grafo referente a questão 9

10 Prove que um grafo conexo G é um grafo euleriano se e somente se seu conjunto de arestas puder ser decomposto em ciclos.

Começamos provando a ida da proposta. Sabemos que o grafo é conexo e um grafo de Euler, portanto, todo grau de vértice do grafo é par. O grafo pode possuir apenas um circuito que seria o caso mais básico, se não, sobrariam mais arestas para serem percorridas. Como o grafo é conexo então existe um caminho entre qual par de vértices. Logo, existe um caminho entre algum vértice do circuito até uma aresta que não foi inclusa no circuito C.

Imagine que esse caminho seja formado pela aresta (j, k) onde j pertence a c e k pertence a g. Se isso ocorre deve-se percorrer o grafo oa partir de g visitando as novas arestas sem acessar nenhuma aresta em C. Esse novo circuito C_0 pode ser unido a C formando um único circuito. Se ainda existir arestas, repete-se o processo.

Portanto o circuito de Euler $E = c_1, c_2, \ldots, c_n$ onde a intercessão das aresta do circuito é nula.

Para provar que a reciproca vale supor um grafo G que pode ser decomposto em circuitos, isto é, G é uma união de circuitos disjuntos. Como todo vértice de um circuito possui grau par, o grau de cada vértice de G é necessariamente par, então G é um grafo de Euler.

11 Mostre que o ciclo *ebacde* é uma solução para o problema do caixeiro-viajante, para o grafo mostrado a seguir.

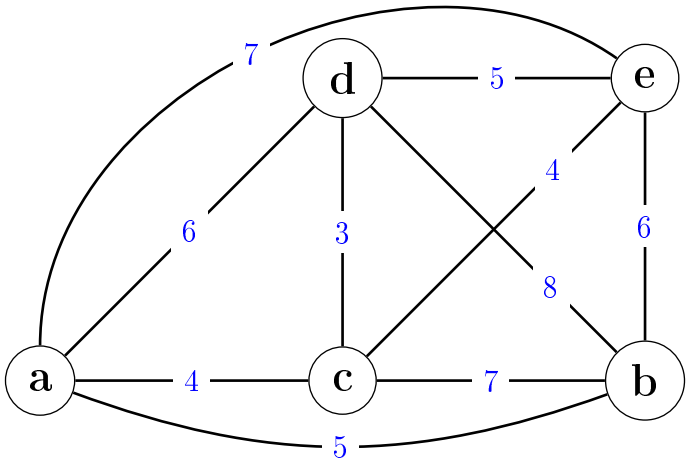


Figura 7: Grafo referente a questão 11

Para ser solução do problema de cacheiro viajante temos que o grafo tem que ser hamiltoniano e o ciclo hamiltoniano deve ter peso mínimo. Vemos que o caminho é hamiltoniano pois ele percorre todos os vértices no circuito (E, B, A, C, D, E). Vemos que esse circuito não repete nenhum vértice intermediário, apenas o primeiro e ultimo vértice, para fechar o circuito. Pode se conferir que o peso do caminho fechado é minimo por inspeção, já que analisar todas possibilidades geraria um problema $O(n!)$ para um grafo completo, ou seja, cerca de 120 operações para $n = 5$.

12 Resolva o problema do caixeiro-viajante para o grafo a seguir:

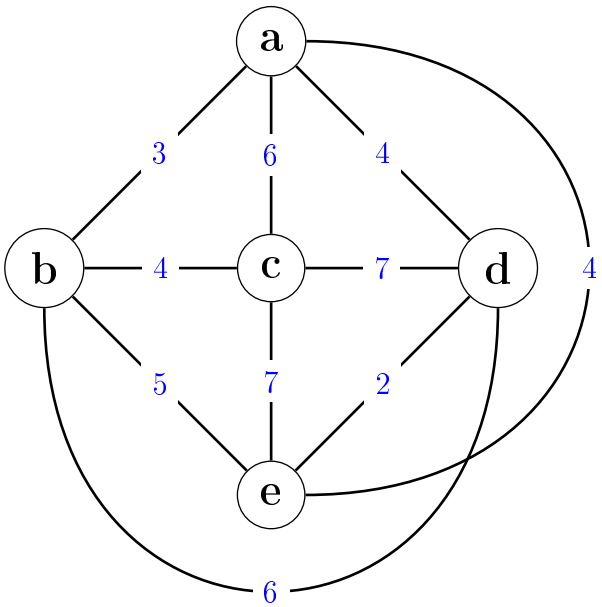


Figura 8: Grafo referente a questão 12

A solução é dada pelo caminho fechado (B, A, D, E, C, B) onde encontramos este resultado por inspeção. Neste caso todos os vértices são contidos no caminho e o único vértice repetido é o inicial/final logo

temos um grafo e caminho hamiltoniano, então esse caminho fechado sendo o de custo minimo no grafo, logo concluímos temos uma solução para o problema do caixeiro viajante com custo 20.

13 Resolva o problema do carteiro chinês para o grafo a seguir:

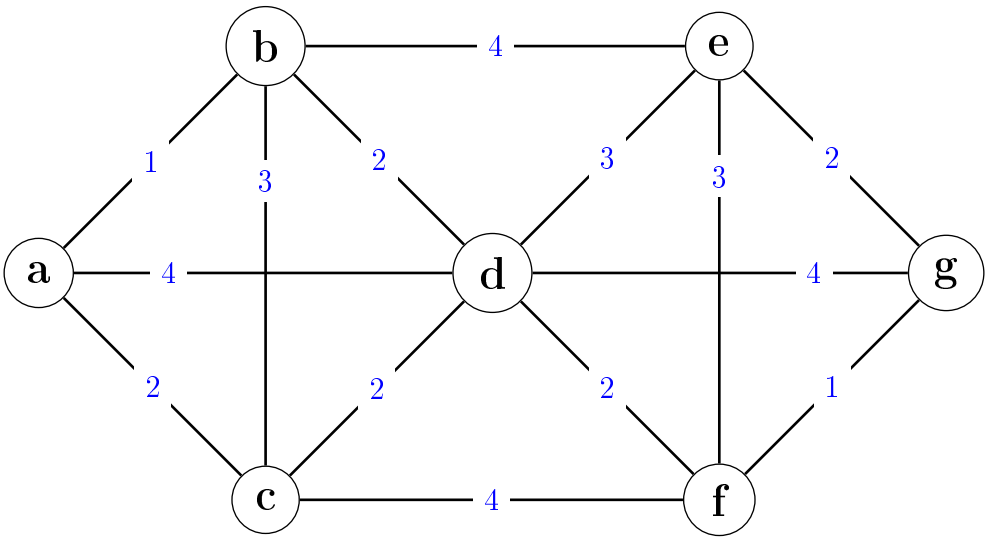


Figura 9: Grafo referente a questão 13. Na primeira figura a esquerda vemos o grafo após adicionar as arestas para tornar todo os graus pares. Na segunda figura vemos o caminho fechado que é a solução do problema.

Para resolver o problema precisamos encontrar o grafo euleriano com menor custo. Vemos que o grafo dado não é euleriano pois existem dois vértices com grau 3, e sabemos que para um grafo ser euleriano, todos os vértices devem ter grau par. Para resolver esse problema vamos usar as técnicas de duplicar arestas de maneira que o grafo obtido tenha o menor peso. Precisamos adicionar o menor numero de arestas possível e com o menor peso.

Analisando o grafo vemos que ao adicionar mais uma aresta (a, b) com peso 1,(b, d) com peso 2, (d, f) com peso e por ultimo outra (f, g) com peso 1, teremos um grafo euleriano, pois todos vértices terão grau par, e o custo será o menor, já que serão adicionados apenas duas arestas com o peso somado igual a 6.

A solução do problema é dado pelo conjunto de arestas $E = E_1, E_2, \ldots, E_{17}, E_{18}$ com inicio e fim em A onde essas arestas são dadas na figura 4. Notamos que as arestas foram direcionadas apenas para facilitar a leitura da figura. A resposta final possui um custo 43 onde 37 é a soma do peso de todas as arestas do grafo original e 6 é o peso das arestas que duplicamos, somando os dois, obtemos 43.

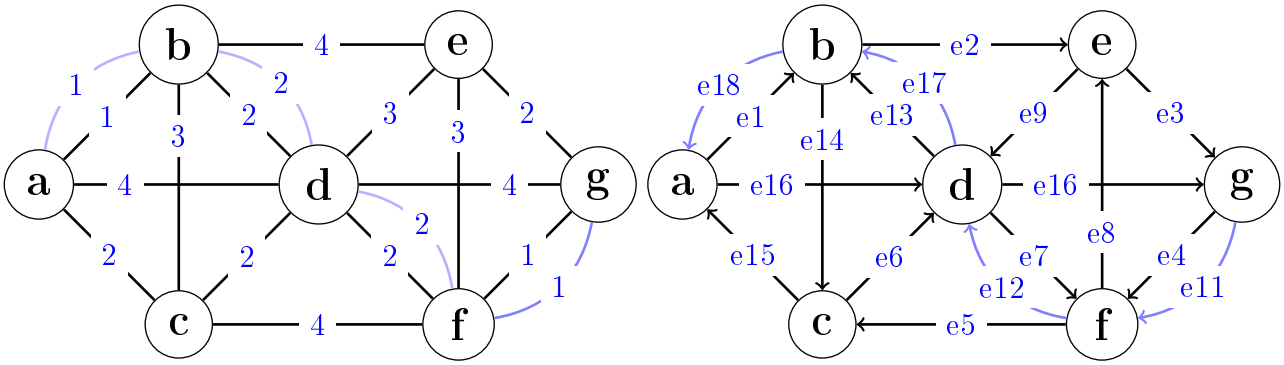


Figura 10: Grafo referente a questão 13. Na primeira figura a esquerda vemos o grafo após adicionar as arestas para tornar todo os graus pares. Na segunda figura vemos o caminho fechado que é a solução do problema.

14 Mostre que, se e for qualquer aresta de K_5, então $K_5 - e$ é planar.

Para realizar essa prova devemos mostrar o porquê que os K_5 não é planar.

Sabemos que o K_4 é planar e usamos isto para a demonstração. Se temos 4 vértices conectados entre si, devemos então inserir mais um vértice ligado aos outros existentes. Sabemos que o vértice v_5 a ser inserido deve estar entre uma das 4 regiões de K_4, em int($c1$), int($c2$), int($c3$), onde $c1$ é o ciclo (V_1, V_2, V_4, V_1), $c2$ é o ciclo (V_2, V_3, V_4, V_2) e $c3$ é o ciclo (V_1, V_3, V_4, V_1).

Como nenhum dos ciclos inclui todos os vértices, ao inserir v_5 em algum deles, sempre existirá uma aresta fechando a curva de Jordan, fazendo assim que não possa existir a ligação sem o cruzamento de arestas, então se removermos uma das arestas de algum dos ciclos, não existiria a curva de Jordan, e concluímos que o grafo $K_5 - e$ seria planar.

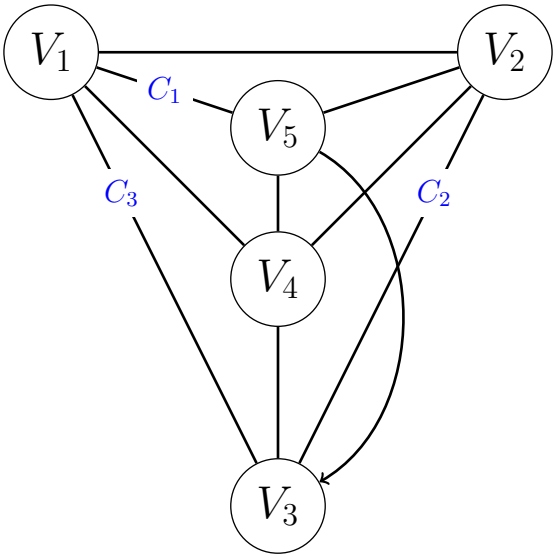


Figura 11: Grafo referente a questão 14. Um exemplo para visualizar o que acontece quando inserimos V_5 dentro de uma face. Temos que alguma aresta de v_5 sempre vai ter que cortar um limite da região.

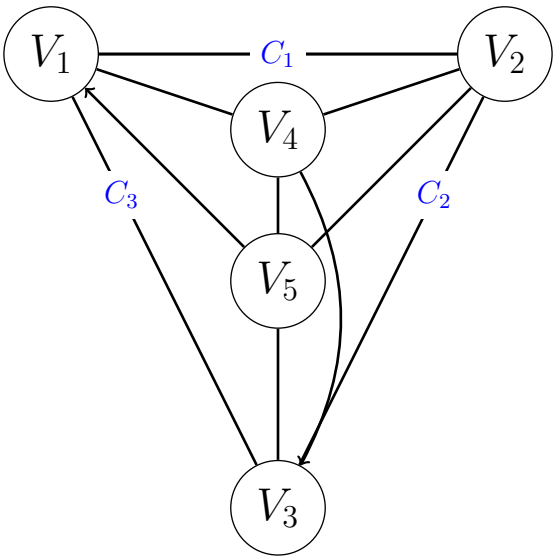


Figura 12: Grafo referente a questão 14. Um segundo exemplo para visualizar o que acontece quando inserimos v_5 dentro de uma face. É fácil observar que v_5 sempre corta uma única aresta, então removendo a mesma, temos que o grafo é planar. Vale ressaltar que há de uma aresta que pode ser removida para tornar o grafo planar e a figura é apenas para ilustrar que sempre há um corte por alguma aresta.

15 Considere G um grafo plano 4-regular com 10 faces. Determine quantos vértices G tem e desenhe G.

Sabemos que por ser um grafo regular a soma dos graus de cada vértice é o dobro do total de arestas. Portante temos que $4v = 2e$ e temos que

$$\begin{aligned} \Rightarrow f &= e - v + 2 \\ \Rightarrow 10 &= 2v - v + 2 \\ \Rightarrow v &= 10 - 2 \\ \Rightarrow v &= 8 \end{aligned}$$

Vemos um exemplo do desenho de G figura.

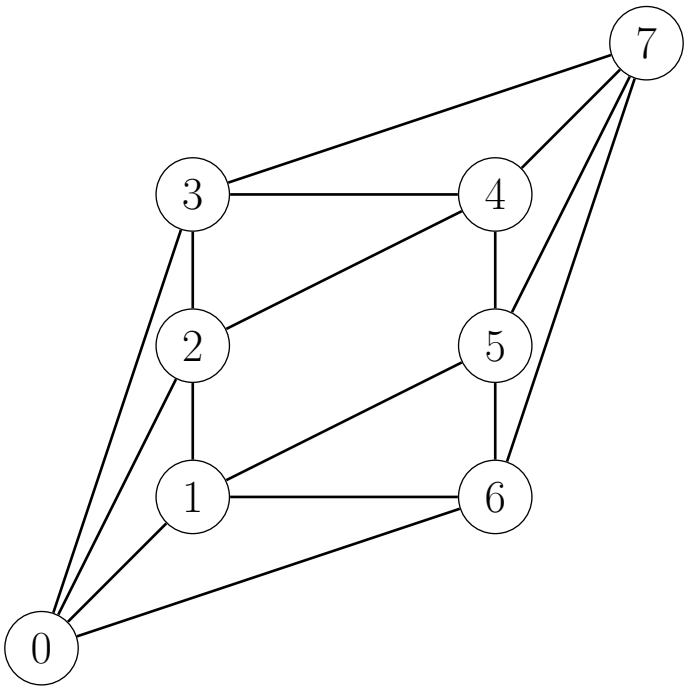
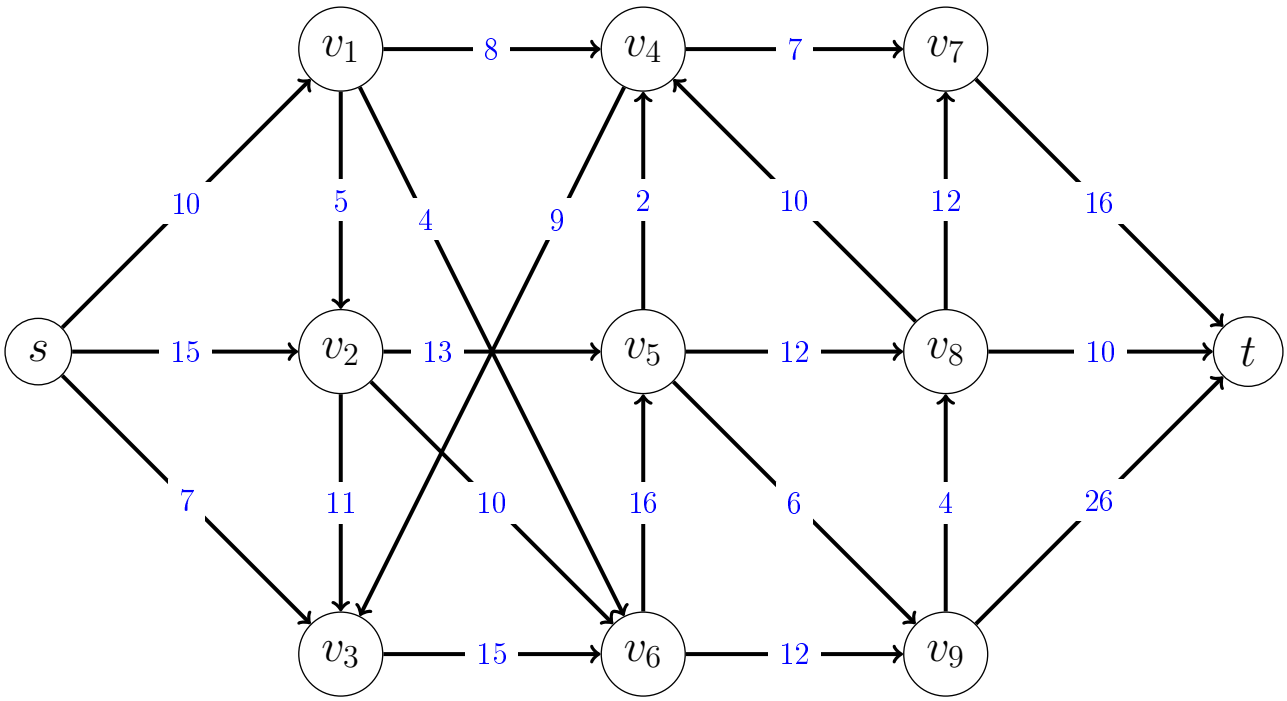


Figura 13: Grafo que representa uma representação de G do resultado da questão 15.

16 Aplique o algoritmo de Ford-Fulkerson no grafo abaixo. Apresente o passo-a-passo e o resultado final. Para cada passo, apresente o grafo residual.

A estratégia geral para resolver essa solução enquanto existir caminho aumentante de fluxo com inicio em t e fim em s, onde o caminho aumentante foi escolhido de maneira aleatória, encontramos o bottleneck do caminho (aresta com menor possibilidade de fluxo) e então acrescentamento esse fluxo a todo as arestas do caminho desde que todos tinha capacidade. Atualizamos o grafo residual onde as arestas azuis demonstram a quantidade de fluxo remanescente que poderia ser passado por cada aresta e inserimos ou atualizamos a aresta vermelha (v, u) que representa a quantidade de fluxo que já passa de u para v. Fazemos isto até não existir mais caminhos aumentantes. Realizando esse processo e analisando a última figura temos que o fluxo máximo é 32.



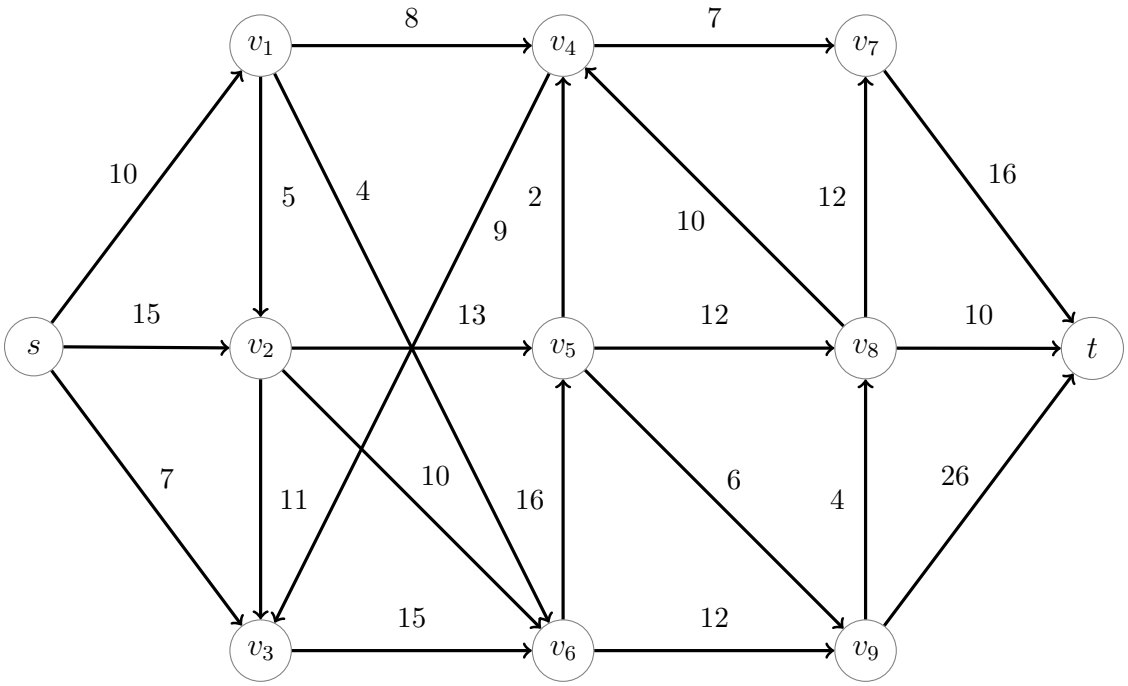


Figura 14: Grafo encontrado na iteração 0 da questão 16. Notamos que este passo consiste em apenas iniciar todos os valores do grafo de fluxo como 0 e fazer uma cópia do grafo original para o grafo residual.

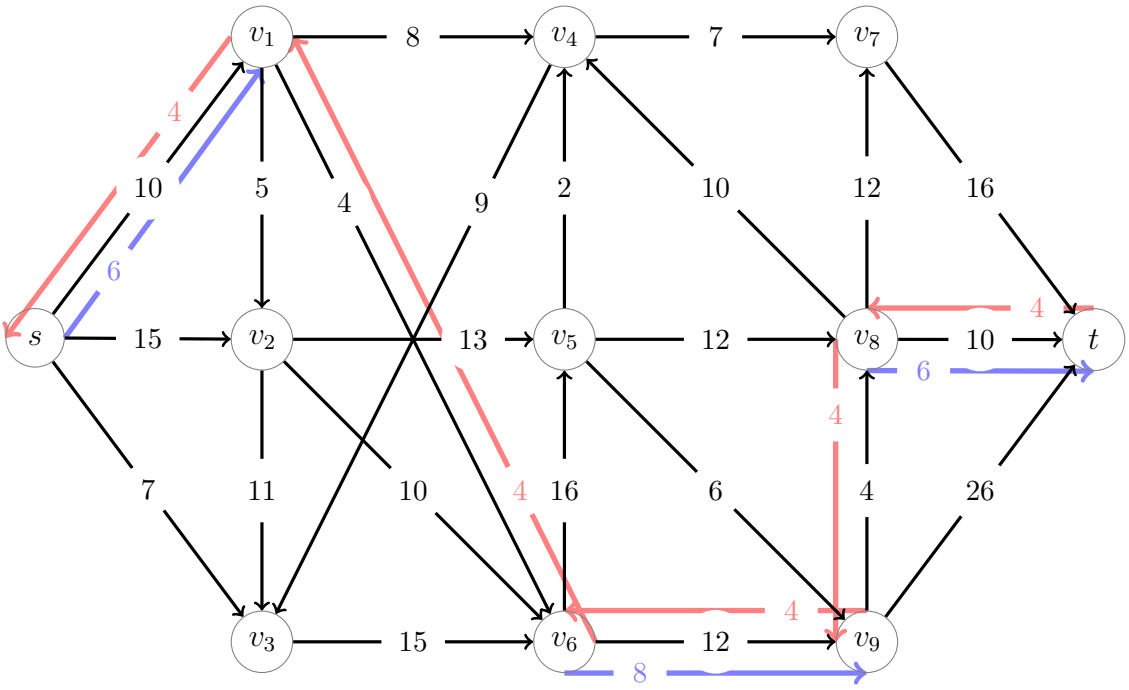


Figura 15: Dois grafos encontrados na iteração 1 da questão 16. Escolhemos de forma aleatória o caminho $(t, v_1, v_6, v_9, v_8, s)$ que possui um valor de bottleneck 4 e então atualizamos os dois grafos.

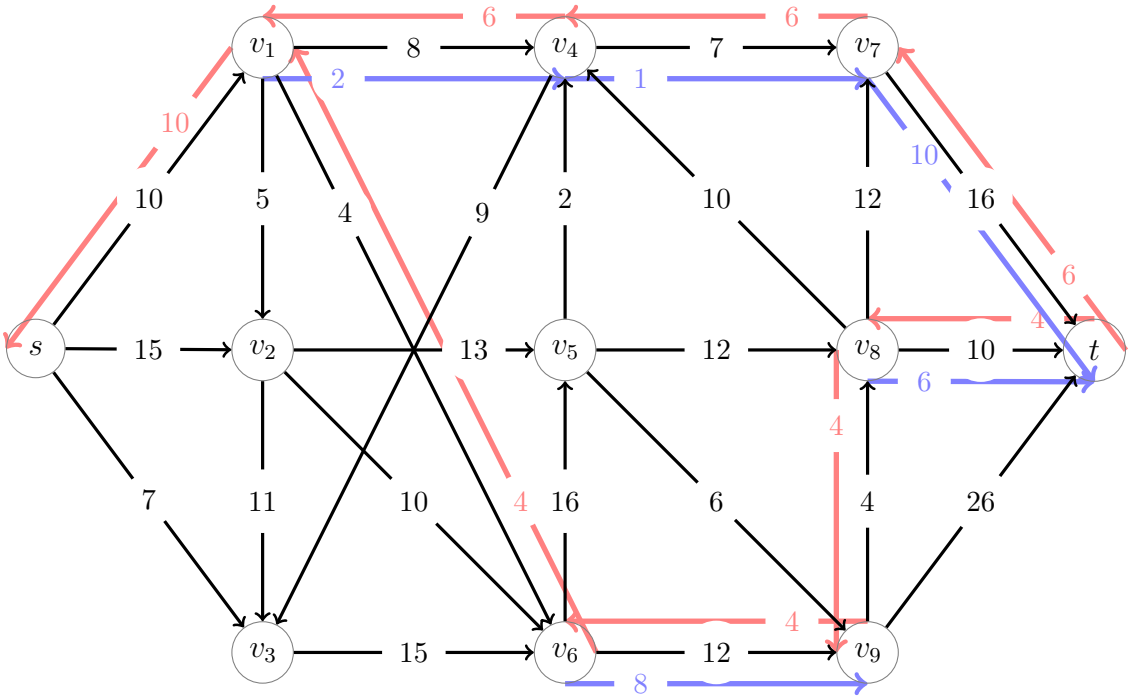


Figura 16: Grafo encontrado na iteração 2 da questão 16. Nesta iteração escolhemos o caminho aumentante (s, v_1, v_4, v_7, t) que possui um valor de bottleneck 6 e atualizamos os grafos. Notamos que a aresta que liga v 1 a fonte já possui fluxo máximo então não podemos mais incluir ela nos caminhos aumentantes.

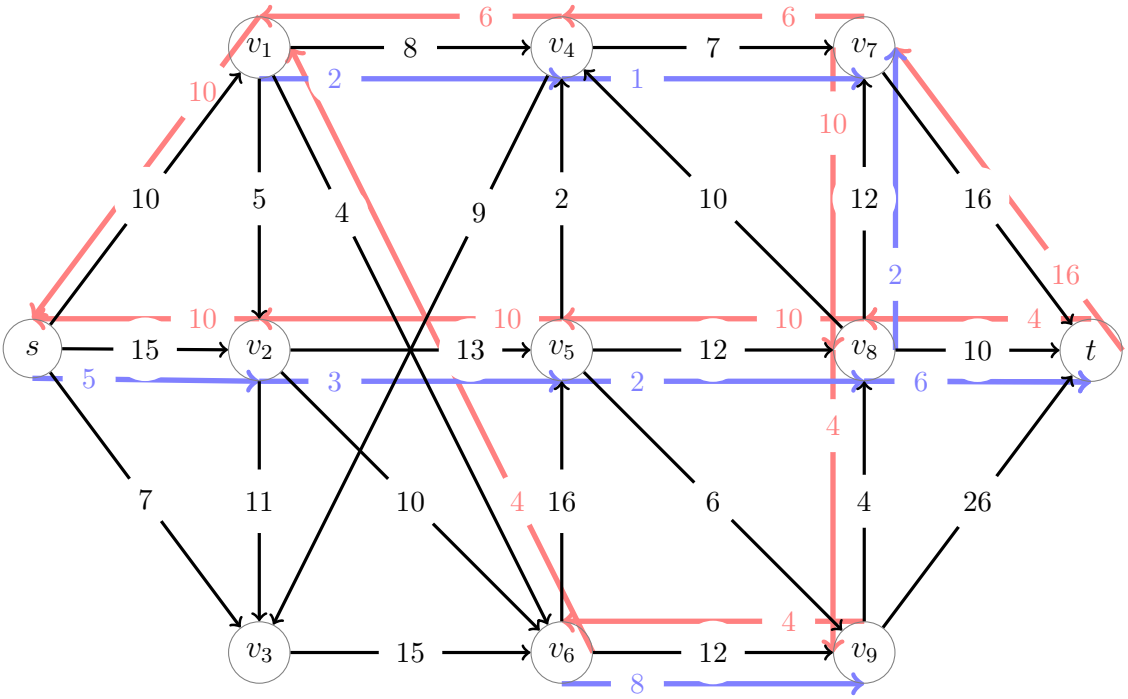


Figura 17: Grafo encontrado na iteração 3 da questão 16. Aqui escolhemos o caminho aumentante $(s, v_2, v_5, v_8, v_7, t)$ que possui um valor de bottleneck 10. Notamos que fazendo isto não temos que levar mais enconta a aresta que liga v_7 ao sumidouro, pois o fluxo já é máximo.

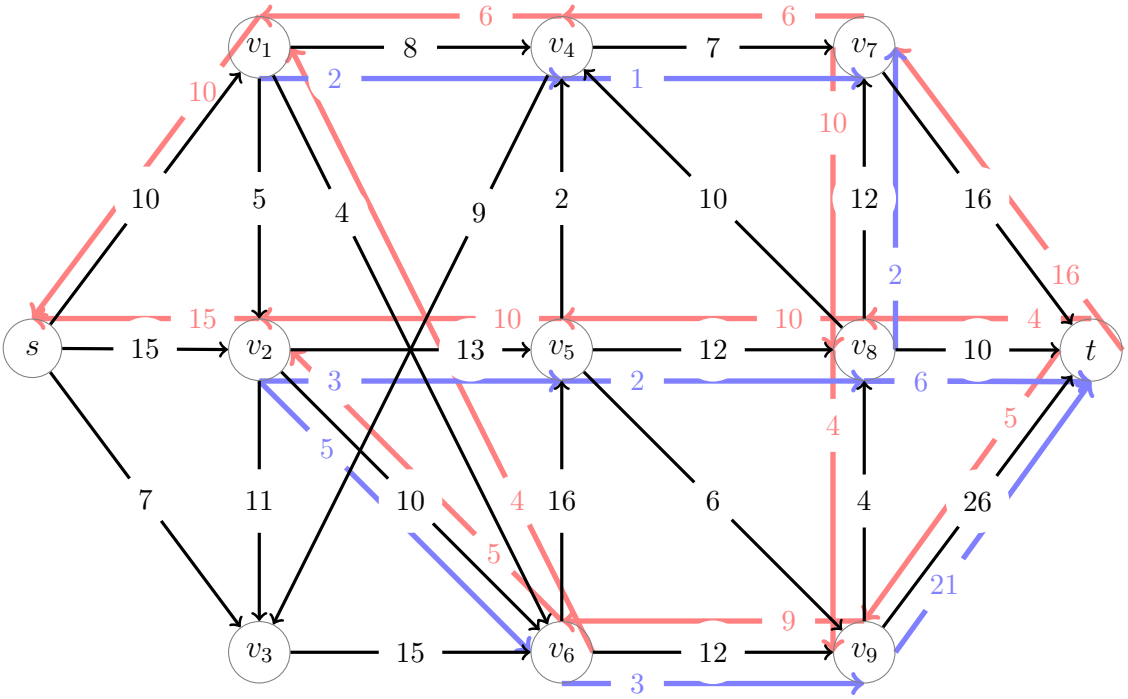


Figura 18: Grafo encontrado na iteração 4 da questão 16. Utilizamos o caminho aumentante (t, v_2, v_6, v_9, s) com um valor de bottleneck 5 visando ter fluxo máximo pela aresta que liga a fonte ao vértice v_2. Após este passo temos apenas uma aresta por onde a fonte pode fornecer fluxo.

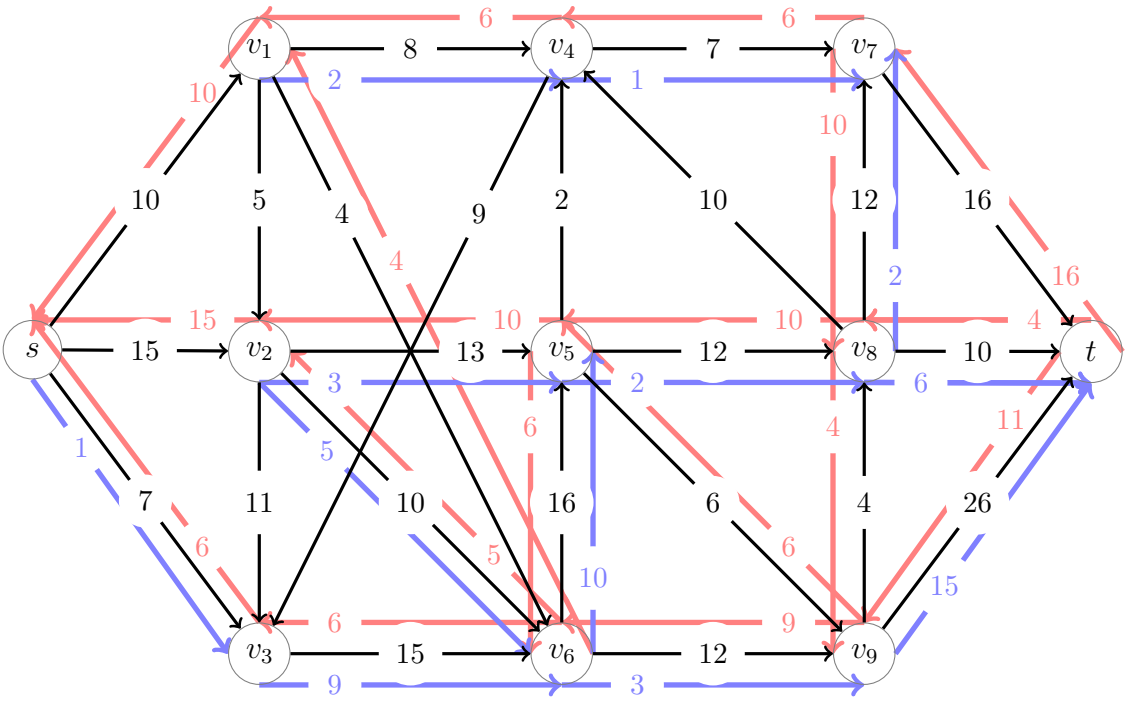


Figura 19: Grafo encontrado na iteração 5 da questão 16. Nesta iteração escolhemos o caminho $(s, v_3, v_6, v_5, v_9, t)$ com um valor de bottleneck 6. É fácil perceber que este vai ser o penúltimo passo do nosso algoritmo pois tentamos escolher um caminho aumentante com o maior valor de fluxo possível e há apenas uma unidade de fluxo restante para passar por v_3.

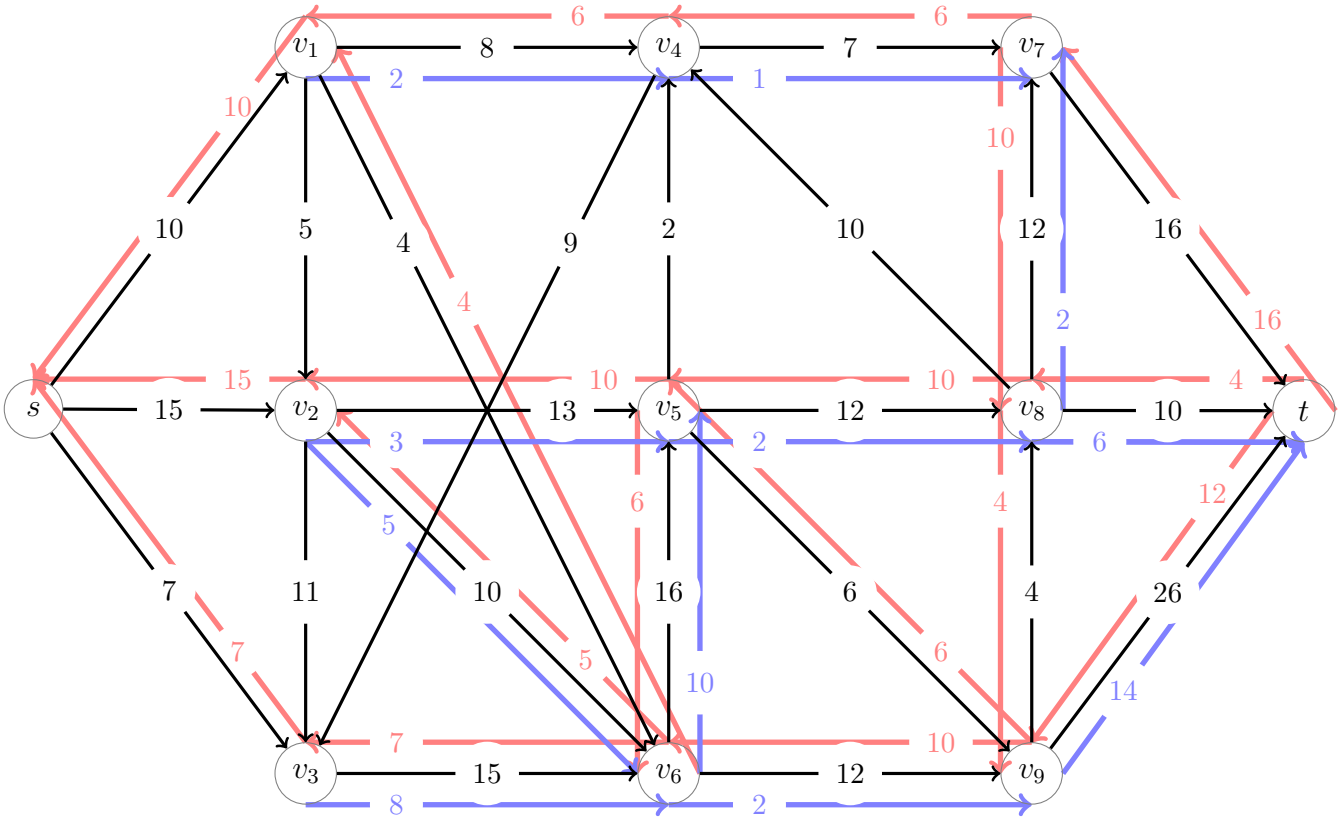


Figura 20: Grafo encontrado na iteração 6 da questão 16. Aqui encontramos a ultima iteração do algoritmo que vamos realmente mudar a estrutura do grafo. Escolhemos o caminho (s, v_3, v_6, v_9, t) usando um bottleneck de 1. Atualizamos os grafos o nosso programa tentar executar uma ultima iteração. Encontra se que não há mais caminho aumentante (não é possível sair de t e chegar em s no grafo residual), portanto o nosso algoritmo finaliza sua execução e nosso fluxo máximo é a soma dos pesos das arestas que ligam a s no nosso grafo de fluxo que é 32.