QM3

Téma 36.

Zjišťování izomorfizmu mezi grafy

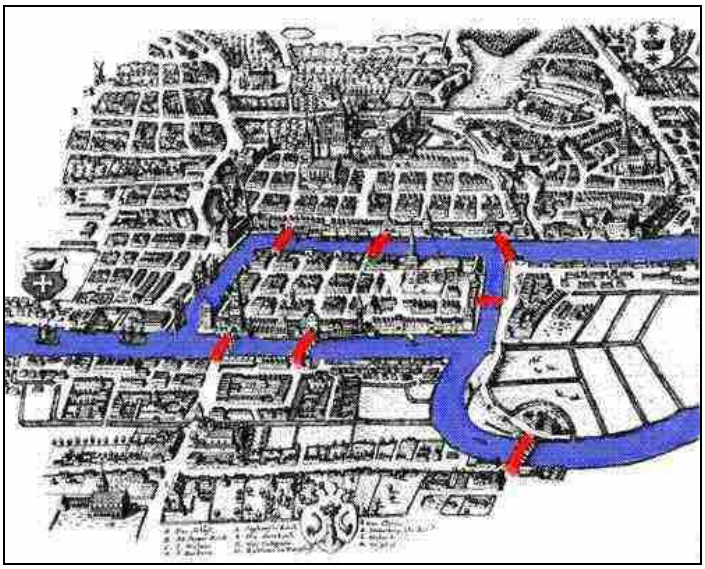
Petr Chmelař

Obsah

1 Úvod

„Co je to graf?“ Na tuto otázku formálně odpovídá druhá kapitola, nicméně na úvod předkládám neformální seznámení s teorií grafů prostřednictvím důležitých milníků.

Tradičně se za zakladatele teorie grafů považuje Leonhard Euler[1], který v Solutio problematis ad geometriam situs pertinentis. *In Commetarii Academiae Scientiarum Imperialis Petropolitanae 8* z roku 1736 na příkladě úlohy sedmi mostů Königsbergu – předložil k diskuzi, jestli je možné projít se po Königsbergu (Obrázek 1.1) tak, že se po každém z jeho mostů jde právě jednou a vrátí se na stejné místo – vytvořil topologii a podmínky nezbytné k takovéto procházce.



Obrázek 1.1: Sedm mostů Königsbergu (převzato z [SA98])

V roce 1845 publikoval Gustav Kirchhoff zákony, které platí v elektrických obvodech a slouží k výpočtu napětí a proudu v jednotlivých větvích obvodu. V teorii grafů našly své uplatnění při studiu tzv. toků v sítích.

Posléze, v roce 1852 předložil Francis Gurthire takzvaný problém čtyř barev – otázku, proč je možné obarvit libovolnou mapu pomocí nejvýše čtyř barev tak, aby každé dvě sousední země (které mají společnou hranici delší než jediný bod) měly odlišnou barvu. Tento problém byl vyřešen až o více než sto let později, v roce 1976 Kennethem Appelem a Wolfgangem Hakenem, přičemž při jeho řešení bylo zavedeno mnoho zásadních konceptů teorie grafů, zejména těch rovinných.

V roce 1936 Dénes König publikoval knihu *Theorie der endlichen und unendlichen Graphen*, ve které systematizoval teorii grafů. V této době už byl graf chápán

[1] Leonhard Euler 15. 4. 1707 (Basel) – 18. 9. 1783 (St. Petersburg), na univerzitě v Basileji měl studovat teologii, ale studoval fyziku, astronomii, medicínu a matematiku prostřednictvím Johanna Bernoulli. Jeho téměř 800 publikací z působení v Petrohradě a Berlíně zasahují do optiky, mechaniky, elektřiny a magnetismu, ale jeho největší přínos je v geometrii a matematice obecně, zejména její systematizací, zavedením *e, i, π, Σ, ∞, sin, cos, ∫a f(x)*.

jednoduše jako množina vrcholů a hran, které je propojují. Zřejmě tato jednoduchost, oproti náročnosti aplikace způsobila, že se problémy z teorie grafů začaly ubírat z obecně teoretické roviny spíše k nástrojům počítačového. To vedlo k vytvoření oboru, někdy nazývaného experimentální matematika, která využívá namísto některých důkazů algoritmy.

Moderní trendy a problémy ukazují, že tento přechod byl oprávněný, protože pomohl vyřešit například některé z 50 problémů shrnutých v [BM82]. Jsou reprezentovány například statistickými metodami určujícími průměrný stupeň vrcholů, které vyvstaly z problému obarvení a propojují je s teorií invariantů. Dále sem patří věta o pravidelnosti, extrémní a Ramseyho teorie grafů nebo dekompozice stromů. Tyto a další problémy jsou podrobněji popsány v [Di00, Wi06], v této práci se omezím na izomorfizmus a nastíním některé spřízněné oblasti.

„Co je to izomorfizmus?“ Izomorfizmus mezi grafy může být neformálně chápán jako shoda jejich struktury, respektive jejich identita až na označení. Někdo by mohl namítnout, že je to jednoduché, ale opak je pravdou, protože algoritmus pro obecný izomorfizmus grafů spadá do tzv. NP problémů, u kterých není známo řešení v polynomiálním (rozumném) čase. Proto je nutné nalézt algoritmy, které dokáží částečně obejít jeho obrovskou časovou a paměťovou náročnost.

„Jaké má použití?“ Teorie grafů spojuje algebru a geometrii. Už to samo o sobě stačí, ale proč se tak rychle rozvíjí? Hlavní místem zájmu o co nejefektivnější metody jsou oblasti, kde je třeba zpracovávat obrovské množství dat, pro jejich indexování a rychlé zpřístupnění, stanovení vazeb a nalézání nových informací… například databáze, bioinformatika, počítačové sítě, elektronické obvody, v chemii pro nalézání nových sloučenin nebo dopravní inženýring a geografii. Vlastně asi všechny vědní a technické, ale i ekonomické obory.

Mým speciálním zájem je pravděpodobnostní modelování v počítačovém vidění. Zde je možné definovat několik málo sémantických objektů, jako je člověk nebo automobil. Ty ale generují obrovskou množinu syntaktických modelů, zejména pokud uvažujeme jejich možný morfizmus – každý objekt v reálném světě má 6 stupňů volnosti, ale lidské tělo samo více než 100 – je nesmírně obtížné v obrovském množství vizuální informace, jako je video se ztrátou jednoho reálného rozměru, zjistit jejich vztah. Zřejmě nejspolehlivější způsob, kterým lze obě strany mince spojit, je metodou konečných prvků modelovat objekty, jejich perspektivní projekcí a následným porovnáním s extrahovanými vlastnostmi stanovit, zda se v obraze skutečně jedná o daný objekt a zjistit jeho stav. Přidáním časové osy a vhodné databáze získáme informace požadované pro některé bezpečnostní aplikace, jako jsou kamerové systémy letišť.

„Jak lze tedy izomorfizmus zjistit?“ Tomu je věnován zbytek této práce, která má dvě tváře – obecnou a neformální, kterou se snažím nastínit problémy, jejichž formalizace by značně překročila rozsah této práce poznámkami pod čarou nebo odkazy na literaturu.

2 Formální úvod

„Jak zní definice grafu a jejich izomorfizmu?“ V této části jsou uvedeny základní pojmy nezbytné pro dobré chápání této práce a matematický základ algoritmů pro izomorfizmus a jeho efektivní zjištění.

Jako prerekvizita této práce jsou považovány přednášky předmětu QA3, Prof. Václava Havla, UMAT FEKT VUT v Brně. Prerekvizity, které jsou pro tuto část třeba, je ale možné nalézt také v [Ko84].

Vzhledem k povaze této práce se nicméně nesnažím o přesné znění všech definic, výčet vlastností, vět, ani jejich důkazů, které lze najít v učebnicích, jako jsou [BM82] nebo [Di00].

2.1 Graf

„Co je to graf?“ Graf je jednoduše množina bodů a spojnic mezi nimi:

Graf definujeme jako trojici $G=(V, E, I)$, kde V nazýváme **množina vrcholů**[2], E **množina hran**[3] grafu G; V je neprázdná; V, E konečné, vzájemně disjunktní množiny a I je **incidenční relace**, pro kterou platí:

$$\mathrm{I} : E \to \binom{V}{2} \tag{2.1}$$

pokud se jedná o zobrazení prosté, jde o graf (zjednodušujeme na dvojici $G=(V, E)$).

Pro nejběžnější **jednoduchý**[4] **graf** platí neostrá inkluze:

$$E \subseteq \binom{V}{2} \tag{2.2}$$

Existují ještě jiné typy grafů. **Obecné grafy** připouštějí i smyčky, tedy také $E \subseteq V$.

Orientované[5] **grafy** mají všechny hrany orientované, jsou reprezentovány uspořádanou dvojicí $E \subseteq V \times V$.

Pro **multigrafy** platí obecná relace $E \to V \times V$ a připouštějí tedy i násobné hrany.

V literatuře se lze setkat také s jinými definicemi, incidenční relaci lze definovat například výčtem. Nebo není uvažována konečnost V, E u **nekonečných grafů**, což není pro zjišťování izomorfizmu přijatelné, proto je nebudeme uvažovat.

Doplním některé dále použité pojmy, jejichž lepší vysvětlení i s výčtem vlastností je možné najít v každé dobré učebnici teorie grafů:

---

[2] též uzly, anglicky vertices, nodes nebo points

[3] anglicky edges nebo lines

[4] také prostý, obyčejný, anglicky simple

[5] nebo digrafy

Sledem v grafu $G=(V, E)$ rozumíme posloupnost vrcholů v_i a hran e_i $P = (v_1, e_1, v_2, \ldots e_{n-1}, v_n)$ pro kterou platí $e_i = \{v_i, v_{i+1}\}$, případně $e_i = (v_i, v_{i+1})$ pro orientovaný graf. Pokud se ve sledu neopakují vrcholy a tedy ani hrany, mluvíme o **cestě** z v_1 do v_n. V případě, že je cesta uzavřená, tedy $v_1 = v_n$, jde o **kružnici**[6]. Počet hran nejkratší cesty z v_1 do v_n se nazývá **vzdálenost** uzlů v_1 a v_n.

Graf je **souvislý**, právě když pro každé dva uzly existuje cesta, která je spojuje. U orientovaného grafu se souvislost po deorientaci nazývá slabá, o silné souvislosti mluvíme v případě, že pokud pro každé dva vrcholy u, *v* existuje cesta z *v* do *v* i z *v* do *u*. V jiném případě je graf nesouvislý. **Komponenta** je v nesouvislém grafu každý maximální souvislý podgraf (viz. níže).

Úplný graf[7] je jednoduchý graf, v němž jsou každé dva vrcholy spojené hranou, pro počet vrcholů *x* se značí K_x. Pokud se jedná největší takový podgraf (viz. níže) nazývá se **klika**.

Bipartitní graf se označuje graf, jehož množinu vrcholů je možné rozdělit na dvě disjunktní množiny tak, že vrcholy ze stejné množiny nejsou spojeny hranami.

Značení vrcholů **grafu** *G* je jednoznačné přiřazení značek – čísel či barev – každému jeho vrcholu. Lze možné je definovat jako bijekci:

$$\upsilon: \{1, 2, \ldots |V|\} \rightarrow V \tag{2.3}$$

Obdobně se tvoří značení hran grafu.

2.1.1 Skóre grafu

„Jaké skóre má graf?“ Každý graf má skóre, je to posloupnost stupňů jeho vrcholů:

Stupeň[8] **vrcholu** $v \in V$ jednoduchého grafu $G=(V, E)$ je nezáporné celé číslo, reprezentuje počet hran $|E(v)|$, které incidují s vrcholem *v*, označuje se jako $d_G(v)$.

Vrchol $v \in V$ se nazývá **izolovaný**, pokud platí: $d_G(v) = 0$.

U orientovaných grafů existují dva typy (polo)stupňů – příchozí $d^+_G(v)$ a odchozí $d^-_G(v)$ reprezentované kladným, resp. záporným celým číslem, nebo *0*.

Skóre grafu je nerostoucí posloupnost stupňů vrcholů jednoduchého grafu *G*:

$$D = (d_G(v), v \in V) \tag{2.4}$$

Způsob, jakým lze rozhodnout, zda je daná posloupnost skórem grafu, ukazuje věta V. Havla, například v [WM03].

U orientovaných grafů je skóre reprezentováno souborem dvojic $(d^+_G(v), d^-_G(v))$.

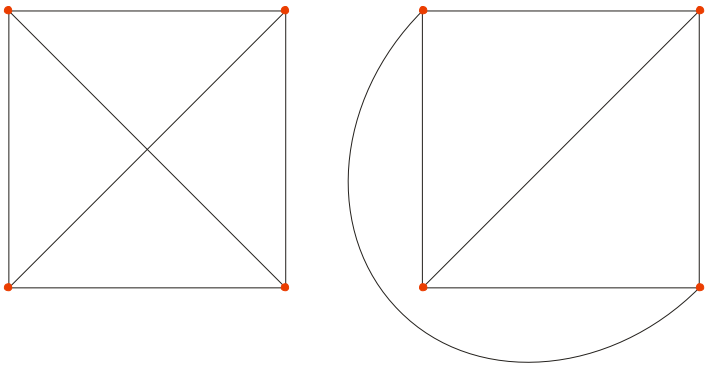
[6] nebo cyklus

[7] také kompletní

[8] též valence, anglicky degree

2.2 Reprezentace grafů

„Proč se grafy nazývají grafy?“ Grafy jsou takto zvány vzhledem ke grafické povaze jejich zobrazení – vrcholy jako body v prostoru a hrany jako křivky, které některé z nich spojují. Nicméně neexistuje jednoznačný způsob, jak grafy kreslit, příkladem je Obrázek 2.1.



a) b)

Obrázek 2.1: Úplný graf K_4 a), zakreslený i jako rovinný b).

Úplný graf K_4 je tzv. **rovinný**[9] **graf**, protože je možné jej zakreslit tak, že se jeho hrany v Euklidově rovině nekříží.

Tento problém řeší **matice adjacence**[10] konečného grafu $G=(V, E)$ s $n=|V|$ vrcholy je matice neorientovaného, respektive orientovaného grafu G velikosti $n \times n$: $A(G) = [a_{ij}]$:

$$a_{ij} = \begin{cases} 1, \text{ pokud } \{v_i, v_j\} \in E, \text{ respektive } (v_i, v_j) \in E \\ 0 \text{ jinak} \end{cases} \quad (2.5)$$

hodnoty diagonály a_{ii} reprezentují smyčky a jsou v případě prostého grafu nulové. Neorientovaný graf má matici adjacence symetrickou.

Matice incidence[11] prostého grafu G s n vrcholy a m hranami $B(G) = [a_{ij}]$ typu (n, m):

$$b_{ij} = \begin{cases} 1, \text{ pokud } \{v_i, v_j\} \in E \\ 0 \text{ jinak} \end{cases} \quad (2.6)$$

2.3 Izomorfizmus

„Co znamená izomorfizmus grafů?“ Mnoho grafů se liší pouze způsobem kreslení a zejména označením svých vrcholů a hran, což vystihuje pojem izomorfizmus:

[9] též planární

[10] také matice sousednosti

[11] nebo matice souvislosti

Dva grafy $G=(V, E)$ a $G'=(V', E')$ jsou **izomorfní**, zapisujeme $G \cong G'$, pokud existuje bijektivní zobrazení (**izomorfizmus**):

$$\varphi: V \to V' \text{ takové, že platí:}$$
$$(v_i, v_j) \in E \Leftrightarrow (\varphi(v_i), \varphi(v_j)) \in E' \tag{2.7}$$

tedy zobrazení zachovává vrcholy i hrany dané incidenční relací z definice.

Problémem rozhodnutí o izomorfizmu se zabývá 3. kapitola, ale zde bych chtěl ještě uvést odhad počtu neizomorfních grafů dle [Wi03]:

Na dané n-prvkové množině V ($n=|V|$) je $2^{\binom{n}{2}}$ různých grafů, ale těch, které nejsou izomorfní je podstatně méně. Například pro $n=3$ existuje 8 grafů, z toho jen 4 nejsou izomorfní. Určit přesný počet neizomorfních grafů je obtížné, ale jsme schopni stanovit odhady: horní $2^{\binom{n}{2}}$ a dolní $\frac{2^{\binom{n}{2}}}{n!}$. Abychom se přesvědčili, že funkce neroste rychleji než $2^{\binom{n}{2}}$, odhady zlogaritmujeme a upravíme:

$$log_2\left(2^{\binom{n}{2}}\right) = \binom{n}{2} = \frac{n^2}{2}\left(1-\frac{1}{n}\right),$$
$$log_2 \frac{2^{\binom{n}{2}}}{n!} = \binom{n}{2} - log_2\, n! \geq \frac{n^2}{2} - \frac{n}{2} - n\, log_2\, n = \frac{n^2}{2}\left(1-\frac{1}{n}-\frac{2\, log_2\, n}{n}\right). \tag{2.8}$$

Vidíme, že pro větší n se logaritmy obou funkcí chovají nejhůře jako $\frac{n^2}{2}$, což znamená velké množství neizomorfních grafů.

Automorfizmus je izomorfizmus grafu na sebe sama, píšeme $G = G'$. Často nerozlišujeme izomorfní grafy, například předpokládáme, že každý úplný graf K_5 je totožný.

O **homomorfizmus** se jedná i když zobrazení φ v (2.7) není bijekce. Pokud to nechceme více komplikovat, stačí říci, že jde o zobrazení vrcholů na vrcholy, hran na hrany při zachování matice incidene, zapisuje se $G \sim G'$.

2.3.1 Podgrafy

Podgraf nebo subgraf je graf, který je izomorfní s nějakou částí nadgrafu, tedy pro φ v (2.7) je postačující injektivní zobrazení, respektive:

Říkáme, že graf $G'=(V', E')$ je **podgrafem** grafu $G=(V, E)$, pokud platí:

$$V' \subseteq V \quad a \quad E' \subseteq E \tag{2.9}$$

Posledním typem izomorfizmu, který zmíním, není ve skutečnosti klasický izomorfizmus: Pokud je možné (souvislý) graf rozložit na podgrafy, spojené jediným vrcholem a neprochází-li napříč jimi žádná (další) kružnice daného grafu G tak, že po úplné separaci vznikne nesouvislý graf, skládající se z dále

nerozložitelných komponent, tyto komponenty se poté nazývají **bloky** a graf **separabilní**.

Pokud jsou všechny bloky grafu *G* izomorfní s bloky grafu *G'*, jsou grafy tzv. ***1*-izomorfní**.

2.4 Invarianty grafu

„Co je to invariant grafu?“ Invarianty jsou základní vlastnosti izomorfních grafů.

Pokud máme grafy $G=(V, E)$ a $G'=(V', E')$ a chceme-li určit, zda jsou izomorfní, požadujeme funkci splňující následující vlastnosti:

(i) jestliže platí $\gamma(G) = \gamma(G')$, pak *G* je izomorfní s *G'*

(ii) jestliže *G* je izomorfní s *G'*, pak také platí $\gamma(G) = \gamma(G')$

Výsledek funkce γ, zvané někdy kódování, je tzv. **invariant grafu**, reprezentovaný číslem, polynomem nebo maticí. Příklady invariantů splňující (i) jsou:

- **Počet vrcholů** grafu *G:* $n = |V|$.
- **Počet hran** $m = |E|$.
- **Nejvyšší a nejnižší stupeň** grafu *G:* $\Delta(G)$ *a* $\delta(G)$.
- **Průměrný stupeň**: $d(G)$, platí: $\Delta(G) \geq d(G) \geq \delta(G)$.
- **Skóre grafu** je souhrnný invariant, z něhož je možné vyvodit výše uvedené.

2.4.1 Matice adjacence

Matice adjacence je další zajímavý invariant. Matice sama o sobě splňuje podmínku (ii), ale ne (i). Nicméně, pokud máme adjacenční matice $A(G)$ a $A(G')$, pak grafy *G* a *G'* jsou izomorfní pokud existuje permutační matice *P* a platí:

$$PA(G)P^{-1} = A(G') \tag{2.10}$$

To znamená, že $A(G)$ a $A(G')$ jsou stejné až na permutaci jejich řádků a sloupců. Toto je nutná a postačující podmínka pro izomorfizmus grafů (i), (ii) a zároveň vodítko, jak izomorfizmus zjistit – najít permutační matici.

Zároveň je z matice adjacence možno odvodit kompletní množinu vlastních hodnot, zvanou **spektrum grafu**, splňujících podmínku (i), například stejné minimální a charakteristické polynomy, determinant i stopu[12].

Matice incidence má podobné vlastnosti jako matice adjacence, navíc ale pro hodnost této matice platí zajímavý vztah:

$$h(B(G)) = |U| - p \tag{2.11}$$

kde *p* je počet komponent. Matice incidence ale neumožňuje tyto komponenty určit a pro určení izomorfizmu existuje $m!n!$ permutací namísto $m!$ adjacenční matice, proto

[12] anglicky trace, německy spur – součet hodot na diagonále, které reflektují smyčky grafu

se pro izomorfizmus nevyužívá. Navíc se pomocí matice incidence špatně zaznamenávají smyčky a tím pádem i je vyloučena práce s orientovanými grafy a multigrafy.

2.4.2 Matice dosažitelnosti

Matice dosažitelnosti grafu G je jako matice adjacence $R(G)=[r_{ij}]$ velikosti $n \times n$:

$$r_{ij} = \begin{cases} 1, \text{ pokud existuje cesta z } v_i \text{ do } v_j \\ 0 \text{ jinak} \end{cases} \quad (2.12)$$

její konstrukce je popsána v [Ko84]. Matice dosažitelnosti umožňuje přímočarým způsobem určit spojité komponenty grafu, oddělit je a zjišťování izomorfizmu aplikovat na tyto bloky.

Podobné vlastnosti má i **vzdálenostní matice** $D(G)=[d_{ij}]$:

$$d_{ij} = \text{ vzdálenost z } v_i \text{ do } v_j \quad (2.13)$$

konstrukce se provádí pomocí Dijkstrova [Ko84], nebo lépe Floyd-Warshallova [Wi06] algoritmu. Matice vzdáleností produkuje spektrum s obdobnými vlastnostmi jako adjacenční matice. Také výrazným způsobem může omezit počet možných permutací adjacenční matice pro zjištění izomorfizmu.

2.4.3 Další invarianty

Další invarianty jsou zmíněny pouze pro zajímavost, v literatuře je možné jich nalézt až 50. Nalezení dalších invariant má také často stejnou´, nebo ještě větší složitost, než grafový izomorfizmus a nejsou tedy k tomuto účelu příliš vhodné.

- **Uzlové chromatické číslo** – minimální počet barev nutných k obarvení[13] všech uzlů.
- **Hranové chromatické číslo** – minimální počet barev nutných k obarvení hran.
- **Pokrytí uzlů** – nejmenší počet vrcholů tak, že každá hrana inciduje alespoň s jedním vrcholem.
- **Pokrytí vrcholů** – minimální počet hran tak, aby sousedily se všemi vrcholy.

Při izomorfizmu grafů zůstávají také cyklomatické číslo, hodnost grafu, nezávislost, klikovost, obvod[14] grafu, dominace grafu. Tato čísla se nazývají charakteristická čísla grafu, jejich definici je možné najít v každé lepší učebnici grafů.

[13] angl. colouring, provádí se tak, aby sousední uzly měly odlišnou barvu reprezentovanou číslem

[14] anglicky girth – kružnice obsažené v grafu, pokud graf neobsahuje kružnici, je obvod nekonečno

3 Problém izomorfizmu

„Jaký je izomorfizmus grafů problém?“ Jedná se o problém určení, zda jsou grafy *G = (V, E)* a *G' = (V', E')* izomorfní. Tento problém, označovaný jak GI problém, je navzdory jednoduchosti definice „těžký“. Spadá do třídy NP[15], na rozdíl od jeho generalizace – izomorfizmu podgrafů, ale nebylo dokázáno, že spadá také do třídy NP-úplných[16] problémů. Pokud by se prokázalo, že GI do NP-úplných problémů náleží, vedlo by to ke zhroucení této klasifikace a velké problémy současné kryptografie, více v [Sk90].

Nejjednodušším způsobem zjištění izomorfizmu je užitím „brutální síly,“ řešením je permutace všech vrcholů *V a V'*, složitost tohoto přístupu roste exponenciálně s počtem vrcholů, což není v současné době „rozumné“. Naštěstí ale existuje několik technik, které dokáží tuto složitost značně zlepšit, jak je popsáno níže. Na druhou ale stranu ale nebylo prokázáno, že lze GI problém zařadit do nějaké „praktické“ třídy, jako P, RP nebo BPP. Důkaz lze najít v [Sk90].

Některé speciální případy izomorfizmu nicméně lze zařadit do třídy složitosti P[17]:

- Stromy – *O(n)*.
- Rovinné grafy – *O(n)*.
- Grafy omezeného stupně obecně. V [Lu82] je ukázáno, že pro většinu „běžných“ grafů s $\Delta(G) < k$ má řešení izomorfizmu polynomiální složitost.

Dále je zřejmě možné vhodným algoritmem ukázat polynomiální složitost i pro izomorfizmus dalších grafů, například plně separabilních nebo kompletních grafů.

Je také předpokládáno dle [Wi06], že obecný izomorfizmus mezi grafy je řešitelný v polynomiálním čase pomocí metody Monte Carlo[18], respektive stroje oracle, tj. Turingova stroje napojeného na „černou skříňku“ – oracle, která umožňuje správné rozhodování v jednom kroku. My ale potřebujeme nějaké praktičtější řešení.

3.1 Efektivní zjišťování izomorfizmu

„Jak lze efektivně zjistit izomorfizmus mezi grafy?“ Bylo vyvinuto mnoho technik, kterými lze zjišťovat izomorfizmus mezi grafy, jejich základem je zejména omezení stavového prostoru permutací. Vzhledem k tomu, že se v naprosté většině algoritmů permutuje matice adjacence, což odpovídá (stromovému) prohledávání grafu do

[15] NP – (nedeterministicky polynomiální) je třída problémů, které lze řešit v polynomiálně omezeném čase na nedeterministickém počítači. Alternativně lze v polynomiálním čase ověřit jeho správnost, obecně ale ne nalézt řešení v polynomiálním čase pomocí klasického Turingova stroje.

[16] Množina, na níž jsou polynomiálně redukovatelné všechny ostatní problémy z NP. Jsou to v jistém smyslu ty nejtěžší úlohy z NP a předpokládá se, že nejsou řešitelné v polynomiálním čase.

[17] Problém řešitelný deterministickým Turingovým strojem v polynomiálním čase, složitost se zapisuje jako *O(P(n))*, *P(n)* je polynom proměnné n, reprezentující počet vrcholů grafu.

[18] Statistická metoda vyčísluje (matematické) veličiny a funkce pomocí generování náhodných čísel.

šířky, případně hloubky, jde o to, aby prohledávaný prostor byl co nejmenší, jinak řečeno aby se neprovádělo více permutací, než je nezbytně nutné – nejhůře *n!*, na což nemusí stačit ani dostupná paměť a o izomorfizmu tedy nelze rozhodnout. Tento problém lze částečně řešit pomocí následujících bodů:

- Využití invariant. Jedná se o základní krok, je nezbytně nutné určit, zda je alespoň možné, aby byly grafy izomorfní. Pokud jsou invarianty různé, není nutné prohledávat (celý) stavový prostor. Jako nejefektivnější invarianty se jeví skóre grafu a spektrum grafu algebraicky odvozené z matice adjacence. Důležité je, aby čas potřebný k jejich vytvoření byl nejhůře polynomiální, například u skóre grafu záleží na efektivní implementaci řadícího algoritmu pro dvě pole číselných hodnot – $O(n\log n)$.
- Rozklad a vytvoření matice vzdáleností. Dobré je nalézt spojité komponenty grafu, pomocí matice dosažitelnosti a především matice vzdáleností. Složitost její tvorby je $O(n^4)$, v případě, že se jedná o Dijkstrův algoritmus a $O(n^3)$, pokud o Floyd-Warshallův algoritmus, viz. [Ko84]. Tímto lze, pokud je to možné, rozložit graf na spojité podgrafy, což umožňuje omezit počet potřebných permutací. Dále je možné pomocí matice vzdáleností vytvořit rozhodovací pravidlo, které nebude zkoušet permutace, které nejsou možné, jak je to popsáno v [Kh03].
- Pokud není (další) rozklad možný, je vhodné ve vzdálenostní matici nalézt některé specifické podgrafy, jako jsou například cykly C_X nebo kliky K_X, tato metoda se nazývá slovníková.
- Využití efektivních algoritmů vyvinutých v rámci umělé inteligence, jako je back-tracking[19] [Ul76], forvard-checking[20] nebo bidirectional search[21] případně informovaných algoritmů.
- Vzhledem k povaze stávajících počítačů je velice důležitá také efektivní reprezentace grafů do binární podoby – řídká matice[22] a kódování. Je možné se pokusit vytvořit nějakou formu více vhodnou pro zjišťování izomorfizmu, například standardní nebo kanonickou formu, ale pouze tak, aby bylo využito předchozích poznatků (invariant) a složitost problému nebyla stejná jako u samotného izomorfizmu.
- Většina algoritmů je také značně paralelizovatelná, protože využívá velmi mnoho jednoduchých operací, které lze provádět současně. Nejefektivněji ale pomocí hardware speciálně vyvinutého pro tyto účely, například v [Ul76].

---

[19] metoda, která prochází všechny možné permutace, vytváří stavový prostor, který se obvykle znázorňuje jako (vyhledávací) strom – uzly jsou (vnitřní) stavy, výsledné permutace tvoří listy.

[20] na rozdíl od back-trackingu dopředně omezuje stavový prostor vyřazením nereálných stavů.

[21] vyhledávání ve dvou směrech, které se setká „na půli cesty“. Výhodou je zlomek generovaných možností, nevýhodou nutnost jejich porovnání, grafy musí být (libovolně) označené.

[22] posloupnost hran např: (1,2), (1,3), (2,3) reprezentuje graf K_3, užívá se reálných čísel.

3.2 Základní algoritmy

„Jak vypadají algoritmy pro zjišťování izomorfizmu?“ Předně musím konstatovat, že moderní algoritmy jsou kombinací mnoha technik a není možné je celé obsáhnout v této práci aniž by obsahovala stovky stran.

Naprostá většina aplikací schopných rozhodnout o izomorfizmu velkých grafů pracuje v několika fázích. Obvykle je nutné provést několik kroků „předpřípravy“. Může se jednat o konstrukci matice adjacence, dosažitelnosti a matice vzdáleností. Vhodné je nalézt spojité komponenty a vypočítat invarianty každé z nich potřebné k vyloučení izomorfizmu grafů, případně ke konstrukci rozhodovacích funkcí pro níže uvedené algoritmy. Také je možné najít kružnice a kompletní podgrafy pro porovnání „slovníku“.

V případě, že se bude jednat o zjištění izomorfizmu pomocí matice adjacence, je vhodné ji uspořádat podle vzdálenosti a to tak, že vrcholy, které spolu sousedí umístíme v matici vedle sebe (pokud to jde), aby se později prováděné permutace omezili na co možná nejvíce „lokální“ úroveň.

Přípravné algoritmy nebo jejich popis, je možné nalézt v učebnicích grafů, stejně jako v implementovaných programech. Předpříprava časové složitosti $O(n^3)$ nám sice celkovou složitost nezhorší[23], ale pokud chceme ušetřit čas, například pro (velmi) malé grafy, můžeme použít následující algoritmus.

3.2.1 Permutace vrcholů grafu

Nejjednodušší přístup hledá permutační matici mezi vrcholy vyzkoušením všech možností, proto se někdy namísto algoritmu mluví o využití „brutální síly“.

Protože tento algoritmus, za předpokladu, že zkouší každou možnost maximálně jedenkrát, například pomocí back-trackingu, může být dostatečně rychlý pro malé nepravidelné grafy s malou mírou automorfizmu a zřejmě bude dobře použitelný s nástupem kvantových počítačů, je vhodné jej doplnit o nějaký typ předzpracování.

Klasickým příkladem je využitím stupňů vrcholů. Permutujeme spolu jen ty vrcholy, které mají shodný stupeň. Už tento přístup umožňuje výrazné zrychlení algoritmu využívajícího brutální síly až na případ, kdy je málo tříd stupňů vrcholů, například když mají všechny uzly stejný stupeň. Lépe to řeší až následující algoritmus.

3.2.2 Ullmannův algoritmus

Ullmannův algoritmus, navržený v roce 1976 J. R. Ullmannem ve článku [Ul76] je jedním z nejpoužívanějších algoritmů, protože je možné jej relativně efektivně použít jak pro izomorfizmus grafů, tak i podgrafů. Využívá zjemňující proceduru pro omezení prohledávaného stavového prostoru založenou na jednoduchém předpokladu:

[23] při součtu složitostí je brán vždy jen „nejtěžší“ člen, tedy $O(n^4+n^3) = O(n^4)$

Pokud jsou v grafu G dva vrcholy sousedící, pak mohou být tyto vrcholy namapovány opět pouze na sousedící vrcholy grafu G'.

Tato podmínka je testována rekurzivně, pokud není splněna, nelze nalézt izomorfizmus podgrafů. Zjemňující procedura je volána v každém (i interním) stavovém uzlu. To sice přidává režii, ale celkově je zisk této metody obrovský. Je toho dosaženo efektivním ořezáním vyhledávacího stromu. Ullmann formuloval tuto proceduru takto:

Pokud $A(G)$ a $A(G')$ jsou adjacenční matice grafů $G=(V, E)$, respektive $G'=(V', E')$. Matice $M = [m_{ij}]$ $(1< i <|V|, 1< j <|V'|)$ je definována:

$$m_{ij} = \begin{cases} 1, \text{ pokud je možné zobrazení } v_i \in V \text{ na } v_j \in V' \\ 0 \text{ jinak} \end{cases} \quad (3.1)$$

Ullman původně při výpočtu používal výpočet, který je možné naznačit pomocí:

$$r_{xj} = (\exists y)(m_{xy} \cdot a^{G'}_{yj})$$
$$m_{ij} = m_{ij} \cdot (\forall x)(a^{G}_{ix} \vee r_{xj}) \quad (3.2)$$

kde r_{xj} jsou hodnoty matice pomocných proměnných, více v [Ul76].

Protože má tento postup složitost $O(n^4)$, což není příliš praktické, bývá nahrazován obdobnými, například v [Ic00] je uveden algoritmus se složitostí $O(n^2)$, který zkouší pouze už přiřazené vrcholy, namísto všech.

Zajímavý přístup ke tvorbě obdobné matice je možné nalézt v [Kh03], kde zjišťují nejen zda je možné zobrazení těchto vrcholů, ale počítají pro ně i předpokládanou nejkratší cestu k dalším vrcholům. Tímto je možné vytvořit ještě řidší matici možných permutací, respektive více oříznout stavový strom pro jejich vyhledávání.

3.2.3 Užití kanonické formy

Kanonická forma je typ standardní formy, na kterou lze převést graf pro zjištění izomorfizmu. Charakteristický vektor orientovaného grafu je tvaru:

$$\alpha(G) = \{a_{11}, a_{12}, \ldots a_{1n}, a_{21}, a_{22}, \ldots a_{nn}\} \quad (3.3)$$

kde a_{ij} jsou prvky adjacenční matice $A(G)$ orientovaného grafu G. Prvky jsou reprezentovány binární hodnotou. Pokud jsou lexikograficky uspořádány, pak lze zapsat kanonickou formu jako:

$$Kannon(G) = \max \alpha(G^{P_n}) \quad (3.4)$$

kde P_n je množina všech permutací grafu o $n=|V|$ vrcholech.

Problémem je, že tvorba této formy má stejnou složitost, jako řešení všech permutací grafu. Proto je nutné použít inteligentnější algoritmus, popsaný například v [Kr01], který generuje pouze nepřiřazené vrcholy a do vektoru kanonické formy přidá vždy ten s „největším“ značením. Složitost algoritmu je obdobná jako u Ullmannova algoritmu, který je sice pro některé typy grafů lepší, přesto je kanonická

forma výrazně lepší než naivní algoritmus. Nicméně přidání nějaké formy předpřípravy je pro větší grafy nutné, viz. [Mc05].

3.2.4 Další algoritmy

Na odborných konferencích, jako je ACM (http://portal.acm.org/) se objevilo mnoho různých algoritmů. Například založené na statistických vlastnostech pomocí metody Monte Carlo, viz. [Ga87].

Velice zajímavý algoritmus zveřejnili pánové J. Lopez-Presa a A. Fernandez v roce 2004 v [LP04]. Technika kombinuje jak Ullmannův princip, tak kanonickou formu. Pracuje ve třech fázích. Nejprve vytvoří sekvenci oblastí grafu, pak se saží nalézt automorfizmus jednoho grafu bez použití back-trackingu, protože jeho nalezení je zejména pro úplné grafy náročné. Nakonec generuje pro druhý graf sekvenci oblastí kompatibilních s těmi v prvním grafu pomocí back-trackingu. Pokud je toto možné, jsou grafy izomorfní, jinak ne. Podrobný postup je popsán je autory v [LP04].

3.3 Vybrané aplikace

„Jaký je současný state-of-art?“ Přestože v současné době neexistuje nástroj, který by dokázal využít všechny dostupné techniky a optimalizace, popis (byť i stručný) jeho algoritmů by zabral několik desítek stran. Proto uvádím odkazy na programy, které jsou šířené včetně jejich zdrojových kódů i dokumentace a slouží pro efektivní zjišťování izomorfizmu:

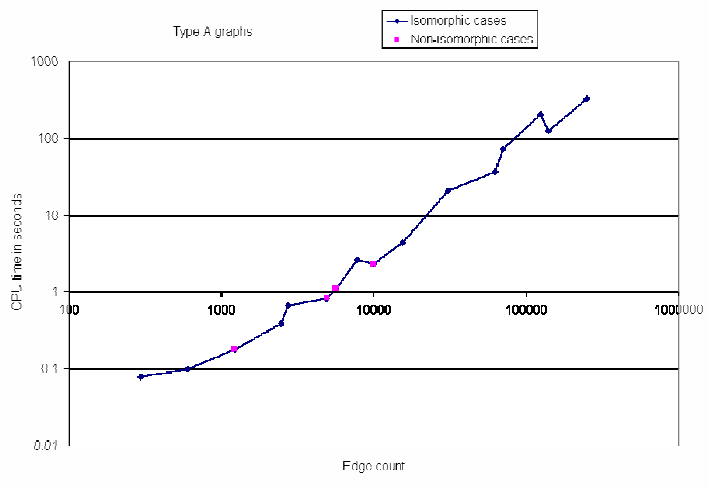
- **VFlib** (verze 2, dostupný z: http://amalfi.dis.unina.it/graph/) je v současné době považován za nejlepší program pro zjišťování izomorfizmu mezi grafy a podgrafy. Je založen na deterministickém vyhledávání podgrafů metodou back-tracking vylepšené o statistickou rozhodovací funkci realizující i forvard-checking. Obsahuje také implementaci Ullmannova algoritmu.
- **nauty** (verze 2.2, dostupný z: http://cs.anu.edu.au/~bdm/nauty/) Program, mimochodem také považován za nejlepší, je založen na kombinaci algoritmů pro rozklad grafu, užití uzlových invariant pomocí kanonické formy.
- Desítky dalších, volně šiřitelných, aplikací a knihoven nabízejících grafové algoritmy i programy pro jejich generování a zobrazení je možné nalézt například na http://sourceforge.net po zadání dotazu „graph isomorphism“ nebo „graph matching“.

3.4 Shrnutí problému izomorfizmu

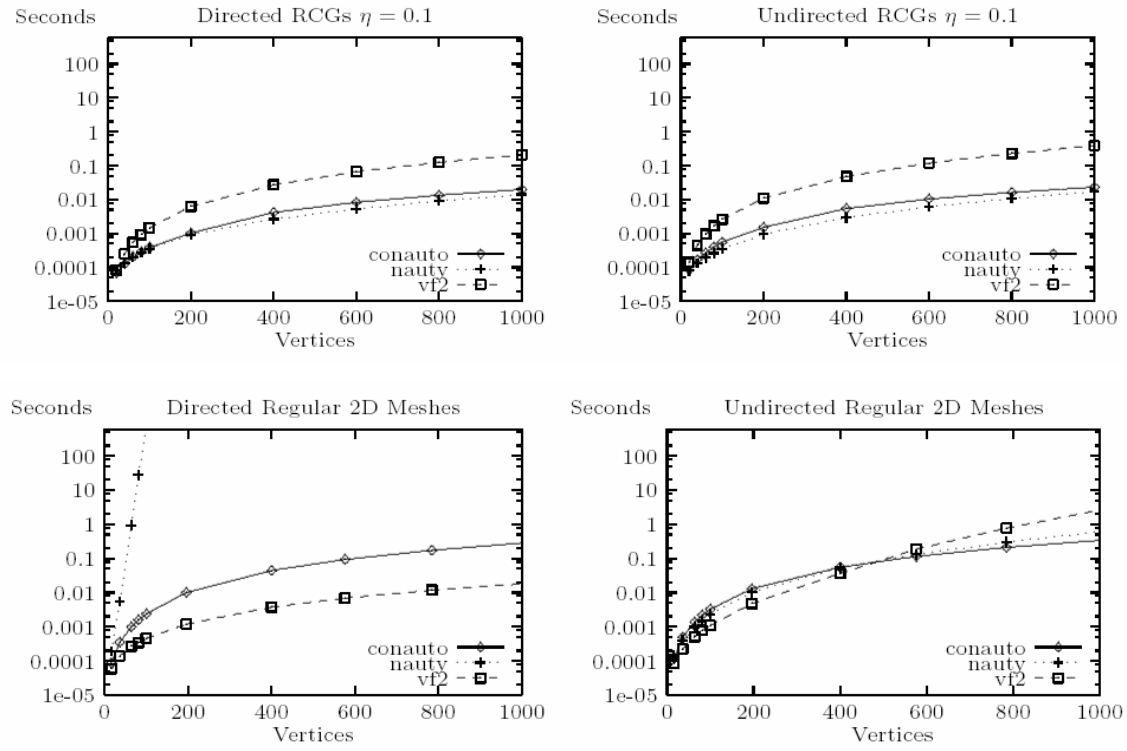
„Který algoritmus je nejrychlejší?“ Ve skutečnosti je velmi obtížné určit, která technika je nejefektivnější, velice záleží na typu a velikosti grafů. Experimenty v [FS01, LP04] ukazují, že časová složitost zjišťování izomorfizmu „naprosté většiny“ grafů je „téměř polynomiální“.

Pokud bych mohli složitost odhadnout pro „naprostou většinu“ grafů, jistě bude výsledek lepší než $O(n^5)$, jak ukazují obrázky 3.1, 3.2. Podobné je to ale i s paměťovými nároky, složitost by zde byla $O(n^2) – O(n^3)$ dle použitého algoritmu.

Tyto hodnoty umožňují běžné dnešní technice porovnávat grafy obsahující až *n = 5000* vrcholů v řádu sekund.



Obrázek 3.1: Příklad časové složitosti algoritmů (převzato z [Kh03])



Obrázek 3.2: Příklad časové složitosti algoritmů (převzato z [LP04])

V současné době (i pro komerční aplikace) je zřejmě nejběžněji využíván Ullmannův algoritmus, který je vhodný jak pro hledání izomorfizmu grafů, tak i podgrafů. Jeho implementace je jednoduchá a nabízí velké možnosti vylepšení, jak je uvedeno například v [Ic00]. Druhou možností je převést graf na jeho kanonickou formu. V každém případě je pro přesné algoritmy výhodné použít invarianty, případně rozložit graf na menší oblasti a provádět porovnávání nejprve na lokální úrovni, což je obecně asi nejefektivnějším řešením.

4 Závěr

V této práci jsem shrnul techniky, pomocí kterých je možné zjistit izomorfizmus mezi dvěma grafy se složitostí menší než *O(n!)*. I když není možné prokázat, že je potřebná složitost polynomiální [Sk90], je možné ukázat [LP04], že pro „naprostou většinu“ grafů je složitost popsaných algoritmů „přijatelná“.

K tomuto výsledku je možné dojít různými technikami, zejména využitím invariantů, rozkladu grafů, nalezením specifických podgrafů, využitím algoritmů založených na back-trackingu a obdobně. Další přínosem pro techniku izomorfizmu grafů by mohlo být využití algoritmů bidirectional search a vytvoření efektivní paralelní báze pro uložení dat včetně mezivýsledků.

Přestože problému izomorfizmu v historii věnovalo a stále věnuje množství času hodně vědců a inovátorů po celém světě, nebyl nalezen nejlepší univerzální, natož optimální algoritmus, ani není známa jeho složitost. To otevírá možnost navrhnout a implementovat nové, lepší algoritmy založené především na poznatcích, které jsem v této práci shrnul.

Zvláště přínosné by bylo implementovat takovéto algoritmy v rámci nějakého open-source databázovém systému, aby byl pomocí standardního rozhraní (např. SQL) přístupný pro libovolné aplikace zmíněné v úvodu této práce. Vzhledem k tomu, že na naší fakultě nebývají vypisována témata bakalářských a diplomových prací z této oblasti, zvážím jejich nabídku.

5 Literatura

[BM82] BONDY, J. A., MURTY U. S. R. *Graph Theory With Applications*. 5th Edition. New York : Macmillan Press, 1982. 264s. ISBN 0-444-19451-7. [cit. 2005-12-23]. Dostupný z: http://www.skidmore.edu/~adean/MC3020509WebFiles/BondyMurtyText/index.html

[Co02] CORDELLA, L. P. et. al. *An Improved Algorithm for Matching Large Graphs* [online]. Italy, 2002. [cit. 2006-01-10]. Dostupný z: http://amalfi.dis.unina.it/graph/db/papers/vf-algorithm.pdf

[Di00] DIESTEL, Reinhard. *Graduate Texts in Mathematics: Graph Theory*. Electronic version of 2nd edition. New York: Springer, 2000. 312s. Dostupný z: http://www.emis.de/monographs/Diestel/en/GraphTheoryII.pdf. ISBN 0-387-98976-5.

[FS01] FOGGIA, P., SANSONE, C., VENTO M. Performance Comparison of Five Algorithms for Graph Isomorphism, *Workshop on Graph-based Representations in Pattern Recognition* [online]. May 23-25, 2001. [cit. 2006-01-06] Dostupný z: http://amalfi.dis.unina.it/people/vento/lavori/gbr01bm.pdf.

[Ga87] GALIL, Zvi et al. An O(n3log n) deterministic and an O(n3) Las Vegs isomorphism test for trivalent graphs. *Journal of the ACM* [online]. Vol. 34, Issue 3, 1987. [cit. 2006-01-11]. Dostupný z: http://doi.acm.org/10.1145/28869.28870.

[Ic00] ICHIKAWA, S. et. al. Evaluation of Accelerator Designs for Subgraph Isomorphism Problem, *In Proceedings of 10th Int.l Conf. on Field Programmable Logic and Applications* [online]. Springer, p. 729-738. 2000. [cit. 2006-01-10]. Dostupný z:http://citeseer.ist.psu.edu/ichikawa00evaluation.html.

[Kh03] KHOJASTEPOUR, A.M., NEELAMANI, R., WILLE, T. *Project Report for Graph Isomorphism* [online]. 2003. [cit. 2006-01-08]. Dostupný z: http://www-dsp.rice.edu/~neelsh/comp482/

[Ko84] KOLÁŘ, Josef. *Grafy*. Praha : ČVUT, 1984. 231s.

[Kr01] KŘENA, Bohuslav: The Graph Isomorphism Problem *In: Proceedings of 7th Conference Student FEI 2001* [online], Brno, 2001. Dostupný z: http://www.fit.vutbr.cz/research/view_pub.php?id=6047.

[Lo00] LOCKE, Stephen C. *Graph Theory* [online] 2000. [cit. 2005-12-23]. Dostupný z: http://www.math.fau.edu/locke/GRAPHTHE.HTM

[LP04] LOPEZ-PRESA, J. L., FERNANDEZ, A. *Graph Isomorphism Testing Without Full Automorphism Group Computation* [online]. Spain, 2004. [cit. 2006-01-10]. Dostupný z: http://gsyc.escet.urjc.es/publicaciones/tr/isomorfismo-04.pdf.

[Lu82] LUKS, E. M. Isomorphism of Graphs of Bounded Valence Can Be Tested in Polynomial Time. *In Journal of Computer System Science.* s. 42-65, 1982.

[Mc05] McKAY, Brendan D. *nauty User's Guide* [online]. Australia, 2005. [cit. 2006-01-10]. Dostupný z: http://cs.anu.edu.au/~bdm/nauty/nug.pdf.

[SA98] University of St Andrews, Scotland, School of Mathematics and Statistics. *Leonhard Euler* [online] 1998. [cit. 2005-12-23]. Dostupný z: http://www-gap.dcs.st-and.ac.uk/~history/Mathematicians/Euler.html.

[Sk90] SKIENA, S. Graph Isomorphism. *In Implementing Discrete Mathematics: Combinatorics and Graph Theory with Mathematica*. Reading, MA: Addison-Wesley, pp. 181-187, 1990.

[Sv01] SVRŠEK, Jiří. Matematikové v historii: Leonhard Euler. *Vědecká sekce deníku neviditelný pes* [online] 2001. [cit. 2005-12-23]. Dostupný z: http://archiv.neviditelnypes.zpravy.cz/veda/clanky/13637_0_0_0.html.

[Ul76] ULLMAN, Jeffrey D. An Algorithm for Subgraph Isomorphism. *Journal of the ACM* [online]. 23(1): 31–42. 1976. [cit. 2006-01-09] Dostupný z: http://portal.acm.org/citation.cfm?id=321925

[Wi06] *Graph theory* [online]. 2006 [cit. 2006-01-10]. Dostupný z: http://en.wikipedia.org/wiki/Graph_theory.

[WM03] WINCZER, Michal. *Teória grafov* [online]. 2003 [cit. 2006-01-03]. Dostupný z: http://user.edi.fmph.uniba.sk/winczer/diskretna/pred8z03.pdf.