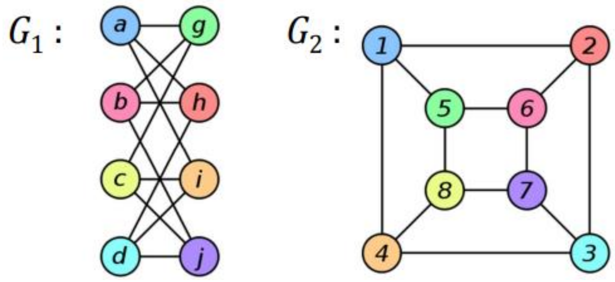
[5] Isomorphism of general graphs and of trees

Isomorfismus grafu

- [**Definice**] Dva grafy $G_1=(V_1,E_1)$ a $G_2=(V_2,E_2)$ jsou izomorfní, existuje-li bijekce f: V1→ V2:

$$\forall\, x, y \in V_1 \;\; : \;\; \{f(x), f(y)\} \in E_2 \;\; \Leftrightarrow \;\; \{x, y\} \in E_1$$

- Zobrazení f je pak isomorfismem mezi G_1 a G_2.



f :
$f(a)=1$
$f(b)=6$
$f(c)=8$
$f(d)=3$
$f(g)=5$
$f(h)=2$
$f(i)=4$
$f(j)=7$

- [**Definice**] Nechť je $\mathcal{F}$ skupina grafů. *Invarianta* $\mathcal{F}$ je funkce Φ s definičním oborem $\mathcal{F}$:

$$\forall\, G_1, G_2 \in \mathcal{F} \;\; : \;\; \Phi(G_1) = \Phi(G_2) \;\; \Leftarrow \;\; G_1 \text{ is isomorphic to } G_2$$

- [**Příklad**]
  - |V| grafu G=(V,E) je invarianta.
  - Sekvence stupňů [$\deg(v_1)$, $\deg(v_2)$, $\deg(v_3)$,…, $\deg(v_n)$,]není invarianta.
  - Pokud by však sekvence výše byla seřazena a neklesající, byla by invariantou.

- [**Definice**] Nechť je $\mathcal{F}$ skupina grafů na množině V, nechť D je funkce s definičním oborem ($\mathcal{F}$ × V). Rozčlenění B_G množiny V indukované D (partition B_G of V induced by D) je pak:

$$B_G = [\, B_G[0], B_G[1], \ldots, B_G[n-1] \,] \text{, kde } B_G[i] = \{\, v \in V \;\; : \;\; D(G,v) = i \,\}$$

  - Je-li funkce $\Phi_D(G)$ invariantou, říkáme, že D je *funkce indikující invariantu*.

$$\Phi_D(G) = [\; |B_G[0]|, |B_G[1]|, \ldots, |B_G[n-1]| \;]$$

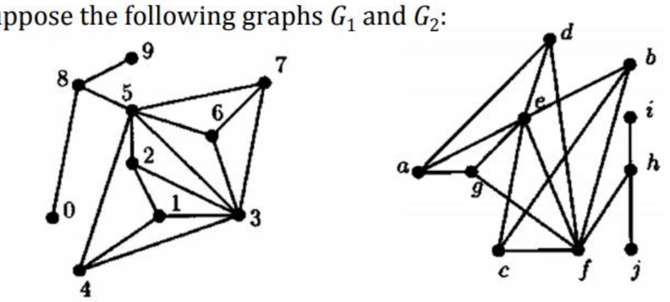
Příklad

Let

- $D_1(G,x)=\deg_G(x)$
- $D_2(G,x)=[d_{j(x)}:\ j = 1, 2, \ldots, \max\{\deg_G(x):\ x \in V(G)\}]$

  where $d_j(x)=|\{y : y \text{ is adjacent to } x \text{ and } \deg_G(y) = j\}|$

Suppose the following graphs G_1 and G_2:



$X_0(\mathcal{G}_1) = \{0,1,2,3,4,5,6,7,8,9\}.$

$X_0(\mathcal{G}_2) = \{a,b,c,d,e,f,g,h,i,j\}.$

x	0	1	2	3	4	5	6	7	8	9
$D_1(\mathcal{G}_1,x)$	1	3	3	6	3	6	3	3	3	1

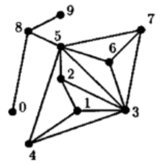
$\Downarrow$

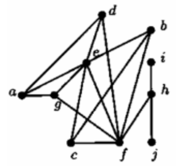
$X_1(\mathcal{G}_1) = \{0,9\},\{1,2,4,6,7,8\},\{3,5\}$

$\bar{x}$	a	b	c	d	e	f	g	h	i	j
$D_1(\mathcal{G}_2,\bar{x})$	3	3	3	3	6	6	3	3	1	1

$\Downarrow$

$X_1(\mathcal{G}_2) = \{i,j\},\{a,b,c,d,g,h\},\{e,f\}.$

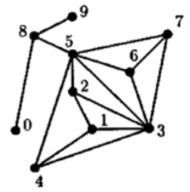




$$
\begin{aligned}
D_2(\mathcal{G}_1,0) &= (0,0,1,0,0,0,0,0,0)\\
D_2(\mathcal{G}_1,1) &= (0,0,2,0,0,1,0,0,0)\\
D_2(\mathcal{G}_1,2) &= (0,0,1,0,0,2,0,0,0)\\
D_2(\mathcal{G}_1,3) &= (0,0,5,0,0,1,0,0,0)\\
D_2(\mathcal{G}_1,4) &= (0,0,1,0,0,2,0,0,0)\\
D_2(\mathcal{G}_1,5) &= (0,0,5,0,0,1,0,0,0)\\
D_2(\mathcal{G}_1,6) &= (0,0,1,0,0,2,0,0,0)\\
D_2(\mathcal{G}_1,7) &= (0,0,1,0,0,2,0,0,0)\\
D_2(\mathcal{G}_1,8) &= (2,0,0,0,0,1,0,0,0)\\
D_2(\mathcal{G}_1,9) &= (0,0,1,0,0,0,0,0,0)
\end{aligned}
$$

$\Downarrow$

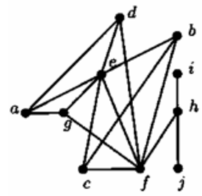
$X_2(\mathcal{G}_1) = \{0,9\},\{8\},\{2,4,6,7\},\{1\},\{3,5\}.$



$$
\begin{aligned}
D_2(\mathcal{G}_2,a) &= (0,0,2,0,0,1,0,0,0)\\
D_2(\mathcal{G}_2,b) &= (0,0,1,0,0,2,0,0,0)\\
D_2(\mathcal{G}_2,c) &= (0,0,1,0,0,2,0,0,0)\\
D_2(\mathcal{G}_2,d) &= (0,0,1,0,0,2,0,0,0)\\
D_2(\mathcal{G}_2,e) &= (0,0,5,0,0,1,0,0,0)\\
D_2(\mathcal{G}_2,f) &= (0,0,5,0,0,1,0,0,0)\\
D_2(\mathcal{G}_2,g) &= (0,0,1,0,0,2,0,0,0)\\
D_2(\mathcal{G}_2,h) &= (2,0,0,0,0,1,0,0,0)\\
D_2(\mathcal{G}_2,i) &= (0,0,1,0,0,0,0,0,0)\\
D_2(\mathcal{G}_2,j) &= (0,0,1,0,0,1,0,0,0)
\end{aligned}
$$

$\Downarrow$

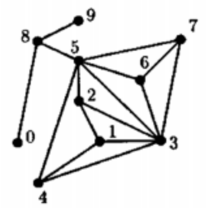
$X_2(\mathcal{G}_2) = \{i,j\},\{h\},\{b,c,d,g\},\{a\},\{e,f\}.$



This restricts a possible isomorphism to bijections between the following sets:

$$
\begin{aligned}
\{0,9\} &\longleftrightarrow \{i,j\}\\
\{8\} &\longleftrightarrow \{h\}\\
\{2,4,6,7\} &\longleftrightarrow \{b,c,d,g\}\\
\{1\} &\longleftrightarrow \{a\}\\
\{3,5\} &\longleftrightarrow \{e,f\}
\end{aligned}
$$

There are 96 = (2!)(1!)(4!)(1!)(2!) bijections giving the possible isomorphisms. Examination of each of these possible isomorphisms shows that only the following eight bijections are isomorphisms.

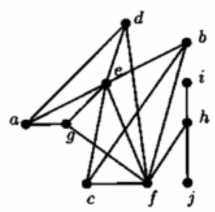


$$\begin{pmatrix}0&1&2&3&4&5&6&7&8&9\\ i&a&d&e&g&f&b&c&h&j\end{pmatrix} \quad \begin{pmatrix}0&1&2&3&4&5&6&7&8&9\\ j&a&d&e&g&f&b&c&h&i\end{pmatrix}$$

$$\begin{pmatrix}0&1&2&3&4&5&6&7&8&9\\ i&a&d&e&g&f&c&b&h&j\end{pmatrix} \quad \begin{pmatrix}0&1&2&3&4&5&6&7&8&9\\ j&a&d&e&g&f&c&b&h&i\end{pmatrix}$$

$$\begin{pmatrix}0&1&2&3&4&5&6&7&8&9\\ i&a&g&e&d&f&b&c&h&j\end{pmatrix} \quad \begin{pmatrix}0&1&2&3&4&5&6&7&8&9\\ j&a&g&e&d&f&b&c&h&i\end{pmatrix}$$

$$\begin{pmatrix}0&1&2&3&4&5&6&7&8&9\\ i&a&g&e&d&f&c&b&h&j\end{pmatrix} \quad \begin{pmatrix}0&1&2&3&4&5&6&7&8&9\\ j&a&g&e&d&f&c&b&h&i\end{pmatrix}$$



Algoritmy

```
Function FINDISOMORPHISM (set of invariant inducing functions I; graph G1, G2) : set of isomorphisms
try {
    ( N, X, Y ) = GETPARTITIONS (I, G1, G2) ;
}
catch ("G1 and G2 are not isomorphic!") { return ∅ ; }
for i = 0 to N – 1 do {
    for each x ∈ X[i] do {
        W[x] = i ;
    }
}
return COLLECTISOMORPHISMS(G1, G2, 0, Y, W, f)
```

```
Function GetPartitions (set of invariant inducing functions I; graph G_1; graph G_2) : (number of partitions, paritition of G_1, parititions of G_2)
X[0] = vertices of G_1;  Y[0] = vertices of G_2;  N = 1;
for each D ∈ I do {
    P = N;
    for i = 0 to P − 1 do {
        Partition X[i] into sets X_1[i], X_2[i], X_3[i], … , X_m[i] where x,y∈X_j[i] ⇔ D(G_1,x)=D(G_1,y);
        Partition Y[i] into sets Y_1[i], Y_2[i], Y_3[i], … , Y_n[i] where x,y∈Y_j[i] ⇔ D(G_2,x)=D(G_2,y);
        if n ≠ m then throw exception "G_1 and G_2 are not isomorphic!";
        Order Y[i] into sets Y_1[i], Y_2[i], Y_3[i], … , Y_n[i] so that
            ∀x ∈ X[i], ∀y ∈ Y[i] : D(G_1,x) = D(G_2,y) ⇔ x ∈ X_j[i] and y ∈ Y_j[i];
        if ordering is not possible then throw exception "G_1 and G_2 are not isomorphic!";
        N = N + m − 1;
    }
    Reorder the partitions so that: |X[i]|=|Y[i]| ≤ |X[i+1]|=|Y[i+1]| for 0 ≤ i < N − 1;
}
return (N, X, Y)
```

```
Function CollectIsomorphisms (graph G_1, G_2; starting vertex of G_1 v; paritition of G_2 Y; partition mapping W as array [vertices of G_1] of indices of partitions of G_1; current isomorphism f as array [vertices of G_1] of vertices of G_2) : set of isomorphisms
if v = number of vertices of G_1 then return {f};
R = ∅;
p = W[v];
for each y ∈ Y[p] do {
    OK = true;
    for u = 0 to v − 1 do {
        if ( ({u,v} ∈ edges of G_1 and {f[u], y} ∉ edges of G_2)
             or
             ({u,v} ∉ edges of G_1 and {f[u], y} ∈ edges of G_2) ) then OK = false;
    }
    if OK then {
        f[v] = y;
        R = R ∪ CollectIsomorphisms(G_1, G_2, v+1, Y, W, f);
    }
}
return R
```

Certifikát

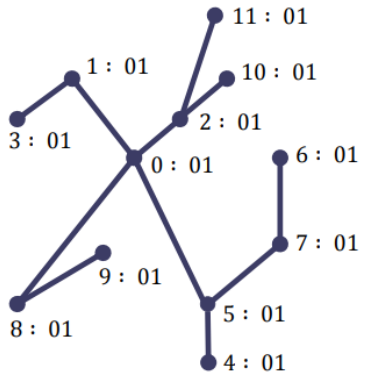
- [**Definice**] Certifikát Cert pro skupinu grafů $\mathcal{F}$ je funkce taková, že:

$$\forall\, G_1, G_2 \in \mathcal{F} \;:\; Cert(G_1) = Cert(G_2) \;\Leftrightarrow\; G_1 \text{ is isomorphic to } G_2$$

- nejrychlejší současné algoritmu pro výpočet isomorfismu v grafech jsou založeny na výpočtu certifikátů
- ten funguje nejen pro obecné grafy, ale i pro některé jejich třídy (např. stromy)

Výpočet certifikátu stromu

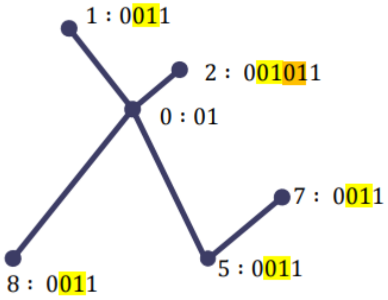
1.) Všechny vrcholy G si označíme řetězcem *01*.
2.) Dokud máme více jak vrcholy G, pro každý vrchol, který není listem proveďme:
    a. Nechť *Y* je množina označení (labels) listů sousedících s *x* a label *x*. Počáteční *01* z *x* odstraníme.
    b. Label *x* nahradíme spojením labelů v *Y* ve vzestupném lexikografickém pořadí, s předponou *0* a příponou *1*.
    c. Odstraníme všechny listy sousedící s *x*.
3.) Zbývá-li jen jeden vrchol, vraťme label *x* jako certifikát.
4.) Zbývají-li dva vrcholy *x* a *y*, vraťme labely *x* a *y*, spojené ve vzestupném lexikografickém pořadí, jako certifikáty.



number of vertices: 12

non−leaves vertices:

$0:\ Y = \langle\ \ \rangle$
$1:\ Y = \langle 01\rangle$
$2:\ Y = \langle 01,01\rangle$
$5:\ Y = \langle 01\rangle$
$7:\ Y = \langle 01\rangle$
$8:\ Y = \langle 01\rangle$



number of vertices: 6

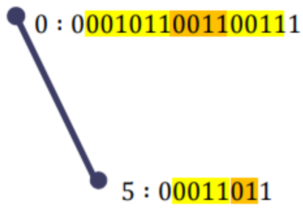
non−leaves vertices:

$$0:\ Y = \left\langle \begin{matrix} 001011, \\ 0011, \\ 0011 \end{matrix} \right\rangle$$

$$5:\ Y = \left\langle \begin{matrix} 0011, \\ 01 \end{matrix} \right\rangle$$

---

number of vertices: 2



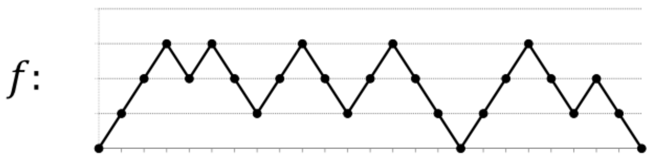
Certificate=$\overbrace{0001011001100111}\overbrace{00011011}$

- **vlastnosti certifikátu:**
    - délka je 2*|V|
    - počet 1 a 0 je stejný
    - počet 1 a 0 je stejný pro všechny subsekvence vzniknuvší z každého labelu vrcholu (během celé doby běhu algoritmu)

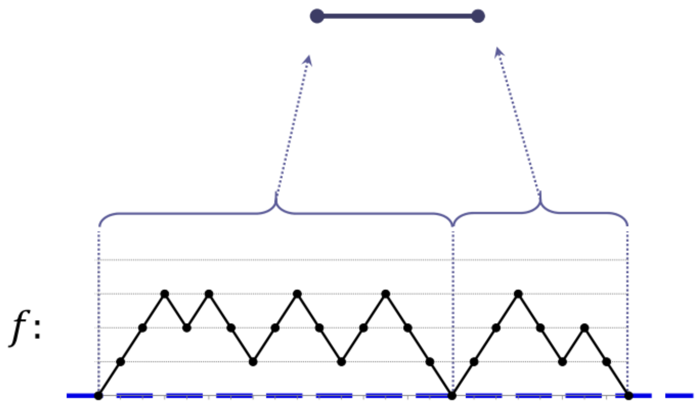
Rekonstrukce stromu z certifikátu

$$f(0) = 0$$

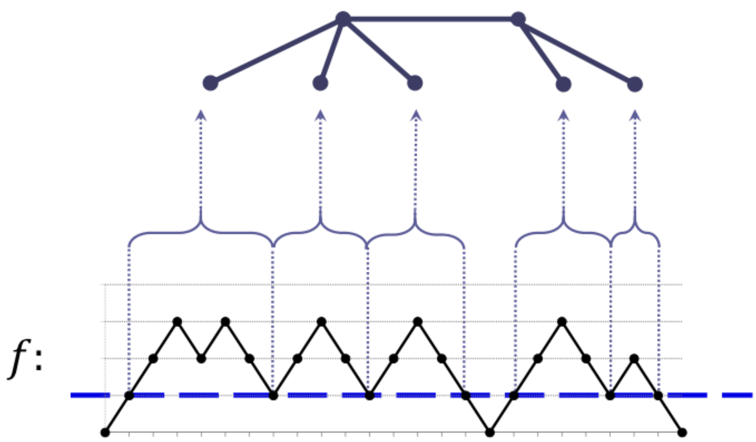
$$f(x+1) = \begin{cases} f(x) + 1, & Cert(G)[x] = 0 \\ f(x) - 1, & Cert(G)[x] = 1 \end{cases}$$



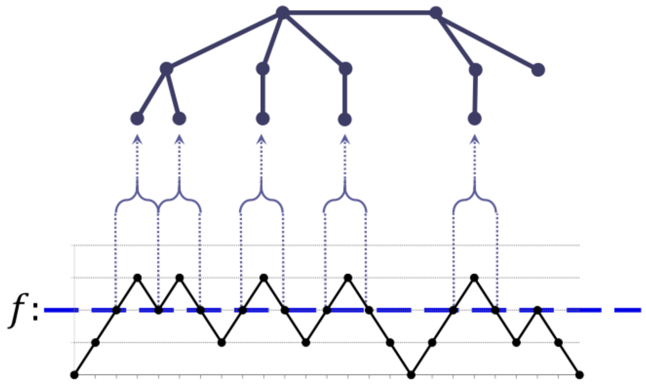
$$Cert(G) = 000101100110011100011011$$



$$Cert(G) = 000101100110011100011011$$



$$Cert(G) = 000101100110011100011011$$



$Cert(G) = 000101100110011100011011$

Algoritmy

```
Function FindSubMountains(integer l, certificate as string C) : number of submountines in C
k = 0;  M[0] = the empty string;  f = 0;
for  x = l − 1  to  |C| − l  do  {
        if  C[x] = 0  then { f = f + 1; } else { f = f − 1; }
        M[k] = M[k] · C[x];
        if  f = 0  then { k = k + 1;   M[k] = the empty string;   f = 0; }
}
return  k;
```

```
Function CertificateToTree (certificate as string C) : tree as G = (V, E)
n = |C|/2;  v = 0;  (V, E) = empty graph of order n;   V = {0, ..., n − 1};
k = FindSubMountains(1, C);
if  k = 1  then  { Label[v] = M[0];  v = v + 1; }
   else { Label[v] = M[0];  v = v + 1; Label[v] = M[1];  v = v + 1;  E = E ∪ {{0,1}}; }
for  i = 0  to  n − 1  do  {
if  |Label[i]| > 2  then {
        k = FindSubMountains(2, Label[i]);  Label[i] = "01";
        for  j = 0  to  k − 1  do  { Label[v] = M[j];  E = E ∪ {{i, v}};  v = v + 1; }
}
return  G = (V, E);
```

$O(|C|^2)$

```
Function FastCertificateToTree (certificate as string C) : tree as G = (V, E)
(V, E) = empty digraph of order |C|/2;   V = {0, ..., |C|/2};
n = 0;
p = n;
for  x = 1  to  |C| − 2  do  {
        if  C[x] = 0  then {
            n = n + 1;
            E = E ∪ {(p, n)};
            p = n;
        } else  {
            p = parent†(p);
        }
}
return  G = (V, remove_orientation(E));
```

† $parent(x)$ returns the parent of a node x. It returns x in the case where x has no parent.

Zdroj: https://cw.fel.cvut.cz/old/_media/courses/a4m33pal/pal07.pdf