Kapitola 2

Efektívne učenie I

2.1 Pohľad na teóriu zložitosti

Predmet *Výpočtová zložitosť* študuje vzťahy medzi veľkosťou vstupu do algoritmu a časom, ktorý algoritmus spotrebuje, aby vypočítal výstup pre vstup tejto veľkosti teda zaoberá sa efektívnosťou (účinnosťou) algoritmov.

Veľkosť vstupu do algoritmu môže byť meraná rôznymi spôsobmi.

Ak sa zaoberame booleovskými funkciami - počet bitov, ktoré sú na vstupe, ale ak uvažujeme reálne čísla, tak môžu vzniknúť problémy ?

Čas behu algoritmu je závislý od toho, ako rýchlo môže byť výpočet vykonaný. Pretože toto budeme robiť nezávisle od zariadenia, budeme počítať počet potrebných operácií (obvykle pre najhorší prípad!). Budeme sa zaujímať len o závislosti, preto budeme používať nasledujúcu definíciu:

Nech A je algoritmus, ktorý akceptuje vstupy rôznej veľkosti s. Hovoríme, že čas behu A je $O(f(s))$, ak pre ľubovoľný vstup veľkosti s, počet operácií požadovaných na získanie výstupu algoritmu A je najviac $K * f(s)$, kde K je nejaká konštanta, $K > 0$.

2.1.1 Splniteľnosť booleovskej formuly (satisfiability)

Inštancia: Booleovská formula Φ o n premenných
Question:Existuje pozitívny príklad z $\langle\Phi\rangle$?

NP-ťažký; redukovateľnosť.
NP-úplný, ak $\in$ NP a každý problém z NP je naň redukovateľný;

Budeme aplikovať idey teórie výpočtovej zložitosti:

Predpokladajme, že Π je problém, o ktorý sa zaujímame, Π_0 je problém o ktorom je známe, že je NP-úplný. Predpokladajme, že môžme demonštrovať, že ak existuje polynomiálny algoritmus pre Π, potom existuje aj pre Π_0. V tomto prípade náš problém sa nazýva NP-ťažký problém.
V prípade, že platí $P \neq NP$, potom dôkaz, že M je NP-ťažký znamená, že preň neexistuje polynomiálny algoritmus.

2.1.2 Cas behu učiacich algoritmov

Väčšina učiacich algoritmov prediskutovaných v predchádzajúcom texte sa zaoberá booleovskými konceptami. V týchto prípadoch príkladový priestor je $\{0,1\}^n$ pre nejaké pevné n a hypotézový priestor je množina funkcií definovaných na príkladovom priestore. Pre každý z týchto algoritmov parameter n je ľubovoľný v zmysle, že algoritmus je definovaný pre ľubovoľné n a navyše operuje v podstate tým istým spôsobom pre každú hodnotu n. Napríklad, štandartný učiaci algoritmus pre priestor M_n monočlenov je definovan'y celkom všeobecne, hoci potrebujeme špeciálny "stroj" na jeho implementáciu pre danú hodnotu n. Chceme kvantifikovať chovanie sa učiacich algoritmov vzhľadom na n, a je obvykle používať nasledujúce definície.

Definícia 2.1.1 *Hovoríme, že zjednotenie hypotézových priestorov $H = \bigcup H_n$ je odstupňované (graded) príkladmi veľkosti n, ak H_n označuje hypotézový priestor definovaný na n príkladoch z X^n.*

Definícia 2.1.2 *Učiaci algoritmus pre $H = \bigcup H_n$ je funkcia L z množiny tréningových príkladov pre hypotézy v H do priestoru H taký, že ak $\overline{s}$ je tréningová vzorka pre $h \in H_n$, tak z toho vyplýva $L(\overline{s})) \in H_n$. Teda L podporuje stupňovanie (grading) priestoru H.*

Uvažujme učiaci algoritmus L pre bool. hypotézový priestor $H = \bigcup H_n$, odstupňovaný veľkosťou príkladov. Vstup do L je tréningová vzorka, ktorá pozostáva z m n-bitových vektorov spolu s m 1-bitovými označeniami. Celkový počet bitov na vstupe je $m(n+1)$ a bolo by možné použiť toto jediné číslo ako mieru veľkosti vstupu. Avšak je výhodné sledovať aj m aj n oddelene a použijeme označenie $R_L(m,n)$ **na označenie najhoršieho času behu L na tréningovej vzorke m n-bitových vektorov.**

Príklad 1: Nech L je učiaci algoritmus pre monočleny popísaný vyššie. Hypotézový priestor je zjednotenie $\bigcup M_n$. Hlavný krok algoritmu vyžaduje kontrolu každého bitu u každého pozitívneho príkladu a možno vymazanie niektorých literálov. V najhoršom prípade, každý príklad v tréningovej vzorke môže byť pozitívny príklad, a tak by sme mali ošetriť tento krok m-krát, každý krok obsahuje kontrolu n bitov. Iné časti výpočtu vyžadujú porovnateľne toľko operácií, takže môžme hovoriť, že čas behu $R_L(m,n)$ je v tomto prípade $O(m * n)$.

Príklad 2: Vyššie sme popísali učiaci algoritmus pre priestor $D_{n,k}$ disjunkcií malých mnohočlenov. Ako obvykle, považujeme k za pevné a n za premennú. Každý krok algoritmu obsahuje kontrolu, či jeden z m príkladov v tréningovej vzorke je pozitívny alebo negatívny a ak je negatívny, vyhodnotenie niektorých monočlenov v $M_{n,k}$. Na začiatku zoznam relevantných monočlenov má dĺžku okolo $(2n)^k$ a v každom stave môže byť niektorý z nich vymazaný. Pretože k je pevné, 2^k je konštanta, a teda čas behu je $O(m, n^k)$.

Oba vyššie prediskutované algoritmy sú bezpamäťové on-line algoritmy, a to znamená, že výpočet času behu je veľmi jednoduchý. V takom algoritme L vyžaduje najviac $S_L(n)$ operácií na spracovanie jediného n-bitového príkladu, potom jeho čas behu je $R_L(m,n) \leq mS_L(n)$. Pre algoritmy, ktoré nie sú tohto typu, výpočet času behu môže byť zložitejší.

Príklad: Analyzujme algoritmus na učenie v rozhodovacích zoznamoch $DL(K_n)$. Toto zrejme nie je bezpamäťový on-line algoritmus, pretože v každom kroku je nutné kontrolovať všetky zostavajúce príklady so zoznamom dvojíc (g,c), kde $g \in K_n$ a $c \in \{0,1\}$. Ak je na začiatku m príkladov v trénujúcej vzorke, bude tu $2|K_n|m$ kontrôl v 1. kroku v najhoršom prípade. Aspoň 1 príklad bude vymazaný, takže ďalší krok vyžaduje $2|K_n|(m-1)$ kontrôl. Opakovaním tých istých argumentov dostávame celkový počet kontrôl najviac

$$2(m + m - 1 + \ldots + 3 + 2 + 1)|K_n| = O(m^2|K_n|).$$

Ak K_n je $M_{n,k}$, pre ktorého kardinalitu máme ohraničenie $(2n)^k$, čas behu je $O(m^2 n^k)$.

2.2 Prístup k efektívnosti PAC učenia

Všeobecný prístup k dokazovaniu vlastnosti PAC pre nejaký učiaci algoritmus bol uvedený vyššie. V kontexte odstupňovaného hypotézového priestoru $H = \bigcup H_n$ bool. funkcií môžme popísať procedúru schématicky nasledovne:

$$H_n \text{ konečný} \quad \Rightarrow \quad H_n \text{ potencionálne naučiteľný}$$

H_n potencionálne naučiteľný $\wedge$ L je konzistentný pre $H_n \Rightarrow L$ PAC učí H_n

S ohľadom na účinnosť vzniká prirodzená otázka: pre dané požadované úrovne presnosti a dôvery, ktorá (aká) podmienka zaručí, že čas behu, v ktorom L PAC učí H_n, je polynomiálny v n?

V tomto bode nám pomôže zavedenie ďalšej terminológie na popis podobných pojmov. Predpokladajme, že sú dané reálne čísla $0 < \delta, \epsilon < 1$ a nech L je učiaci algoritmus pre konceptový priestor C a hypotézový priestor H. (Predpoklad $C = H$ tu nie je požadovaný.) Hovoríme, že zložitosť vzorky pre L na podmnožine $T \subseteq C$ je najmenšia hodnota $m_L(T, \delta, \epsilon)$ taká, že pre všetky cieľové koncepty $t \in T$ a všetky pravdepodobnostné rozloženia μ

$$\mu^m\{s \in S(m,t) \quad | \quad err(L(s)) < \epsilon\} > 1 - \delta$$

pre všetky $m \geq m_L(T, \delta, \epsilon)$; inak povedané vzorka dĺžky $m_L(T, \delta, \epsilon)$ postačuje k záruke, že výstupná hypotéza je PAC pre dané ϵ, δ. V praxi sa skôr zaoberáme obvyklou dolnou hranicou $m_0 \geq m_L$, než m_L samotným; teda $m_0(T, \delta, \epsilon)$ bude označovať ľub. hodnotu postačujúcu k záruke, že PAC (ako bolo uvedené vyššie) platí pre všetky $m \geq m_0$.

Úplné zovšeobecnenie definície bude použiteľné na prípady v ďalších kapitolách, ale vo väčšine aplikácií budeme uvažovať $T' = C = H$. Napríklad, použitím tejto terminológie veta ukazuje, že pre konzistentný učiaci algoritmus na konečnom priestore H, dolná hranica pre zložitosť vzorky $m_L(H, \delta, \epsilon)$ je $m_0 H, \delta, \epsilon) = \lceil \frac{1}{\epsilon} ln \frac{|H|}{\delta} \rceil$.

Zložitosť vzorky poskytuje prepojenie medzi časom behu $R_L(m, n)$ učiaceho algoritmu (tj. počet operácií potrebných produkovať svoj výstup na vzorke dĺžky m, ak príklady sú dĺžky n) a jeho časom behu ako PAC učiaceho algoritmu (tj. počet operácií potrebných na vytvorenie výstupu, ktorý je PAC s danými parametrami). Pretože vzorka dĺžky $m_0(H_n, \delta, \epsilon)$ postačuje pre vlastnosť pac, počet vyžadovaných operácií je najviac $R_L(m_0(H_n, \delta, \epsilon), n)$. V prípade konzistentného algoritmu toto poskytuje odpoveď na otázku položenú vyššie.

Veta 1 *Predpokladajme, že L je konzistentý učiaci algoritmus pre hypotézový priestor* $H = \bigcup H_n$. *Ak*

- $R_L(m, n)$ *je polynóm v* m *a* n,
- $ln|H_n|$ *je polynóm v* n,

potom pre dané hodnoty parametrov pre presnosť ϵ *a pre dôveru* δ *čas behu, počas ktorého L bude produkovať pravdepodobnostne aprixomavanú správnu hypotézu je polynomiálny v n.*

Dôkaz: Pretože L je konzistentný, horná hranica pre zložitosť vzorky algoritmu L na H_n je

$$\lceil \frac{1}{\epsilon} ln \frac{|H_n|}{\delta} \rceil$$

Teda je nutné potvrdiť, že keď podmienky platia, tak výraz $R_L(\lceil \frac{1}{\epsilon} ln \frac{|H_n|}{\delta} \rceil, n)$ sa dá upraviť na polynóm vzhľadom na n. □

Tento výsledok vrhá svetlo na účinnosť učiacich algoritmov prediskutovaných skôr. Napríklad, priestor monočlenov má mohutnosť $|M_n| = 3^n$ a tak $\ln |M_n| = n \lg 3$. Štandartný učiaci algoritmus pre monočleny je konzistentný a má čas behu $O(mn)$. Z toho vyplýva, že algoritmus PAC učí M_n v polynomiálnom čase vzhľadom na n - špecificky, čas behu je $O(mn^k)$ a teda vieme, že $|D_{n,k}|$ je najviac $2^{(2n)^k}$. Z toho vyplýva, že $\ln |D_{n,k}|$ je ohraničený konštantným násobkom n^k a teda implikuje, že algoritmus PAC učí $D_{n,k}$ v čase $O(n^{2k})$.

Predchádzajúca veta nám však neumožňuje načrtnúť žiadny záver o účinnosti všeobecnejších učiacich algoritmov. Napríklad, algoritmus pre učenie očíslovaním, vyžaduje zakaždým m označených príkladov v tréningovej vzorke na kontrolu (porovnanie) s hypotézami v H_n. Teda je to konzistentný algoritmus, ktorého čas behu $R_L(m, n)$ je $O(m|H_n|)$. V tomto prípade prvá podmienka predchádzajúcej vety vyžaduje, že $|H_n|$ samotné (skôr než jeho logaritmus) rastie polynomiálne. Toto je veľmi obmedzujúca podmienka, pretože aj celkom ohraničené hypotézové priestory ako M_n a $D_{n,k}$ majú kardinalitu, ktorá rastie exponenciálne.

V dôsledku toho, hoci algoritmus očíslovaním môže byť použitý pre ľub. konečný priestor, existuje mnoho prípadov kde minulá veta nemôže byť aplikovaná k tomu, aby ukázala, že alg. bude produkovať pravdepodobnostne aproximované korektné hypotézy v polynomiálnom čase.

Aplikujme vetu na učiaci algoritmus pre $DL(K_n)$, pre ktorý čas behu je $O(m^2|K_n|)$. Prvá podmienka vyžaduje, že mohutnosť základného priestoru K_n je polynomiálna funkcia v n. V skutočnosti to je všetko, čo je potrebné, pretože druhá podmienka vyplýva z toho automaticky; tj. $ln|DL(K_n)|$ je polynóm v $|K_n|$. Aby sme to overili odhadneme, že $ln(N!) \leq NlnN$, a tak máme

$$\ln |DL(K_n)| \leq |K_n|(\ln |K_n| + \ln 3)$$

a zrejme toto je ohraničené polynómom v n vždy, keď je aj $|K_n|$ - napríklad $K_n = M_{n,k}$.

2.3 Problém konzistencie tréningovej vzorky

Budeme študovať odraz teórie o NP-ťažkých problémoch v tejto oblasti. Nech $H = \bigcup H_n$ je hypotézový priestor bool. funkcií odstupňovaný príkladmi veľkosti n. Problém konzistencie pre H môže byť stanovený nasledovne:

H - KONZISTENCIA:

- Inštancia tréningovej vzorky s vyjadrenej n-bitovými označenými vektormi.

- Otázka: Existuje hypotéza v H_n konzistentná s $\overline{s}$?

Ukážeme, že v niektorých netriviálnych prípadoch je tento problém NP-ťažký. Na to, aby sme vyjadrili praktický dosah tohto výsledku, potrebujeme urobiť niekoľko všeobecných komentárov.

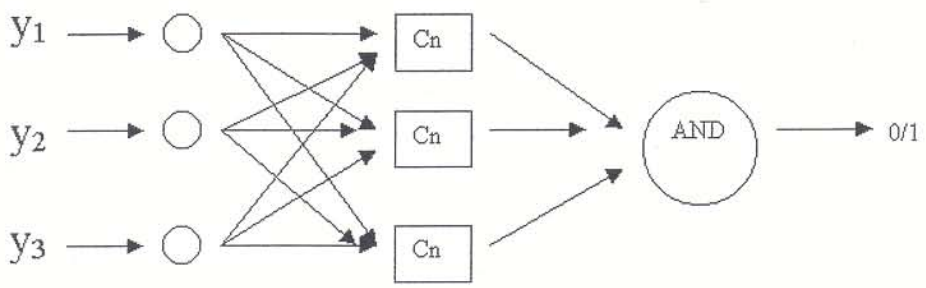
Prvý komentár, ak uvažujeme len tie inštancie problému, v ktorých dĺžka vzorky $\overline{s}$ je ohraničená nejakým fixným polynómom v n, potom máme ohraničený tvar problému konzistencie. Sú také ohraničené tvary, ktoré budeme sledovať v tejto predáške a uvidíme, že niektoré z týchto problémov sú NP-ťažké. Upozorňujeme, že ak ohraničený tvar H-kozistencie je NP-ťažký, potom aj H-kozistencia samotná je NP-ťažká.

Ďalší komentár, v praxi si skôr prajeme nájsť konzistentnú hypotézu, než len vedieť o tom, či existuje. Inak povedané, máme riešiť problém "hľadania" skôr než problém "existencie". Ale tieto problémy sú v priamom vzťahu. Predpokladajme, ako vyššie, že uvažujeme len tie $\overline{s}$ s dĺžkou ohraničenou nejakým polynómom. Potom, ak môžme nájsť konzistentnú hypotézu v polynomiálnom čase v n, tak môžeme odpovedať na existenčnú otázku pomocou nasledujúcej procedúry:

Spustiť hľadací algoritmus na čas (polynomiálny v n), v ktorom je zaručené nájdenie hypotézy, ak existuje. Potom skontrolovať výstupnú hypotézu explicitne s príkladmi zo vzorky s, čo nám povie, či je konzistentná alebo nie. Táto kontrola môže byť tiež urobená v polynomiálnom čase v n. Teda, ak ukážeme, že ohraničený tvar existenčného problému je NP-ťažký, to znamená, že neex. polynomiálny algoritmus pre odpovedajúci vyhľadávací problém (pokiaľ neplatí P=NP).

2.3.1 Výsledok o zložitosti

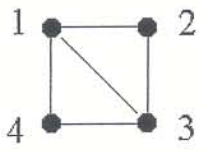
Pitt a Valiant (1988) boli prví, ktorí ukázali príklad hypotézového priestoru H, pre ktorý ohraničený tvar problému konzistencie je NP-ťažký. Nasleduje o niečo zjednodušený popis ich metódy. Nech C_n je priestor klauzúl - množina bool. fukcií n premenných, ktoré môžu byť reprezentované v tvare klauzúl; tj. formuly tvaru $u_2 \vee \overline{u_3} \vee u_6$ čo sú disjunkcie literálov. Nech C_n^k je priestor bool. funkcií, ktoré môžu byť reprezentované ako konjukcie k klauzúl. Môžme si predstaviť C_n ako hypotézový priestor stroja duálneho k stroju pre monočleny a C_n^k ako hypotézový priestor stroja vytvoreného spojením k $C_n - strojov$ paralelne a prechodom ich výstupov cez AND jednotku s n vstupmi.



Ukážeme, že pre pevné $k \geq 3$, problém konzistencie pre $C^k = \bigcup C_n^k$ je NP-ťažký. Teda nie je pravda, že ex. polynomiálny učiaci algoritmus pre C_n^k, ktorý produkuje konzistentnú hypotézu. Dôkaz spočíva v prevedení problému farbenia grafu na tento problém s n vrcholmi pomocou k farieb, čo je NP-úplný problém pre $k \geq 3$ (Gray, Johnson 1979).

Problém farbenia: G=(V,E), k-farbenie je funkcia $\chi : V \to \{1, 2, \ldots, k\}$ s vlast. $\langle v_i, v_j \rangle \in E$ potom $\chi(v_i) \neq \chi(v_j)$. Predp., že máme G=(V,E), V=1,2, ... n. Skonštruujeme trénujúcu vzorku s(G) nasledovne: Pre každý vrchol $i \in V$ určíme záporný príklad vektor v_i, ktorý má 1 v pozícii i-tej súradnice a 0 inde. Pre každú hranu $\langle i, j \rangle \in E$ vezmeme ako pozitívny príklad vektor $v_i + v_j$.

Príklad:



v_i	
1000	0
0100	0
0010	0
0001	0
1100	1
1010	1
1001	1
0110	1
0011	1

Tvrdenie: Ex. funkcia C_n^k, ktorá je konzistentná so vzorkou $s(G) \iff$ graf G je k -zafarbiteľný.

Dôkaz:

$\Rightarrow$

Predpokladajme, že $h \in C_n^k$ a je konzist. s trén. vzorkou. Podľa def. h je konjunkcia $h = h_1 \wedge h_2 \wedge \ldots h_k$ klauzúl. Pre každý vrchol $i \in G$, $h(v_i) = 0$ a teda musí ex. aspoň 1 klauzula h_f, pre ktorú $h_f(v_i) = 0$. Funkcia $\chi : V \to \{1, 2, \ldots k\}$ taká, že:

$$\chi(i) = min\{f \quad | \quad h_f(v_i) = 0\}$$

Zostáva ukázať, že χ je farbenie grafu (G); inak povedané, ak i a j sú dva vrcholy, pre ktoré $\chi(i) = \chi(j)$, potom $\langle i, j \rangle \notin E$. Predp., že $\chi(i) = \chi(j) = f$ a teda $h_f(v_i) = h_f(v_j) = 0$. Pretože h_f je klauzula, každý literál, ktorý sa v nej vyskytuje musí byť 0 na v_i a na v_j. Teraz v_i má 1 len v i-tej pozícii a tak $h_f(v_i) = 0$ implikuje, že len jeden negovaný literál, ktorý sa môže vyskytnúť v h_f je $\overline{u_i}$. Pretože to isté platí pre $\overline{u_j}$, dostávame, že h_f obsahuje len niektoré literály u_z, pre ktoré $z \neq i, j$. Teda $h_f(v_i + v_j) = 0$ a $h(v_i + v_j) = 0$. Ak by $\langle i, j \rangle$ bola hranou v G, potom by platilo $h(v_i + v_j) = 1$, pretože sme predpokladali, že h je konzistentná s s(G). Teda $\langle i, j \rangle$ nie je hranou v G a χ je farbenie.

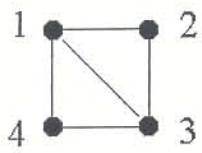
$\Leftarrow$

Predpokladajme, že je dané farbenie $\chi : V \to \{1, 2, \ldots, k\}$. Pre $1 \leq f \leq k$ definujme h_f ako klauzulu $\langle \vee u_{i\chi(i) \neq f} \rangle$ a definujme $h = h_1 \wedge h_2 \wedge \ldots \wedge h_k$. Tvrdíme, že h je konzistentná s $s(G)$.

Najprv, predpokladajme, že pre vrchol i $\chi(i) = g$. Klauzula h_g je definovaná tak, že obsahuje len tie (nie negované) literály, ktoré zodpovedajú vrcholom nezafarbeným farbou g, a teda u_i sa nenachádza v h_g. Teda $h_g(v_i) = 0$ a $h(v_i) = 0$.

Ďalej, nech ij je hrana v G. Pre každú farbu f aspoň jedno v_i alebo v_j nemá farbu f; označme vhodný výber $i(f)$. Potom h_f obsahuje literál $u_{i(f)}$, ktorý je 1 na $v_i + v_j$. Teda klauzula h_f je 1 na $v_i + v_j$ a $h(v_i + v_j) = 1$, ako sme požadovali. □

Príklad:



má 3-farbenia χ

$\chi(1)=1$
$\chi(2)=2$
$\chi(3)=3$
$\chi(4)=2$

Teda je funkcia h v C_4^3 je konzistentná s odpovedajúcou tréningovou vzorkou

$$h = h_1 \wedge h_2 \wedge h_3 = \langle (u_2 \vee u_3 \vee u_4) \wedge (u_1 \vee u_3) \wedge (u_1 \vee u_2 \vee u_4) \rangle$$

Tento graf nemôže byť zafarbený 2 farbami a teda môžme povedať, že neex. C_4^2, ktorá je konzistentná s trén. vzorkou.

Predchádzajúce tvrdenie vyjadruje súvislosť medzi k-farbením grafov a problémom C^k -konzistencie. Ak zvolíme kladné celé číslo k pevne, potom môžeme tieto 2 problémy vyjadriť formálnejšie:

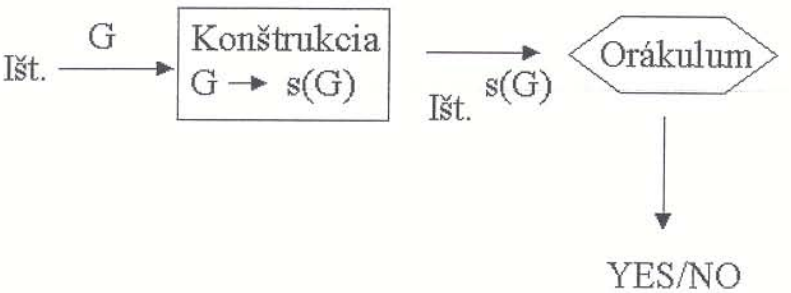
k-FARBENIE GRAFU

Je daný graf G s n vrcholmi. Existuje k - farbenie grafu G?

C^k - KONZISTENCIA

Je daná trénujúca vzorka s s označenými n-bitovými vektormi. Ex. funkcia v C_n^k, ktorá je konzistentná s s?

Dôkaz, že C^k - konzistencia je NP-ťažký problém je uvedený na obr.



Najprv k danej inštancií G skonštruujeme (v polynomiálnom čase) inštanciu s(G) pre C^k konzistenciu. Poznamenajme, že počet hrán v grafe s n vrcholmi je najviac $n(n-1)/2$, a tak počet príkladov v s(G) je najviac $n + n(n-1)/2$, čo je $O(n^2)$. Teraz predpokladajme, že existuje algoritmus (môžeme si myslieť, vze je to orákulum), ktorý môže dávať odpovede na otázky C^k konzistencie. Ak orákulum operuje v polynomiálnom čase vzhľadom na n, potom by mohlo odpovedať na pôvodnú otázku tiež v polynomiálnom čase. Avšak pôvodný problém je NP-ťažký. Teda už ohraničený tvar C^k - konzistencie, v ktorom je $m \sim O(n^2)$ je NP-ťažký. Predchádzajúci dôkaz o tom, že C^k problém konzistencie je NP-ťažký, platí len pre $k \geq 3$, pretože problém k-farbenia grafu je NP-ťažký len pre $k \geq 3$. Keď $k = 1$, máme $C_1^n = C_n$ a ex. polynomiálny algoritmus pre C_n duálny k známemu algoritmu pre monočleny. Keď $k = 2$, môžme ukázať inými metódami, že problém konzistencie je NP-ťažký.

2.4 Ďalšie poznámky

Prediskutovali sme učiace algoritmy pre odstupňované priestory $H = \bigcup H_n$, kde hypotézový priestor a konceptový priestor koincidujú. Je ľahké definovať učiaci algoritmus pre odstupňovaný konceptový priestor $\bigcup C_n$ pomocou (možno rôznych) stupňového hypotézového priestoru $H = \bigcup H_n$; taký algoritmus vezme vstupné vzorky pre hypotézy v $C = \bigcup C_n$ a výstupné hypotézy v $H = \bigcup H_n$ s vlastnosťou, že ak s je trénujúca vzorka pre hypotézu v C_n, potom $L(s) \in H_n$. Hausler, Littlestone a Warmuth (1988) zaviedli pojem účinná predikcia postupne študovaný Pittom a Warmuthom (1988,1990) a Haussslerom (1988). Zhruba povedané, odstupňovaný konceptový priestor $C = \bigcup C_n$ je účinne predikovateľný ak existuje nejaký stupňový hypotézový priestor $H = \bigcup H_n$ taký, že existuje pac učiaci algoritmus pre $(\bigcup C_n, \bigcup H_n)$ v polynomiálnom čase (n). (Aby sme boli presnejší, od H požadujeme málo "polynomiálne vyhodnotiteľnú" reprezentáciu. Inak, povedané, ak učiaci algoritmus je prezentovaný tréningovou vzorkou pre cieľ v C_n, potom výstupom L je reprezentácia ω hypotézy $h_\omega \in H_n$ taká, že možno určiť v polynomiálnom čase či

je daný príklad pozitívny pre h_ω. (Nebudeme o tom ďalej hovoriť, budeme v ďalšom predpokladať, že všetky reprezentácie v tejto knihe majú túto vlastnosť.

2.5 Úlohy:

1. Prečo je obyčajne vhodné uvažovať veľkosť vstupu v tvare $\lg n$ namiesto n?