MO417 - Ata de exercício (34.2-1)

Matheus Silva Mota - RA100604
23 de junho de 2010

Exercício 34.2-1

Enunciado

Considere a linguagem GRAPH-ISOMORPHISM = $\{\langle G, u, v, k\rangle$: G_1 e G_2 são grafos isomórfos$\}$. Prove que GRAPH-ISOMORPHISM $\in$ NP, descrevendo um algoritmo de tempo polinomial para verificar a linguagem.

Resolução

Antes de apresentar a resolução deste exercício, discutiremos, a seguir, a definição de grafos isomórfos.

Definição de grafos isomórficos

Para que dois grafos $G_1 = (V_1, E_1)$ e $G_2 = (V_2, E_2)$ sejam ditos isomorfos (escreve-se $G_1 \cong G_2$), é intuitivo acreditar que satisfazer as condições apresentadas a seguir seja suficiente.

1. G_1 e G_2 devem ter o mesmo número de vértices, ou seja, $|V_1| = |V_2|$.
2. G_1 e G_2 devem possuir o mesmo número de arestas, ou seja, $|E_1| = |E_2|$.
3. G_1 e G_2 devem possuir mesmo número de vértices com grau[1] n, para qualquer n entre 0 e o número de vértices que o grafo contém.

Porém, perceba que apenas estas três condições não são suficientes para se afirmar que G_1 e G_2 são isomorfos entre si. Por exemplo, os grafos apresentados na Figura 1 respeitam as três condições mas não são isomorfos.

Sendo assim, é preciso definir bem o que é necessário para que dois grafos sejam ditos isomorfos entre si. Segundo [2] e [1], a definição de isomorfismo entre um grafo simples[2] G e um grafo também simples H é uma bijeção $f : V(G) \to V(H)$ de tal forma que $(u, v) \in E(G)$ se e somente $(f(u), f(v)) \in$

[1] O grau de um vértice v em um grafo G também pode ser visto como a quantidade de arestas incidentes em v

[2] Grafo que **não** contém *loops* ou arestas múltiplas

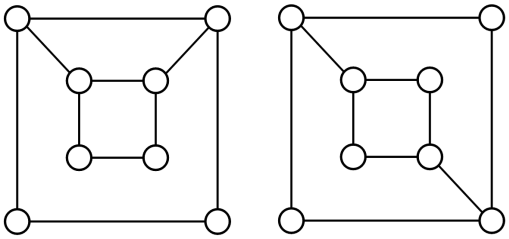

Figura 1: Dois grafos com mesmo número de vértices e arestas, e até mesmos graus, mas que não são isomorfos entre si

$E(H)$. Tratando-se de grafos não-simples[3], o isomorfismo entre G e H é definido como uma bijeção f que mapeia $V(G)$ para $V(H)$ e $E(G)$ para $E(H)$ de tal forma que cada aresta de G com extremidades em u e v é mapeada para uma aresta com extremidades em $f(u)$ e $f(v)$. (**Observação**: *Depois de longo debate em sala, o instrutor Pedro Feijão definiu que a parte de grafos não-simples seria retirada do escopo do problema.*)

Em suma, a função que mapeia os vértices do grafo G_1 para o grafo G_2 deve ser bijetora para que se possa afirmar que os grafos são isomorfos entre si. A definição do conceito de grafos isomorfos é fundamental para a resolução deste exercício que é apresentada a seguir.

Resolução

Sabendo que para afirmar que dois grafos são isomorfos entre si é preciso verificar se a função que mapeia os vértices dos dois grafos é bijetora. O algoritmo de verificação proposto, definido abaixo, possui três argumentos: (i) o grafo $G_1 = (V_1, E_1)$; (ii) o grafo $G_2 = (V_2, E_2)$; (iii) a função f que mapeia os vértices de G_1 para os vértices de G_2. O argumento (iii) é o certificado que o algoritmo deve utilizar para provar que G_1 e G_2 são isomorfos entre si. Sendo assim, o objetivo do algoritmo é verificar, em tempo polinomial, se esta função (certificado) é ou não bijetora, e se conserva adjacências, implicando no isomorfismo ou não de G_1 e G_2.

Para tal, é proposto o seguinte desenho[4] de algoritmo para verificar o certificado.

[3]Grafos que podem possuir *loops* e arestas múltiplas

[4]Não se trata de pseudocódigo, e sim de um esboço do algoritmo

Algoritmo 1 GRAPH-ISOMORPHISM($G_1 = (V_1, E_1)$, $G_2 = (V_2, E_2)$, função de mapeamento f)

```
1: if |V1| ≠ |V2| then
2:     return 'Não são isomorfos'
3: end if
   {Construir g inversa de f, se f for bijetora}
4: for todo v ∈ V2 do
5:     g(v) ← NIL
6: end for
7: for todo v ∈ V1 do
8:     if g(f(v)) = NIL then
9:         g(f(v)) ← v
10:    else
11:        return 'Não são isomorfos'
12:    end if
13: end for
   {Testar arestas de G1 e de G2}
14: for todo e = (u, v) ∈ E1 do
15:    if (f(u), f(v)) ∉ E2 then
16:        return 'Não são isomorfos'
17:    end if
18: end for
19: for todo e = (u, v) ∈ E2 do
20:    if (g(u), g(v)) ∉ E1 then
21:        return 'Não são isomorfos'
22:    end if
23: end for
24: return 'São isomorfos'
```

Em suma, a ideia do algoritmo é checar se os grafos tem o mesmo número de vértices, a seguir construir a inversa de f e finalmente testar a condição de isomorfismo conforme a definição, verificando se existe uma aresta $(u, v) \in E_1$ se e somente se também existe uma aresta $(f(u), f(v)) \in E_2$. Pode-se dizer então que a complexidade do algoritmo é $O(|V_1| + |V_2| + |E_1| + |E_2|)$.

Conclusão

Sabendo-se que todas as verificações podem ser feitas em tempo polinomial, conclui-se que GRAPH-ISOMORPHISM pode ser verificada em tempo polinomial e, portanto, pertence a classe NP.

Referências

[1] Bondy, J. A., and Murty, U. S. R. *Graph theory with applications.* Macmillan, London :, 1976.

[2] West, D. B. *Introduction to Graph Theory (2nd Edition).* Prentice Hall, August 2000.