Výpočtové učenie

8. marca 2006

Kapitola 1

Konzistentné algoritmy a naučiteľnosť

1.1 Potenciálna naučiteľnosť

Učenie v zmysle PAC je vlastnosť algoritmu. Ak je algoritmus daný, môžeme sa pokúšať dokázať, že je PAC, ale môžeme tiež uvažovať vo všeobecnejšej rovine.

V tejto časti budeme popisovať **vlastnosť hypotézového priestoru H**, ktorá zaručí, že konzistentný algoritmus pre učenie H podľa H je **PAC** a ukážeme, že mnoho priestorov má túto vlastnosť.

Definícia 1.1.1 *Nech H je hypotézový priestor funkcií definovaných na príkladovom priestore X. Učiaci algoritmus L pre H je* ***konzistentný****, ak pre ľubovoľnú trénujúcu vzorku $\overline{s}$ a cieľový koncept $t \in H$, výstupná hypotéza $h = L(s) \in H$ súhlasí s t na príkladoch v $\overline{s}$, t. j. $h(x_i) = t(x_i) \quad (1 \leq i \leq m)$.*
Pre dané $\overline{s} \in S(m,t)$ je obvyklé označenie $H[\overline{s}] = \{h \in H| \quad h(x_i) = t(x_i), \quad 1 \leq i \leq m\}, H[\overline{s}]$ je množina všetkých hypotéz konzistentných s $\overline{s}$.

Teda L je konzistentný vtedy a len vtedy, keď $L(\overline{s}) \in H[\overline{s}]$ pre všetky trénujúce vzorky $\overline{s}$.

Z toho vyplýva, že zabezpečiť, aby konzistentný učiaci algoritmus bol PAC, postačí zadať podmienky na množiny $H[\overline{s}]$.

Ako predtým predpokladajme, že je dané pravdepodobnostné rozloženie μ na X. Na moment zafixujme cieľový koncept $t \in H$.
K danému $\epsilon \in (0,1)$ položíme $B_\epsilon = \{h \in H| \quad er_\mu(h) \geq \epsilon\}$ čo môže byť popísané ako množina ϵ-zlých hypotéz pre t. Konzistentný algoritmus pre H dáva výstup, ktorý je v $H[\overline{s}]$ a PAC vlastnosť vyžaduje, aby taký výstup, ktorý je nepravdepodobný, bol ϵ-zlý; inak povedané, zdôrazníme, že je nepravdepodobné, že zlá hypotéza je korektná na vzorke. Toto vedie k nasledujúcej definícii:

Definícia 1.1.2 *Hovoríme, že hypotézový priestor H je* **potenciálne naučiteľný***, ak k daným reálnym číslam $\delta, \epsilon, 0 < \delta, \epsilon < 1$ existuje kladné celé číslo $m_0 = m_0(\delta, \epsilon)$ také, že pre všetky $m \geq m_0$, platí*

$$\mu^m\{s \in S(m,t)| \quad H[s] \cap B_\epsilon = \emptyset\} > 1 - \delta$$

pre ľubovoľné pravdepodobnostné rozloženie μ na X a ľub. $t \in H$.

Veta 1 *Ak H je potenciálne naučiteľný a L je konzistentný učiaci algoritmus pre H, potom L je PAC.*

Dôkaz: L je konzistentný, teda $L(\overline{s}) \in H[\overline{s}]$.

$$H[\overline{s}] = \{h \in H| \quad h(x_i) = t(x_i), \quad 1 \leq i \leq m\}$$

$$B_\epsilon = \{h \in H| \quad err_\mu(h) \geq \epsilon\}$$

$L(\overline{s}) \in H[\overline{s}]$ a zároveň $H(s) \cap B_\epsilon = \emptyset$, teda chyba $L(s)$ je menšia ako ϵ.

$$\forall_\delta \ \forall_\epsilon \ \exists_{m_0} \ \forall_{m>m_0} \ \forall_\mu \ \forall_{t \in H} \ \mu^m\{\overline{s} \in S(m,t)| \quad H[\overline{s}] \cap B_\epsilon = \emptyset\} > 1 - \delta$$

$$\forall_\delta \ \forall_\epsilon \ \exists_{m_0} \ \forall_{m>m_0} \ \forall_\mu \ \forall_{t \in H} \ \mu^m\{\overline{s} \in S(m,t)| \quad err_\mu(L(\overline{s})) < \epsilon\} > 1 - \delta$$

$$err_\mu(L(\overline{s})) = \mu\{x \in X| \quad t(x) \neq L(\overline{s})\}$$

□

1.2 Konečný prípad

Definícia potenciálnej naučiteľnosti je celkom zložitá a môže byť dokázané, že viac-menej je to popis PAC učenia. Našou úlohou je teraz potvrdiť definíciu tým, že ukážeme jej významné aplikácie.

Veta 2 *Ľubovoľný konečný hypotézový priestor H je potenciálne naučiteľný.*

Dôkaz: Predpokladajme, že H je konečný hypotézový priestor a δ, ϵ, t a μ sú dané. Dokážeme, že pravdepodobnosť udalosti $H[\overline{s}] \cap B_\epsilon \neq \emptyset$ (komplement udalosti v definícii) môže byť zvolená menšia než δ vybratím dostatočne veľkej dĺžky vzorky $\overline{s}$.

Pretože B_ϵ je definovaná tak, že obsahuje ϵ-zlé hypotézy, z toho vyplýva, že pre ľub. $h \in B_\epsilon = \{h \in H| \quad \mu\{x \in X, h(x) \neq t(x)\} \geq \epsilon\}$ platí

$$\mu\{x \in X| \quad h(x) = t(x)\} = 1 - err_\mu(h) \leq 1 - \epsilon$$

Teda pre celú vzorku dĺžky m máme

$$\mu^m\{s| \quad h(x_i) = t(x_i), \ 1 \leq i \leq m\} \leq (1-\epsilon)^m$$

Toto je pravdepodobnosť, že hypotéza h je ϵ-zlá pre celú vzorku, a teda je to pravdepodobnosť, že nejaká ϵ-zlá hypotéza je v $H[\overline{s}]$.
Pravdepodobnosť, že nejaká ϵ-zlá hypotéza je v $H[\overline{s}]$, je vyjadriteľná

$$\mu^m\{s| \quad H[\overline{s}] \cap B_\epsilon \neq \emptyset\}$$

a je preto menšia než $|H| \cdot (1-\epsilon)^m$. Toto bude menej ako δ za predpokladu, že položíme $m \geq m_0 = \left[\frac{1}{\epsilon} \cdot \ln \frac{|H|}{\delta}\right]$, pretože v tomto prípade

$$|H| \cdot (1-\epsilon)^m \leq |H| \cdot (1-\epsilon)^{m_0} < |H| \cdot \exp(-\epsilon \cdot m_0) \leq |H| \cdot e^{\ln \frac{\delta}{|H|}} = |H| \cdot \frac{\delta}{|H|} = \delta$$

Dokázali sme, že pre ľub. δ, ϵ, t, μ existuje m_0

$$\forall_{m \geq m_0} \forall_t \quad \mu^m\{s \in S(m,t)| \quad H[s] \cap B_\epsilon \neq \emptyset\} < \delta$$

Ak vezmeme komplementárnu udalosť, dostaneme správny záver. □

Je zrejmé, že toto je použiteľná veta. Pokrýva všetky bool. prípady, kde príkladový priestor je $\{0,1\}^n$ (alebo podmnožina) s pevným n. V ľubovoľnej takej situácii konzistentný algoritmus je automaticky PAC. Napríklad algoritmus pre učenie monočlenov a disjunkcií malých monočlenov prezentovaných vyššie sú PAC. Dôkaz nám ďalej hovorí, koľko príkladov postačí na dosiahnutie predpísaných úrovní dôvery a presnosti.

Pre algoritmus monočlenov vieme, že veľkosť $|M_n|$ hypotézového priestoru je 3^n. Preto

$$m_0 = \left[\frac{1}{\epsilon} \ln \frac{|M_n|}{\delta}\right] = \left[\frac{1}{\epsilon}\left(n \cdot \ln 3 + \ln \frac{1}{\delta}\right)\right]$$

je postačujúci počet príkladov na zabezpečenie, že pre pravdepodobnosť väčšiu než $1-\delta$ výstup algoritmu má chybu menšiu než ϵ.

Navyše pre ľub. konečný hypotézový priestor existuje konzistentný učiaci algoritmus: metóda učenia hypotézy očíslovaním, ktorú sme popísali. Teda bezprostredným dôsledkom vety je, že k ľub. konečnému hypotézovému priestoru H existuje učiaci algoritmus, ktorý je PAC.

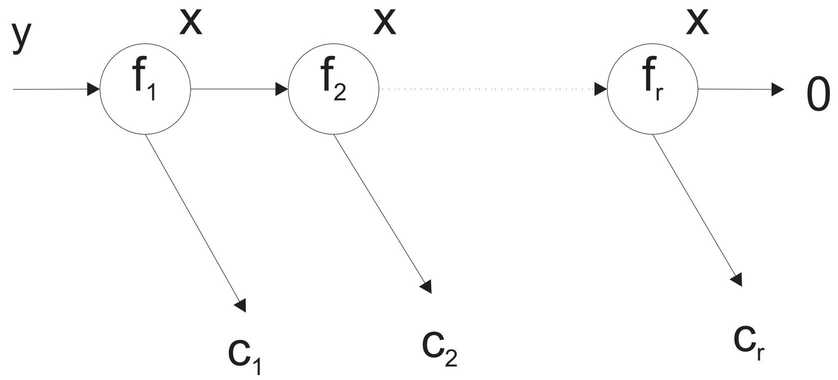
V tomto bode by sa mal čitateľ zadiviť, prečo je to tak. Vytvorili sme komplikovanú podmienku, aby sme dokázali, že je vždy splnená v konečnom prípade, ktorý je najdôležitejší v praxi. Ale praktické predpoklady zavádzajú dodatočné ohraničenie, že počet príkladov by mal byť "ovládateľný" a teda nie je to prípad s metódou učenia očíslovaním. Predpokladajme, napríklad, že hypotézový priestor je množina B_n všetkých booleovských funkcií n premenných. Potom hranica pre vzorku je $m_0 = \left[\frac{2^n}{\epsilon} \ln \frac{2}{\delta}\right]$. Už pri aplik8cii, kde n je okolo 50, dostávame n_0 veľmi veľké. V takýchto prípadoch Veta 2 má malé praktické využitie. Týmto prblémom sa budeme yaoberať] neskôr.

1.3 Rozhodovacie zoznamy

Jeden spôsob popisovania zložitých konceptov je ich vytváranie z menších jednotiek. Viď vytváranie $D_{n,k}$.

V tejto sekcii popíšeme inú metódu konštrukcie, ktorá môže byť aplikovaná na ľub. danú množinu vytvárajúcich blokov.

Definícia 1.3.1 *Nech K je ľubovoľná množina bool. funkcií na $\{0,1\}^n$, n je pevné. Booleovská funkcia f s tým istým oborom ako K sa nazýva* ***rozhodovacím zoznamom*** *(podľa Rivesta) založeným na K, ak môže byť vyhodnotená nasledovne:*
Nech je daný príklad y. Najprv vyhodnotíme $f_1(y)$ pre nejaké pevné $f_1 \in K$.
Ak $f_1(y) = 1$, priradíme do $f(y)$ pevnú hodnotu c_1.
Ak $f_1(y) \neq 1$, vyhodnotíme $f_2(y)$ pre nejaké pevné $f_2 \in K$.
Ak $f_2(y) = 1$, priradíme do $f(y)$ pevnú hodnotu c_2,
inak vyhodnotíme $f_3(y)$, atď.



Obrázok 1.1: Vyhodnotenie funkcie pomocou rozhodovacieho zoznamu

V inom zápise

```
if f_{1}(y)=1 then set f(y)=c_{1}
              else if f_{2}(y)=1 then set f(y)=c_{2}
              ...
              else if f_{r}(y)=1 then set f(y)=c_{r}
                                     else set f(y)=0$
```

Formálnejšie môžeme definovať:

Definícia 1.3.2 *$DL(K)$ je priestor* ***rozhodovacích zoznamov na množine*** *K - je to množina konečných postupností $f = (f_1, c_1), (f_2, c_2), \ldots, (f_r, c_r)$ takých, že $f_i \in K$, $c_i \in \{0,1\}, (1 \leq i \leq r)$. Hodnoty f sú definované*

$$f(y) = \begin{cases} c_j, & ak\ j = min\{i| \quad f_i(y) = 1\}\ existuje \\ 0, & inak \end{cases}$$

Bez újmy na všeobecnosti, keď budeme vyžadovať, aby všetky termy v rozhodovacom zozname boli rôzne, pretože opakovanie danej funkcie $g \in K$ môže byť odstránené bez účinku na vyhodnotenie. Teda dĺžka rozhodovacieho stromu založeného na konečnej množine K je najviac $|K|$ a $|DL(K)|$ je ohraničená funkciou veľkosti $|K|$.

Príklad 1 *Predpokladajme, že $K = M_{3,2}$ je priestor monočlenov dĺžky najviac 2 z 3 premenných. Rozhodovací zoznam*

$$(\langle u_2 \rangle, 1), (\langle u_1 \overline{u_3} \rangle, 0), (\langle \overline{u_1} \rangle, 1)$$

môže byť spracovávaný nasledujúcim spôsobom na príkladovom priestore $\{0,1\}^3$.
Najprv sú vybrané tie príklady, pre ktoré $\langle u_2 \rangle$ *dáva hodnotu 1. Sú to 010, 011, 110, 111.*
ďalej zostanú len tie, ktoré pre $u_1\overline{u_3}$ *dajú 0. Sú to len 100.*
ďalej tie, ktoré pre $\langle \overline{u_1} \rangle$ *dávajú hodnotu 1. Sú to 000, 001.*
Zostáva jediný príklad 101, ktorému je priradená hodnota 0.

Je dôležité poznamenať, že disjunkcia 2 funkcií v K je špeciálny prípad rozhodovacieho zoznamu založeného na K. Explicitne $f \vee g$ je reprezentovaná rozhodovacím zoznamom $(f,1),(g,1)$.

To znamená, že rozhodovací zoznam je zovšeobecnenie disjunkcie. Napríklad, priestor $D_{n,k}$ je obsiahnutý v $DL(M_{n,k})$; v skutočnosti je vlastnou podmnožinou. Iný dôsledok je, že pre dané n, ľubovoľná bool. funkcia n premenných je v nejakom $DL(M_{n,k})$ pre k dostatočne veľké. (Bezprostredne, toto je pravda pre $k = n$ podľa existencie disjunktnej normálnej formy.) Ďalšie detaily v cvičení na konci odseku.

1.4 Konzistentný algoritmus pre rozhodovacie zoznamy

Popíšeme učiaci algoritmus pre $DL(K)$, ktorý pracuje, keď K je ľub. konečná množina. Algoritmus je konzistentný, ale nie je bezpamäťový on-line algoritmus. Samozrejme, učiaci algoritmus očíslovaním má podobné vlastnosti a teda zostáva zistiť, či tento algoritmus je podstatným vylepšením. Tento bod bude prediskutovaný neskôr.

Algoritmus môže byť popísaný nasledovne.

Nech $\overline{s}$ je trénujúca vzorka príkladov označených (x_i, b_i), $1 \leq i \leq m$. V každom kroku konštrukcie požadovaného rozhodovacieho zoznamu niektoré príklady budú vymazané, iné zostanú. Procedúra prebehne cez K hľadajúc funkciu $g \in K$ a bit c taký, že pre všetky zostávajúce príklady x_i, vždy keď $g(x_i) = 1$, tak b_i je konštantná booleovská hodnota c. Dvojica (g,c) je potom vybratá ako ďalší term postupnosti definujúcej rozhodujúci zoznam, a všetky príklady spajúce g sú vymazané. Procedúra je opakovaná, pokým všetky príklady v s neboli vymazané.

Nech $\{g_1, g_2, \dots g_p\}$ je očíslovanie K.

Algoritmus:

```
set I= {1,2, ..., m };
j:=1;
repeat
if forall i in I, g[j](x[i])=1 implikuje b[i]=c
    then begin select (g[j],c);
               delete from I all i for which g[j](x[i])=1;
               j:=1;
         end
   else j:=j+1;
until I is empty set;
```

Príklad 2 $K = M_{5,2}$ *a predpokladajme, že* K *je zaevidované v "slovníkovom poradí" založenom na usporiadaní* $u_1\ u_2\ u_3\ u_4\ u_5\ \overline{u_1}\ \overline{u_2}\ \overline{u_3}\ \overline{u_4}\ \overline{u_5}$ *ako literálov.*

Prvých niekoľko funkcií v slovníku je: monočleny ident. 1

$$\langle\rangle\ \langle u_1\rangle\ \langle u_1u_2\rangle\ \langle u_1u_3\rangle\ \langle \overline{u_1}u_4\rangle$$

Predpokladajme, že trénovacia vzorka je:
$x_1 = 10000 \quad b_1 = 0$
$x_2 = 01110 \quad b_2 = 0$
$x_3 = 11000 \quad b_3 = 0$
$x_4 = 10101 \quad b_4 = 1$
$x_5 = 01100 \quad b_5 = 1$
$x_6 = 10111 \quad b_6 = 1$

Na začiatku vyberieme prvú položku zo slovníka, ktorá splňuje požiadavky podmienky.
$\langle\rangle$ *vylúčime*
$\langle u_1\rangle$ *vylúčime* x_1 *a* x_4 *majú* $b_1 \neq b_4$
$\langle u_1u_2\rangle$ *je splnené len pre* x_3 *a* $b_3 = 0$, *teda vyberieme ako prvý term* $(\langle u_1u_2\rangle, 0)$ *do rozhodovacieho zoznamu*

a vymažeme x_3*.*
ďalšie kroky:
$\langle u_1 u_3\rangle$ *je splnené pre* x_4 *a* x_6 *a* $b_4 = b_6 = 1$*, teda vyberieme* $(\langle u_1 u_3\rangle, 1)$*; vymažeme* x_4 *a* x_6
$\langle u_1\rangle$ *je splnené pre* x_1*,* $b_1 = 0$*, vyberieme* $(\langle u_1\rangle, 0)$
$(\langle \overline{u_1} u_4\rangle, 0)$ *vymaže* x_2
$(\langle\rangle, 1)$ *vymaže* x_5
Vytvorený rozhodovací zoznam je:

$$(\langle u_1 u_2\rangle, 0), (\langle u_1 u_3\rangle, 1), (\langle u_1\rangle, 0), (\langle \overline{u_1} u_4\rangle, 0), (\langle\rangle, 1)$$

Treba poznamenať, že rôzne usporiadania $M_{5,2}$ *dávajú rôzne odpovede.*

Zdá sa nám, že tu pôsobí nejaký záhadný faktor (v príklade), pretože nie je bezprostredne zrejmé, prečo hľadanie g a c je vždy úspešné. Aby sme dokázali, že algoritmus pracuje správne, je nutné ukázať, že vždy keď je daná trénujúca vzorka $\overline{s}$ pre cieľový koncept v $DL(K)$, potom vždy bude nejaká dvojica (g, c), ktorá má požadované vlastnosti.

O trénujúcej vzorke danej vyššie nebolo na začiatku známe, či je kompatibilná s konceptom v $DL(M_{5,2})$, napriek tomu úspešné dokončenie algoritmu ukazuje, že je v skutočnosti v tomto tvare.

Tvrdenie 1 *Predpokladajme, že* K *je hypotézový priestor obsahujúci identicky 1-kovú funkciu. Nech* t *je funkcia v* $DL(K)$ *a nech* S *je konečná množina príkladov. Potom existuje* $g \in K$ *a* $c \in \{0, 1\}$ *také, že:*

1. *množina* $S^g = \{x \in S| \quad g(x) = 1\}$ *je neprázdna;*
2. *pre všetky* $x \in S^g$*,* $\quad t(x) = c$*.*

Dôkaz: Je dané $t \in DL(K)$ také, že jeho reprezentácia pomocou rozhodovacieho zoznamu je:

$$t = (f_1, c_1), (f_2, c_2), \dots, (f_r, c_r)$$

Ak $f_i(x) = 0$ pre všetky $x \in S$ a všetky $i \in \{1, 2, \dots, r\}$, potom všetky príklady v S sú negatívne príklady pre t. V tomto prípade vezmeme g také, že je identicky 1-ková funkcia pre $c = 0$.

Na druhej strane, ak je nejaké i také, že množina $\{x_{i_1}, x_{i_2}, \dots\}$, $x \in S$, pre ktoré $f_i(x_{i_j}) = 1$ nie je prázdna, potom nech q bude najmenším takým indexom, t.j. $q = min\{i_1, i_2, \dots\}$.
Z definície rozhodovacieho zoznamu vyplýva, že $t(x) = c_q$ pre všetky x také, že $f_q(x) = 1$. V tomto prípade môžeme vybrať $g = f_q$ a $c = c_q$. $\Box$

Z tohto tvrdenia vyplýva, že k ľub. trénovacej vzorke pre funkciu v $DL(K)$ existuje vhodný výber dvojice (g, c) pre "prvý term" rozhodovacieho zoznamu. Aplikáciou tohto výsledku rekurzívne vidíme, že algoritmus popísaný vyššie bude vždy úspešný.

1.5 Ďalšie poznámky

Je dôležité zaručiť, že všetky pravdepodobnosti, ktoré sa vyskytujú v definícii tohto odseku, budú dobre definované. Nie sú žiadne problémy, ak príkladový priestor X je spočítateľný, ale pre reálny X musíme stanoviť nejaké merateľné teoretické obmedzenia. Pre ľubovoľné dve hypotézy $t, h \in H$ potrebujeme priradiť pravdepodobnosť chybovej množiny $\{x| \quad h(X) \neq t(x)\}$. Toto sa dá, ak chyby hypotézy sú merateľné funkcie.

Aby sme zaručili, že pre všetky m a pre všetky $t \in H$ množina

$$\{s \in S(m, t)| \quad H[s] \cap B_\epsilon \neq \emptyset\}$$

má dobre definovanú pravdepodobnosť (vzhľadom na μ^m), musia byť zavedené niektoré dodatočné obmedzenia. Postačuje mať H **univerzálne separovateľný**; čitateľa odkážeme na Pollarda (1984) a Blumera et al. (1989).

Poznamenajme, že najviac používané hypotézové priestory majú túto vlastnosť a určite sú predisku-tované v týchto knihách.

Teória potenciálnej naučiteľnosti môže byť ľahko rozšírená na algoritmy, ktoré sú **"takmer konzistentné"**. Pre potenciálnu naučiteľnosť musí byť záruka konzistencie na trénujúcej vzorke dostatočnej dĺžky, čo implikuje dobrú aproximáciu.

Podmienka vyžadovaná pre rozšírenú definíciu je nasledujúca:
Pre ľubovoľnú pevnú konštantu $\alpha < 1$ existuje kladné celé číslo $m_0(\alpha, \delta, \epsilon)$ také, že ak hypotéza h nesúhlasí s najviac $\alpha \cdot \epsilon$ časťou trénujúcej vzorky dĺžky m_0, potom s pravdepodobnosťou aspoň $1 - \delta$ má h skutočnú chybu menšiu ako ϵ.

Táto podmienka môže byť splnená pre ľubovoľný konečný hypotézový priestor H; dôkaz sa získa celkom jednoducho použitím hraníc Angluinovej a Valianta (1979) na určité súčty binárnych čísel.

1.6 Cvičenia:

1. Dokážte, že priestor $H = \{r_\Theta | \quad \Theta \in R\}$ lúčov je potenciálne naučiteľný.
2. Ukážte, že pre hypotézový priestor $D_{n,k} \quad (n \geq k > 1)$ je postačujúce vziať hodnotu $m_0(\delta, \epsilon) = \left[\frac{k}{\epsilon} \cdot \ln 2n + \frac{1}{\epsilon} \cdot \ln \frac{1}{\delta}\right]$ v def. potenciálnej naučiteľnosti.
3. Dokážte, že pre všetky $n \geq k \geq 1$, $D_{n,k} \subset DL(M_{n,k})$. Odvoďte, že ľubovoľná bool. funkcia môže byť reprezentovaná rozhodovacím zoznamom.
4. Skonštruujte bool. funkciu 3 premenných, ktorá nie je $D_{3,2}$, ale je v $DL(M_{3,2})$.
5. Skonštruujte bool. funkciu 3 premenných, ktorá nie je v priestore $DL(M_{3,2})$.
6. Dokážte, že algoritmus pre DL je konzistentný.
7. Dokážte, že pre ľubovoľnú množinu K booleovských funkcií, $3^{|K|} \cdot |K|!$ je horná hranica na $|DL(K)|$.
8. Komplement bool. funkcie h je bool. funkcia $\overline{h}$ taká, že $\overline{h}(x) = 1 \Leftrightarrow h(x) = 0$. Dokážte, že pre ľub. množinu K bool. funkcií obsahujúcu identickú funkciu 1, $h \in DL(K) \Leftrightarrow \overline{h} \in DL(K)$.
   Toto znamená, že $DL(K)$ je uzavretá vzhľadom na komplement.

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
Otázka: Existuje pozitívny príklad pre $\langle\Phi\rangle$?

Problém je 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 to, 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 Čas 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 definovaný 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 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.**

<u>Príklad 1:</u> 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 $O(m * n)$.

<u>Príklad 2:</u> 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 $R_L(m,n) \le mS_L(n)$. Pre algoritmy, ktoré nie sú tohto typu, výpočet času behu môže byť zložitejší.

<u>Príklad 3:</u> 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 $O(m^2n^k)$.

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:

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

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

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 <u>zložitosť vzorky</u> 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 $m \ge m_L(T, \delta, \epsilon)$; 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 $m_0 \ge m_L$, 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 3 *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 $\ln |M_n| = n \ln 3$. 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 ľubovoľný 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 $\ln(N!) \leq N \ln N$, 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: Tréningová vzorka s vyjadrená 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.

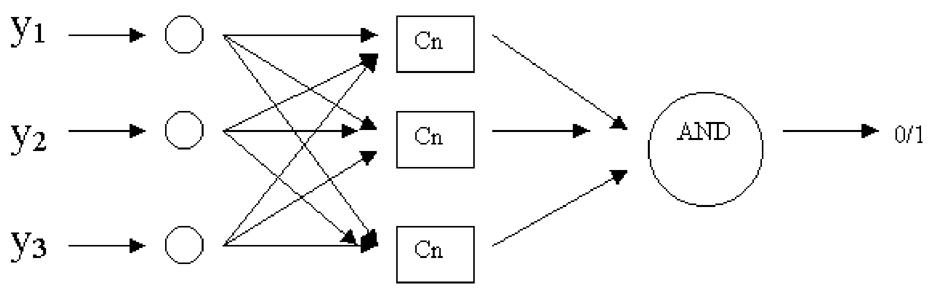
Prvý komentár, ak uvažujeme len tie inštancie problému, v ktorých dĺžka vzorky $\overline{s}$ je ohraničená nejakým fixným polynómom v n, potom máme ohraničený tvar problému konzistencie. Sú také ohraničené tvary, ktoré budeme sledovať v tejto prednáške a uvidíme, že niektoré z týchto problémov sú NP-ťažké.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

2.4 Výsledok o zložitosti

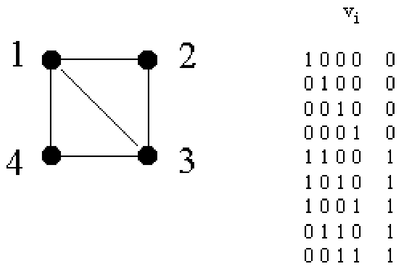
Pitt a Valiant (1988) boli prví, ktorí ukázali príklad hypotézového priestoru H, pre ktorý ohraničený tvar problému konzistencie je NP-ťažký. Nasleduje o niečo zjednodušený popis ich metódy. Nech C_n je priestor klauzúl - množina bool. fukcií n premenných, ktoré môžu byť reprezentované v tvare klauzúl; 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 $G = (V, E), V = \{1, 2, \ldots k\}$. Skonštruujeme tréningovú 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:



<u>Tvrdenie:</u> Existuje funkcia $h \in C_n^k$, ktorá je konzistentná so vzorkou $s(G)$ $\iff$ graf G je k-zafarbiteľný.
Dôkaz:

$\Rightarrow$

Predpokladajme, že $h \in C_n^k$ a je konzistentná s tréningovou vzorkou. Podľa definície h je konjunkcia $h = h_1 \wedge h_2 \wedge \dots h_k$ klauzúl. Pre každý vrchol $i \in V$, $h(v_i) = 0$ a teda musí ex. aspoň 1 klauzula h_f, pre ktorú $h_f(v_i) = 0$. Funkcia $\chi : V \to \{1, 2, \dots k\}$ 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 $\chi : V \to \{1, 2, \dots, k\}$. Pre $1 \leq f \leq k$ definujme h_f ako klauzulu

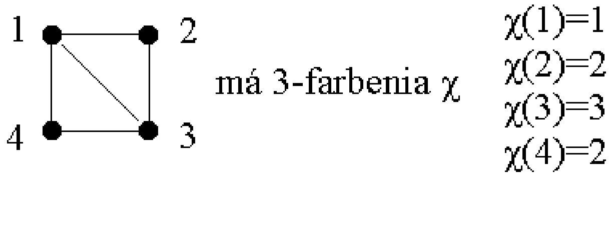
$$\langle \vee u_{i_{\chi(i) \neq f}} \rangle$$

a definujme $h = h_1 \wedge h_2 \wedge \dots \wedge h_k$. 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. $\Box$

<u>Príklad:</u>



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

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

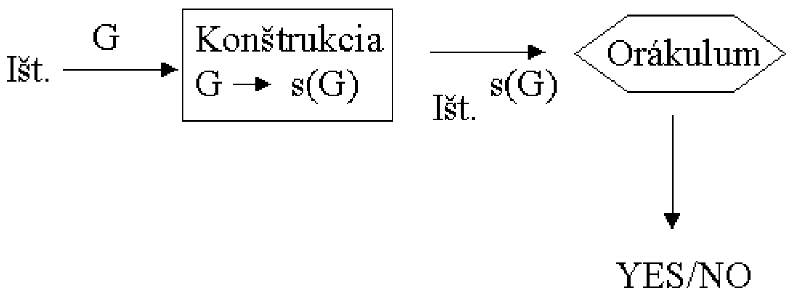
Tento graf nemôže byť zafarbený 2 farbami a teda môžme povedať, že neex. C_4^2, ktorá je konzistentná s trén. vzorkou.

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 tréningová vzorka $\overline{s}$ s označenými n-bitovými vektormi. Existuje funkcia v C_n^k, ktorá je konzistentná s $\overline{s}$?

Dôkaz, že C^k - konzistencia je NP-ťažký problém je pozorovateľný podľa 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, že 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.

2.5 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 $\overline{s}$ je tréningová vzorka pre hypotézu v C_n, potom $L(\overline{s}) \in H_n$. Hausler, Littlestone a Warmuth (1988) zaviedli pojem účinná predikcia postupne študovaný Pittom a Warmuthom (1988,1990) a Hausslerom (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 majú túto vlastnosť.

2.6 Úlohy:

1. Prečo je obyčajne vhodné uvažovať veľkosť vstupu v tvare $\lg n$ namiesto n, keď uvažujeme o otázkach efektívnosti?

2. Nasledujúci algoritmus je rýchlym algoritmom pre výpočet m-tej mocniny daného čísla u. (výstupom je finálna hodnota uložená v *bot*.)

```
bot:=1; top:=u; q:=m;
while q>0 do
begin
   if q mod 2 =1 then bot:=top*bot;
   top:=sqr(top);
   q:=q div 2;
end;
```

   Ukážte, že efektívnosť algoritmu je $O(s)$, kde s je miera veľkosti m ako bolo povedané v predchádzajúcom príklade.

3. Navrhnite algoritmus, ktorý rozhodne, či daný n-bitový reťazec je palindrom a odhadnite jeho efektívnosť vzhľadom na veľkosť vstupu n.

4. Nech $G = (V, E)$ je graf s množinou vrcholov $V = \{1, 2, 3, 4, 5\}$ a množinou hrán $E = \{12, 13, 15, 23, 25, 34, 35\}$. Vytvorte odpovedajúcu tréningovú vzorku $s(G)$ podľa postupu v dôkaze. Nájdite najmenšiu možnú hodnotu k, pre ktorú je funkcia $h \in C_5^k$ konzistentná s $s(G)$ a danú formulu vyjadrite explicitne.

5. Nech $C_n = C_n^1$, čo je priestor booleovských funkcií na $\{0,1\}^n$, ktorý môže byť reprezentovaný jednou klauzulou. Sformulujte konzistentný učiaci algoritmus pre C_n, ktorý je "duálny" k štandardnému algoritmu pre monočleny a vyhodnodtťe tvrdenie, že jeho čas behu je polynomiálny v m a n.

6. Nasledujúci problém je známy ako NP-úplný (Lovász, 1973).

   SET SPLITTING
   Inštancia: Dvojica (U, S), kde U je konečná množina a S je zbierka množín so zjednotením U.
   Otázka: Existujú $U_1, U_2 \subset$ také, že $U = U_1 U_2$ a také, že žiadna množina v S neleží celá v U_1 alebo U_2?

   Redukujte SET SPLITTING na C^2-CONSISTENCY a uvažujte o tom, či problem pre C^2 je NP-ťažký (Pitt a Valiant, 1988).