

Capítulo 4

Grafos e Algoritmos via Indução

Este capítulo se concentra nos algoritmos fundamentais sobre grafos, uma estrutura discreta de importância decisiva na representação e tratamento de inúmeros problemas relacionados a áreas como topologia, lógica e teoria dos números, entre outras.

4.1 Definições Básicas

Definição Um grafo G consiste de um conjunto suporte $V = \{1, 2, \dots, n\}$ de vértices (ou nós) e um conjunto de pares de elementos de V , $E = \{(i_1, j_1), (i_2, j_2), \dots, (i_m, j_m)\}$. Estes pares são chamados de arestas ou arcos e representam uma ligação entre os vértices i_k e j_k para $k = 1, 2, \dots, m$. Assim, V e E determinam o grafo G , denotado por $G = (V, E)$.

Definição Um grafo $G = (V, E)$ é dito orientado se em E os elementos (v, w) e (w, v) são considerados diferentes. O primeiro representa uma ligação de v para w e o segundo de w para v . Em grafos orientados, os elementos de E são mais frequentemente chamados de arcos.

Definição Um grafo $G = (V, E)$ é dito não-orientado se em E os elementos (v, w) e (w, v) são considerados iguais (em E aparece apenas um deles). Nesse caso, ambos representam uma ligação entre v e w . Em grafos não-orientados, normalmente é utilizado o termo aresta para denotar cada um dos elementos de E .

Todo grafo não-orientado $G = (V, E)$ pode ser representado por um grafo orientado $G = (V, E')$, sendo E' tal que $E' = \{(u, w), (w, u) \mid (u, w) \in E\}$. Cada aresta de um grafo não-orientado equivale a dois arcos orientados, cada um deles ligando o par de vértices em um sentido.

Definição Um caminho P de s a t em um grafo $G = (V, E)$ é uma sequência de vértices $s, i_1, i_2, \dots, i_{k-1}, t$ tal que $(s, i_1) \in E$, $(i_q, i_{q+1}) \in E$, $q = 1, 2, \dots, k-2$, e $(i_{k-1}, t) \in E$. Este caminho P possui k arestas ou arcos.

Definição Um grafo $G = (V, E)$ é dito conexo se e somente se para todo par de vértices $s, t \in V$ existe caminho de s para t no grafo não-orientado G' correspondente a G .

4.2 Caminhos Eulerianos

Definição Um caminho euleriano em um grafo $G = (V, E)$ é um caminho fechado em que cada uma das arestas de G ocorre exatamente uma vez.

Teorema Um grafo conexo não-orientado conterá um caminho euleriano se e somente se todos os seus vértices tiverem grau par.

Prova Observe que esse é um teorema do tipo “se e somente se”. Assim sendo, devem ser provadas, na verdade, duas proposições.

Lema (Necessidade) Um grafo conexo não-orientado conterá um caminho euleriano somente se todos os seus vértices tiverem grau par.

Prova Deve-se provar que é *necessário* que todos os vértices de um grafo que contém um ciclo euleriano tenham grau par. A prova é por contradição. Suponha que exista um grafo $G = (V, E)$ que contém um ciclo euleriano e pelo menos um vértice $v \in V$ de grau ímpar. Como zero é par, há pelo menos uma aresta incidente nesse vértice, o que significa que o caminho euleriano obrigatoriamente passa por ele. Como o caminho euleriano é fechado, é possível percorrer todas as arestas do grafo a partir de v e retornar a esse vértice sem passar por nenhuma aresta mais de uma vez. Para evitar repetição, suponha que, à medida que vão sendo visitadas, as arestas sejam marcadas. Como o caminho parte de v , alguma das arestas será marcada na saída. Como o caminho é fechado, obrigatoriamente deve haver uma outra aresta incidente em v pela qual se retornará a esse vértice (essa aresta também será marcada nesse ponto). Assim, sempre que estivermos em v , o número de arestas incidentes não marcadas será ímpar; a cada visita, esse número é reduzido em exatamente duas arestas. Como o grau de v é finito, em algum momento estaríamos em v e haveria apenas uma aresta não marcada. Se ela fosse utilizada, não haveria por onde voltar a v para que o caminho fosse fechado, o que é uma contradição. Portanto, todos os vértices do grafo devem ter grau par. \square

Lema (Suficiência) Um grafo conexo não-orientado conterá um caminho euleriano se todos os seus vértices tiverem grau par.

Prova Um grafo conexo G em que todos os vértices têm grau par obrigatoriamente contém um ciclo (se não contivesse, haveria pelo menos um nó de grau um). A prova será feita justamente por indução forte em c , o número de ciclos (c inclui todos os ciclos, disjuntos ou não).

O caso base é $c = 1$. Se um grafo conexo em que todos os vértices têm grau par contiver um único ciclo C , todas as suas arestas estarão nele. Se isso não fosse verdade, a retirada desse ciclo único provocaria a formação de pelo menos uma árvore, que tem no mínimo um vértice de grau 1. Como a retirada de um ciclo não altera a paridade dos nós do grafo (diminui em exatamente duas unidades — um número par — o grau de cada vértice envolvido), isso significaria que o grafo original teria pelo menos um nó de grau ímpar, o que é uma contradição. Portanto, C contém todas as arestas do grafo e forma um circuito euleriano.

Como hipótese de indução, suponha que seja possível encontrar um caminho euleriano em um grafo conexo com $c < k$ ($k > 1$) ciclos. Seja $G = (V, E)$ um grafo conexo com exatamente k ciclos. Retire desse grafo C , um ciclo qualquer (existe pelo menos um), formando o grafo G' . Devem ser removidos tanto as arestas de C quanto os vértices que tiverem grau 2 (vértices de grau maior serão preservados). Todos os vértices de G' terão grau par, já que sua paridade não é alterada pela remoção de um ciclo. É possível que G' tenha uma ou mais componentes conexas, todas elas com pelo menos um par de arestas e menos de k ciclos. Seja m o número de componentes conexas formadas. Pela hipótese de indução, cada uma dessas componentes possui um caminho euleriano. Sejam eles C_1, C_2, \dots, C_m . Para construir um caminho euleriano em G , basta unir C aos caminhos eulerianos de C_1, C_2, \dots, C_m . Isso pode ser feito de forma relativamente simples. Percorre-se C a partir de um vértice qualquer e seguindo um sentido arbitrariamente escolhido. A cada vértice v visitado, verifica-se se ele também faz parte de algum outro caminho euleriano ainda não percorrido. Digamos que ele faça parte de C_i . Percorre-se todo o caminho C_i e retorna-se a v (isso é sempre possível, já que o caminho é fechado). Nesse momento, C_i deve ser marcado como visitado. A operação se repete até que todos os caminhos ainda não visitados que passam por v sejam percorridos. Ao final desse processo, avança-se para o vértice seguinte de C . Quando todos os vértices de C tiverem sido percorridos, todas as arestas de G terão sido visitadas uma única vez, o que caracteriza o caminho euleriano. \square

Devidamente provados, os dois lemas são suficientes para garantir que o teorema está correto. \square

4.3 Busca em Grafos

4.3.1 Busca em Largura

Definição *Seja $G = (V, E)$ um grafo não-orientado e $v, w \in V$ dois de seus vértices. Diz-se que w pertence à k -ésima vizinhança de v se o caminho com número mínimo de arestas entre v e w contiver exatamente k arestas.*

Observe que a definição de vizinhança leva em conta o *menor* caminho entre v e w . Se houver mais de um caminho entre os dois vértices (como é comum que ocorra), o que determinará em qual vizinhança w estará é aquele como o menor número de arestas. Observe ainda que, se o grafo possuir $|V|$ vértices, haverá no máximo $|V| - 1$ vizinhanças (por quê?).

Teorema *Dado um grafo conexo não-orientado $G = (V, E)$ e um vértice $s \in V$, é possível visitar todos os vértices até a n -ésima vizinhança de s , sendo $n = 1, \dots, |V| - 1$.*

Prova Por indução em n . O case base é $n = 1$. A primeira vizinhança de um nó é composta por todos os seus vizinhos. Basta, portanto, visitá-los. Suponha, como hipótese de indução, que o teorema seja válido para $n = k$ ($k \geq 1$), ou seja, que seja possível visitar todos os nós até a k -ésima vizinhança de s . Deseja-se provar que o teorema é válido para $n = k + 1$, ou seja, que é possível visitar todos os nós cuja distância mínima a s (em número de arestas) é menor ou igual a $k + 1$. Seja w um vértice da $k + 1$ -ésima vizinhança de v . Por definição, existe um caminho P_w com exatamente $k + 1$ arestas entre w e s . Partindo de w , seja w' o segundo vértice desse

caminho (o primeiro é o próprio w). O caminho mínimo de w' a s tem exatamente k arestas, todas as do caminho P_w exceto (w, w') , o que significa que w' pertence à k -ésima vizinhança de v . Assim, qualquer vértice da $k + 1$ -ésima vizinhança tem um vizinho da k -ésima vizinhança. Portanto, para percorrer todos os vértices da $k + 1$ -ésima vizinhança, basta visitar os vértices até a k -ésima vizinhança (o que a hipótese de indução garante ser possível) e, em seguida, visitar todos os vértices *ainda não visitados* adjacentes aos da k -ésima vizinhança. Repare que apenas os vértices da $k + 1$ -ésima vizinhança serão visitados nesse momento: vértices mais próximos de v já terão sido visitados (de acordo com a hipótese de indução) e vértices de vizinhanças mais distantes não podem, por definição, ser vizinhos de vértices de k -ésima vizinhança. \square

A estratégia de busca em grafos baseada nesse teorema é denominada *busca em largura*, já que a visitação é feita em camadas sucessivas.

4.3.2 Busca em Profundidade

Teorema Dado um grafo conexo não-orientado $G = (V, E)$, sendo $|V| = n$, é possível percorrer todos os seus vértices.

Prova Por indução forte em n . O caso base é $n = 1$: se o grafo tiver um único vértice, basta marcá-lo como visitado para completar a busca. Suponha, como hipótese de indução, que seja possível visitar todos os vértices de um grafo com $n \leq k$, $k \geq 1$. Deseja-se provar que o mesmo é válido para $n = k + 1$. Seja v um vértice desse grafo. Visite-o. Sejam $\{v_1, v_2, \dots, v_m\}$ os vértices da vizinhança de v e $\{e_1, e_2, \dots, e_m\}$ as arestas que ligam v a cada um deles. Se essas arestas forem temporariamente removidas, o grafo original será decomposto em duas ou mais componentes: uma delas conterá o vértice v isoladamente e as demais, os vértices restantes. Nenhuma dessas componentes pode ter mais de k vértices. É possível, portanto, aplicar a hipótese de indução sobre cada uma delas. Inicialmente, percorre-se a componente que contém v_1 (o que a hipótese de indução garante ser possível). Em seguida, percorre-se a que contém v_2 , depois v_3 e assim sucessivamente, até v_m . Como dois ou mais desses vértices podem pertencer à mesma componente, deve-se testar se cada um deles já foi visitado antes de se aplicar a hipótese de indução. Para v_1 , a aplicação será sempre necessária; para os demais vizinhos, nem sempre. Independentemente do número de aplicações da hipótese de indução, garante-se que serão visitados todos os vértices que ainda não o tiverem sido. \square

A estratégia de busca baseada nessa é comumente denominada de *busca em profundidade*. Em lugar de se percorrerem inicialmente os vizinhos mais próximos da raiz (como se faz na busca em largura), percorrem-se todos os vértices alcançáveis a partir de um vizinho antes de se visitar o seguinte.

4.4 Caminhos Mínimos

4.4.1 Arestas com Custos Positivos

Teorema Seja $G = (V, E)$ um grafo não-orientado ponderado (com pesos positivos) e $s \in V$ um de seus vértices. É possível encontrar o p -ésimo vértice mais próximo de s , sendo $1 \leq p \leq n - 1$, bem como o valor de sua distância a s ($d(s)$).

Prova Por indução forte em p . O caso base é $p = 1$: é possível encontrar o vértice mais próximo de s . Seja $e_{min} = (s, s')$ a aresta incidente em s que tem peso mínimo. O vértice mais próximo de s é justamente $s_1 = s'$, a outra extremidade da aresta, e a distância $d(s_1)$ é dada pelo peso da aresta e_{min} . Nenhum outro vértice v do grafo pode estar mais próximo de s , pois qualquer caminho de s a v obrigatoriamente passa por uma das arestas incidentes em s . Como os custos são positivos, isso significa que o custo do caminho é pelo menos igual ao peso de e_{min} .

Como hipótese de indução, suponha que o teorema seja válido para $p = k - 1$, ou seja, que seja possível determinar os $k - 1$ vértices mais próximos de s , bem como as respectivas distâncias. Seja $S_{k-1} = \{s = s_0, s_1, s_2, \dots, s_n\}$ o conjunto dos vértices mais próximos e $d(s_0), d(s_1), \dots, d(s_n)$ as respectivas distâncias a s .

Considere o caso $n = k$. Seja s_k o k -ésimo vértice mais próximo de s (justamente o vértice que se deseja determinar). O caminho mais curto entre s e s_k (P_{s_k}) é formado apenas por um subconjunto dos $k - 1$ vértices mais próximos de s , além do próprio s e de s_k . (Se algum outro vértice u não pertencente a S_{k-1} fizesse parte desse caminho, ele estaria mais próximo de s que o próprio s_k , que, dessa forma, não poderia ser o k -ésimo vértice mais próximo.) Considerando o caminho P_{s_k} no sentido de s_k para s , seja s' seu segundo vértice (o vizinho de s_k). A distância de s_k a s será igual à distância de s a s' somada ao peso da aresta (s', s_k) .

Uma vez estabelecidas as propriedades de s_k , é possível determinar exatamente qual é esse vértice. Pela hipótese de indução, é possível determinar o conjunto S_{k-1} e as distâncias de cada um de seus elementos a s . Pelos motivos já expostos, apenas os vértices que são vizinhos de pelo menos um vértice de S_{k-1} são candidatos a ser s_k , o k -ésimo vértice mais distante de s . Seja $C_k = \{c_1, c_2, \dots, c_m\}$ ($m \geq 1$) a lista de candidatos. Para cada vértice $c_i \in C_k$ pode-se determinar sua distância a s passando apenas por vértices de S_{k-1} , denotada por $d_k(c_i)$ da seguinte forma: para cada vizinho s_j ($0 \leq j < k$) de c_i pertencente a S_{k-1} , determina-se a soma de $d(s_j)$ com o custo da aresta (s_j, c_i) . A distância de c_i a s será o mínimo entre os valores obtidos para todos os vizinhos testados. O vértice s_k , por sua vez, será o vértice $c_i \in C_k$ para o qual $d_k(c_i)$ tem valor mínimo. A distância $d(s_k)$ será igual a $d_k(c_i)$. \square

4.5 Árvore Geradora Mínima

4.5.1 Árvores e Florestas

Definição Uma árvore é um grafo acíclico, não-orientado e conexo.

Definição Uma floresta é um grafo acíclico e não-orientado (não necessariamente conexo).

Definição Dado um grafo $G = (V, E)$, diz-se que um grafo $G' = (V', E')$ é um subgrafo de G se e somente se $V' \subseteq V$ e $E' \subseteq E$.

Definição Seja $G = (V, E)$ um grafo não-orientado. Uma árvore geradora $A = (V', E')$ de G é um subgrafo acíclico e conexo que contém todos os vértices de G .

Teorema Qualquer árvore geradora de um grafo $G = (V, E)$, sendo $|V| = n$, tem exatamente $n - 1$ arestas.

Prova Por indução em n . O caso base, $n = 1$, é trivial. A árvore terá exatamente um nó e nenhuma aresta. Como hipótese de indução, suponha que o teorema seja válido para $n = k$, $k \geq 1$: uma árvore com k vértices terá exatamente $k - 1$ arestas. Deseja-se provar que o teorema é válido para $n = k + 1$, ou seja, que uma árvore geradora de um grafo com $k + 1$ nós terá exatamente k arestas. Seja A uma árvore geradora qualquer com $k + 1$ vértices e seja $|E_A|$ o número de arestas em A . Como $k \geq 1$, A deve possuir pelo menos uma aresta para que seja conexo (i.e., $|E_A| \geq 1$). Seja $e = (v, w)$ uma aresta qualquer de A . “Contraia” essa aresta, ou seja, faça com que suas extremidades, v e w , tornem-se um único nó. Como resultado, teremos uma nova árvore A' com k nós e $|E_A| - 1$ arestas. Pela hipótese de indução, qualquer árvore com k nós possui exatamente $k - 1$ arestas; isso significa que $|E_A| - 1 = k - 1$ e, portanto, $|E_A| = k$. \square

Definição *Seja $G = (V, E)$ um grafo não-orientado ponderado. Uma árvore geradora mínima $A = (V', E')$ de G é um subgrafo de G acíclico e conexo que contém todos os seus vértices a tal que a soma dos pesos de suas arestas seja mínima.*

4.5.2 Algoritmo de Kruskal

Uma das maneiras de se encontrar a árvore geradora mínima de um grafo é a através de um algoritmo baseado no seguinte teorema:

Teorema *Sabe-se encontrar uma floresta $F = (V, W)$ contendo a arestas, i.e., $|W| = a$, cuja soma de suas arestas é mínima.*

Esse teorema é suficiente porque a árvore geradora mínima nada mais é que uma floresta mínima com $|V| - 1$ arestas. Portanto, para obter a árvore, basta fazer $a = |V| - 1$ no algoritmo derivado do teorema acima. O algoritmo pode ser obtido da seguinte prova:

Prova Por indução em a , o número de arestas. O caso base, $a = 0$, é trivial. Sendo o número de vértices fixo, existe uma única floresta com nenhuma aresta e ela tem peso zero. Como hipótese de indução, suponha que seja possível encontrar uma floresta de peso mínimo com $a = k - 1$ arestas, $1 \leq k < |V|$. Deseja-se provar que o teorema é também verdadeiro para $a = k$. Para isso, utiliza-se o seguinte lema:

Lema *Uma aresta de peso mínimo e de um grafo estará em pelo menos uma floresta com uma ou mais arestas.*

Prova Por contradição. Suponha que e não esteja em nenhuma floresta mínima. Se inserirmos e em uma dessas florestas, podem ocorrer duas situações:

1. cria-se um ciclo no grafo: nesse caso, basta retirar qualquer outra aresta do ciclo para criar uma floresta de custo menor ou igual à original;
2. não se cria um ciclo: esse caso é ainda mais simples, pois a retirada de qualquer aresta da floresta gerará uma solução de custo não superior ao original;

Repare que, em qualquer dos casos, a adição de e cria uma floresta de custo inferior ou igual a uma floresta de peso mínimo, o que contradiz a afirmação de que nenhuma floresta de peso mínimo contém e . \square

Uma vez estabelecido esse lema, pode-se aplicar o passo indutivo. Dado um grafo $G = (V, E)$, cria-se um grafo $G' = (V', E')$ formado a partir da contração das extremidades de uma aresta de peso mínimo e em um único vértice. Em outras palavras, se $e = (v, w)$, os vértices v e w são substituídos por um novo vértice u e todas as arestas incidentes em v e w (com exceção da própria aresta e e possíveis arestas paralelas a ela) passam a ser incidentes em u , mantendo seus pesos originais e a outra extremidade. A floresta que se deseja construir conterá a aresta e mais $k - 1$ arestas do grafo G' , que podem ser encontradas aplicando-se a hipótese de indução. Se alguma das arestas adicionadas pela aplicação da hipótese de indução for adjacente ao vértice u , basta substituí-la por sua versão original, adjacente a v ou w . \square

Dessa prova indutiva, deriva-se o algoritmo apresentado na figura ???. Ele recebe o nome de *Algoritmo de Kruskal*, numa referência a quem primeiro o descreveu.

```

01 function kruskal ( $G$ ): graph {
02      $W = \emptyset$ ; /*lista de arestas da floresta inicialmente vazia*/
03     for  $i \leftarrow 1$  to  $|V| - 1$  do {
04         seleccione  $e \in E - W$  com menor peso tal que
            $F = (V, W \cup \{e\})$  NÃO contenha ciclos; /* $F$ : floresta*/
05          $W \leftarrow W \cup \{e\}$ ;
06     }
07     return  $F = (V, W)$ ;
08 }
```

Figura 4.1: Algoritmo de Kruskal

O algoritmo é apresentado em sua versão iterativa.

Árvore Geradora Máxima O algoritmo para a árvore geradora mínima e sua prova de correteude podem ser facilmente adaptados para construir a árvore geradora máxima. No caso do algoritmo, a modificação é trivial. Basta mudar o critério de ordenação de E na linha 3 da figura ??. As arestas devem ser analisadas na ordem não-crescente (da maior para a menor). Para provar que essa estratégia é correta, basta provar o seguinte teorema:

Teorema *Sabe-se encontrar uma floresta $F = (V, W)$ contendo a arestas (i.e., $|W| = a$) cuja soma dos pesos de suas arestas é máxima.*

O teorema é semelhante ao apresentado para o caso da árvore geradora mínima; sua prova também é essencialmente a mesma. A única diferença está no lema utilizado, que deve tratar da aresta de peso *máximo* em lugar da aresta de peso mínimo. Uma prova por contradição idêntica à anterior, substituindo-se todas as ocorrências de “máximo” por “mínimo” e “superior” por “inferior” é perfeitamente válida.

4.5.3 Algoritmo de Prim

Teorema *Seja $G = (V, E)$ um grafo ponderado não-orientado e $s \in V$ um vértice qualquer desse grafo. É possível encontrar uma árvore com a arestas que contém s e é um subgrafo de uma árvore geradora mínima de G .*

Prova Por indução simples em a . O caso base, $a = 0$, é trivial. Uma árvore contendo apenas o vértice s possui zero aresta e é um subgrafo de qualquer árvore geradora de G . Como hipótese de indução, suponha que o teorema seja válido para $a = k - 1$ arestas ($0 < k < n$). Deseja-se provar que o teorema é válido para $a = k$, ou seja, que é possível encontrar uma árvore A_k com k arestas que contém s e é um subgrafo de uma árvore mínima de G . Pela hipótese de indução, sabe-se encontrar uma árvore A_{k-1} com $k - 1$ arestas que contém s e é um subgrafo de uma árvore mínima de G . Para encontrar A_k , basta adicionar a A_{k-1} uma aresta que garantidamente pertença a uma árvore mínima de G . Seja S_{k-1} o conjunto dos vértices em A_{k-1} , $\bar{S}_{k-1} = V - S_{k-1}$ e E_k o conjunto de arestas que ligam vértices de S_{k-1} a vértices de \bar{S}_{k-1} .

Lema Seja $e_{min} = (v, w)$ a aresta de peso mínimo de E_k . Essa aresta pertence a uma árvore geradora de peso mínimo de G .

Prova Por contradição. Suponha que $e_{min} = (v, w)$ não pertença a nenhuma árvore mínima de G . Seja A uma árvore mínima qualquer de G . Como A é geradora, existe um caminho em A entre v e w . Como $v \in S_{k-1}$ e $w \in \bar{S}_{k-1}$, existe uma aresta f nesse caminho que liga um vértice de S_{k-1} a um vértice de \bar{S}_{k-1} . Por hipótese, seu peso não pode ser menor que o de e_{min} . Portanto, se f for substituída por e_{min} , teremos uma nova árvore A' que contém e_{min} e cujo custo não é superior ao de A , o que é uma contradição. \square

Esse lema garante que a aresta de peso mínimo de E_k pode ser adicionada a A_{k-1} para formar A_k , o que completa o passo indutivo. \square

Observe que esse teorema é suficiente para garantir que a árvore geradora mínima de um grafo pode ser calculada. Afinal, quando $a = |V| - 1$, o subgrafo mencionado no teorema é a própria árvore geradora. O algoritmo derivado dessa prova indutiva é conhecido como *Algoritmo de Prim*.

4.6 Girth

Definição O *girth* de um grafo orientado e ponderado (com pesos positivos, negativos ou nulos) $G = (V, E)$ é definido como o peso do ciclo orientado de peso mínimo em G .

Para determinar o valor de *girth* (G) para um grafo G qualquer, é suficiente provar o seguinte teorema:

Teorema Seja $G = (V, E)$ um grafo orientado e ponderado, e seja $V = \{v_1, v_2, v_3, \dots, v_n\}$. Para todo par ordenado de vértices $(v_i, v_j) \in V \times V$, é possível encontrar o caminho de comprimento mínimo entre v_i e v_j que utiliza como vértices intermediários apenas elementos de $V_p = \{v_1, v_2, \dots, v_p\}$, sendo $0 \leq p \leq n$ ($V_0 \equiv \emptyset$).

Seja $d_p(i, j)$ a distância mínima entre v_i e v_j ($1 \leq i, j \leq n$) usando com vértices intermediários apenas elementos de V_p . O valor de *girth* (G) é simplesmente o valor de $\min_i d_n(i, i)$, sendo $i \in \{1, 2, \dots, n\}$. Portanto, se for provado o teorema acima, estará provado o fato de que se sabe encontrar o *girth* de um grafo qualquer. A prova é apresentada a seguir.

Prova Por indução em p . O caso base é $p = 0$: consideram-se apenas os caminhos diretos, que não utilizam vértice algum como intermediário. Dado um par ordenado (v_i, v_j) , define-se

$d_0(i, j) = c(i, j)$, se existir a aresta (v_i, v_j) ($c(i, j)$ é o custo dessa aresta), ou $d_0(i, j) = \infty$, caso contrário.

Suponha, como hipótese de indução, que o teorema seja válido para $p = k - 1$, sendo $k \geq 1$: sabe-se calcular o valor de $d_{k-1}(i, j)$ para todos os pares de vértices $(v_i, v_j) \in V \times V$. Deseja-se provar que o teorema é válido para $p = k$, ou seja, que é possível calcular as distâncias entre todos os vértices pares de vértices utilizando-se como nós intermediários apenas os nós de V_k . A hipótese de indução garante ser possível calcular essas distâncias restritas aos caminhos que utilizam os vértices V_{k-1} . Como $V_{k-1} \subset V_k$, as distâncias assim obtidas serão um limite superior para as distâncias mínimas usando vértices de V_k . Em alguns casos, pode ser interessante utilizar também o vértice v_k , o único não considerado na aplicação da hipótese de indução. Seja um par ordenado de vértices (v_i, v_j) qualquer. A maneira mais eficiente de se utilizar o vértice v_k no caminho mínimo entre v_i e v_j é percorrendo o caminho mínimo conhecido entre v_i e v_k e, em seguida, o caminho mínimo entre v_k e v_j . Como ambos os “subcaminhos” utilizam apenas os $k - 1$ primeiros vértices, seus comprimentos já terão sido determinados pela aplicação da hipótese de indução. O valor de $d_k(i, j)$ será simplesmente o mínimo entre o caminho que não utiliza v_k (fornecido diretamente pela aplicação da hipótese de indução) e o que o utiliza (calculado a partir de dois componentes também originados da hipótese de indução):

$$d_k(i, j) = \min\{d_{k-1}(i, j), d_{k-1}(i, k) + d_{k-1}(k, j)\}.$$

□

Essa prova trata sem problemas o caso de grafos com arestas de custo negativo. Entretanto, se o grafo possuir também *circuitos* de custo negativo, deixa de fazer sentido a própria noção de circuito de custo mínimo, já que um circuito simples de peso negativo pode ser percorrido diversas vezes para que se crie um caminho de comprimento arbitrariamente pequeno. Circuitos de peso negativo podem ser detectados se, no algoritmo resultante dessa prova, o valor de $d_p(i, i)$ for negativo para algum $p \in \{0, 1, 2, \dots, n\}$ e algum $i \in \{1, 2, \dots, n\}$. Trata-se de um caso especial do problema.

O algoritmo completo (incluindo o tratamento desse caso especial) é apresentado na figura ??.

4.7 Partição: Questão 2 da P3

O problema da partição de um conjunto de objetos em dois conjuntos pode ser enunciado da seguinte forma:

Dado um conjunto de n objetos $O = \{o_1, o_2, \dots, o_n\}$ de valores inteiros v_i ($1 \leq i \leq n$), dividi-lo em dois grupos G_1 e G_2 de forma que a diferença entre os valores dos objetos em cada grupo seja mínima.

O seguinte teorema pode ser obtido diretamente a partir desse enunciado:

Teorema *Sabe-se encontrar uma atribuição de objetos a G_1 e G_2 com diferença mínima para um conjunto O com n objetos.*

É fácil perceber que uma prova por indução simples desse teorema pode envolver uma completa redistribuição dos objetos no passo indutivo. Considere um exemplo muito simples, com quatro

```

01 function girth ( $G$ ): integer {
02   /*inicializacao*/
03   for  $i \leftarrow 1$  to  $|V|$  do {
04     for  $j \leftarrow 1$  to  $|V|$  do {
05       if  $((i, j) \in E)$  {
06          $d(i, j) \leftarrow c(i, j)$ ;
07         if  $((i = j) \textbf{ and } (d(i, j) < 0))$  return  $-\infty$ ; /*ciclo negativo*/
08       }
09       else  $d(i, j) \leftarrow \infty$ ;
10     }
11   }
12   /*caminhos minimos*/
13   for  $k \leftarrow 1$  to  $|V|$  do {
14     for  $i \leftarrow 1$  to  $|V|$  do {
15       for  $j \leftarrow 1$  to  $|V|$  do {
16         if  $(d(i, j) > d(i, k) + d(k, j))$  {
17            $d(i, j) \leftarrow d(i, k) + d(k, j)$ ;
18           if  $((i = j) \textbf{ and } (d(i, j) < 0))$  return  $-\infty$ ; /*ciclo negativo*/
19         }
20       }
21     }
22   }
23   /*calculo do girth*/
24    $girth \leftarrow \infty$ ;
25   for  $i \leftarrow 1$  to  $|V|$  do {
26     if  $(d(i, i) < girth)$   $girth \leftarrow d(i, i)$ ;
27   }
28   return  $girth$ ;
29 }

```

Figura 4.2: Algoritmo para [GIRTH]

objetos cujos pesos são 1, 2, 3 e 4. Uma solução ótima para os três primeiros objetos seria $G_1 = \{1, 2\}$ e $G_2 = \{3\}$ (para tornar mais simples a notação, cada objeto está representado por seu peso). Quando se adiciona o quarto objeto, os conjuntos são rearranjados: $G_1 = \{1, 4\}$ e $G_2 = \{2, 3\}$. Se o quarto objeto tivesse peso 7, uma solução seria $G_1 = \{1, 2, 3\}$ e $G_2 = \{7\}$. De forma geral, a organização obtida para os três primeiros objetos pode não ter relação alguma com a solução obtida para quatro objetos.

Uma solução para esse problema é reforçar a hipótese de indução.

Teorema Dado um conjunto de objetos $O = \{o_1, o_2, \dots, o_n\}$ de valores v_i positivos, é possível encontrar todas as possíveis subdivisões desse conjunto em dois grupos G_1 e G_2 .

Observe que, se forem conhecidas todas as subdivisões em dois subconjuntos, basta percorrê-las para determinar a mais equilibrada.

É possível simplificar o problema por meio de uma simples observação. Na verdade, o problema se resume a encontrar apenas um dos grupos (G_1 , por exemplo), já que o outro estará automaticamente determinado (deverá conter todos os objetos restantes). Assim pode-se assumir que o grupo G_1 será sempre o grupo com objetos cuja soma possui menor valor, e o problema se resume em descobrir se é possível escolher objetos cuja soma é $0, 1, \dots, \lfloor V_n/2 \rfloor$, sendo $\lfloor V_n/2 \rfloor$ o maior valor inteiro cuja soma é inferior ou igual à soma dos valores de todos os objetos dividida por 2. Logo, o maior valor entre $0, 1, \dots, \lfloor V_n/2 \rfloor$ para o qual for possível encontrar um subconjunto de O com tal soma será o que levará à diferença mínima.

O fato de que os **valores dos objetos são inteiros** permite enunciar um novo teorema.

Teorema *Seja $O_K = \{o_1, o_2, \dots, o_K\}$ um conjunto de objetos de valores $v_i, i = 1, 2, \dots, K$. Para cada valor inteiro $x \in \{0, 1, \dots, \lfloor V_n/2 \rfloor\}$, sabe-se determinar se existe ou não um subconjunto de O cuja soma vale x .*

Prova Por indução simples em K . Seja $Existe[x, K]$ igual a 1 se e somente se existir um subconjunto de O_K tal que a soma seja x . O caso base, $K = 0$, é trivial: só existe um subconjunto para o valor $x = 0$. Portanto $Existe[0, 0] = 1$ e $Existe[x, 0] = 0$ para $x = 1, 2, \dots, \lfloor V_n/2 \rfloor$.

Suponha que o teorema seja válido para K . Deseja-se prová-lo para $K + 1$. No passo indutivo, há duas possibilidades:

1. Em O_K , $Existe[x, K] = 1$ e, portanto, em O_{K+1} também existirá um subconjunto com valor x (pode ser que haja outros, mas o próprio subconjunto que existe em O_K também existirá em O_{K+1}); logo, se $Existe[x, K] = 1$, então $Existe[x, K + 1] = 1$.
2. Em O_K , $Existe[x, K] = 0$. Nesse caso, somente existirá um subconjunto em O_{K+1} cuja soma de valores é x se existir um subconjunto cuja soma acrescentada do valor do objeto o_{K+1} (i.e. v_{K+1}) seja x . Logo, $Existe[x, K + 1] = 1$ se $Existe[x - v_{K+1}, K] = 1$; caso contrário, $Existe[x, K + 1] = 0$.

□

Observe que, depois de determinados todos os subconjuntos possíveis, calcular o que determina a divisão mais equilibrada é imediato. Basta escolher aquele cujo valor total mais se aproxima de $V_n/2$. O grupo escolhido pode compor G_1 ; G_2 será formado pelos elementos de $O - G_1$.