Induktív logikai programozás

Varga Péter

2004. október 25.

Indukció

- def.: következtetés az egyesből az általánosra
- Francis Bacon óta a tudomány eszközének tartják

Indukció

- def.: következtetés az egyesből az általánosra
- Francis Bacon óta a tudomány eszközének tartják
- matematikai indukció vs. empirikus indukció

Indukció

- def.: következtetés az egyesből az általánosra
- Francis Bacon óta a tudomány eszközének tartják
- matematikai indukció vs. empirikus indukció
- az indukció konfirmáltsági foka:

$$\forall x.H(x) \supset F(x) \longleftarrow H(a) \wedge F(a)$$

Indukció

- def.: következtetés az egyesből az általánosra
- Francis Bacon óta a tudomány eszközének tartják
- matematikai indukció vs. empirikus indukció
- az indukció konfirmáltsági foka:

$$\forall x.H(x) \supset F(x) \longleftarrow H(a) \wedge F(a)$$

- az indukció Hempel-féle paradoxona:

$$\forall x.H(x) \supset F(x) \equiv \forall x.\neg F(x) \supset \neg H(x) \longleftarrow \neg F(b) \wedge \neg H(b)$$

Fogalmak induktív gépi tanulása: az alapok

Fogalom: $\mathcal{U} : \mathcal{C} \subseteq \mathcal{U}$

Fogalmak induktív gépi tanulása: az alapok

Fogalom: $\mathcal{U} : \mathcal{C} \subseteq \mathcal{U}$

Fogalomleíró nyelv Pl. elsőrendű logika nyelve

Fogalmak induktív gépi tanulása: az alapok

Fogalom: $\mathcal{U} : \mathcal{C} \subseteq \mathcal{U}$

Fogalomleíró nyelv Pl. elsőrendű logika nyelve

Objektumleíró nyelv Pl. attribútum-érték párok halmaza

Fogalmak induktív gépi tanulása: az alapok

Fogalom: $\mathcal{U} : \mathcal{C} \subseteq \mathcal{U}$

Fogalomleíró nyelv Pl. elsőrendű logika nyelve

Objektumleíró nyelv Pl. attribútum-érték párok halmaza

Fogalom definíciója: extenzionális vagy intenzionális

Fogalmak induktív gépi tanulása: az alapok

Fogalom: $\mathcal{U} : \mathcal{C} \subseteq \mathcal{U}$

Fogalomleíró nyelv Pl. elsőrendű logika nyelve

Objektumleíró nyelv Pl. attribútum-érték párok halmaza

Fogalom definíciója: extenzionális vagy intenzionális

Címkézett példák

Fogalmak induktív gépi tanulása: az alapok

Fogalom: $\mathcal{U} : \mathcal{C} \subseteq \mathcal{U}$

Fogalomleíró nyelv Pl. elsőrendű logika nyelve

Objektumleíró nyelv Pl. attribútum-érték párok halmaza

Fogalom definíciója: extenzionális vagy intenzionális

Címkézett példák

Hipotézis

Fogalmak induktív gépi tanulása: az alapok

Fogalom: $\mathcal{U} : \mathcal{C} \subseteq \mathcal{U}$

Fogalomleíró nyelv Pl. elsőrendű logika nyelve

Objektumleíró nyelv Pl. attribútum-érték párok halmaza

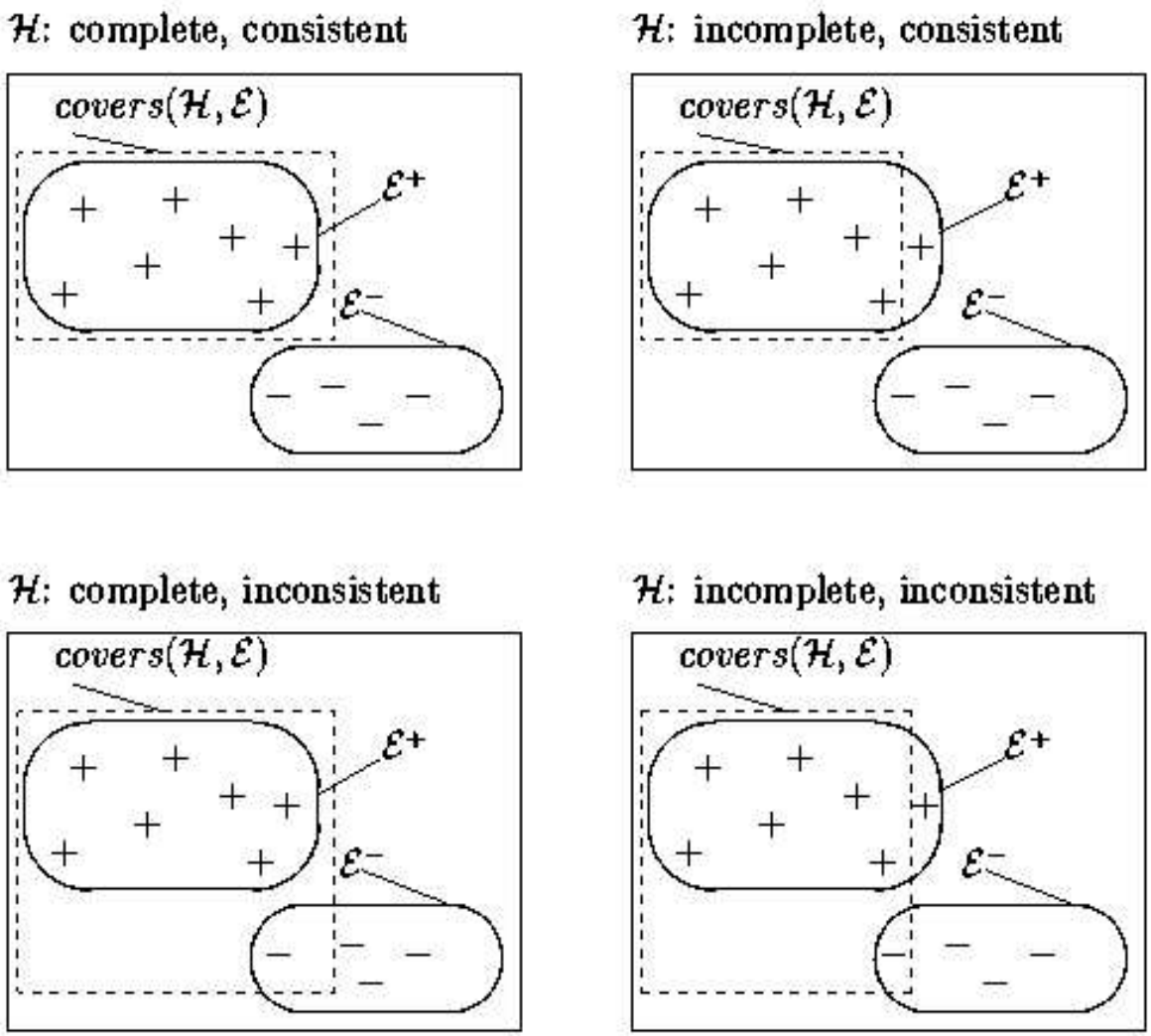
Fogalom definíciója: extenzionális vagy intenzionális

Címkézett példák

Hipotézis

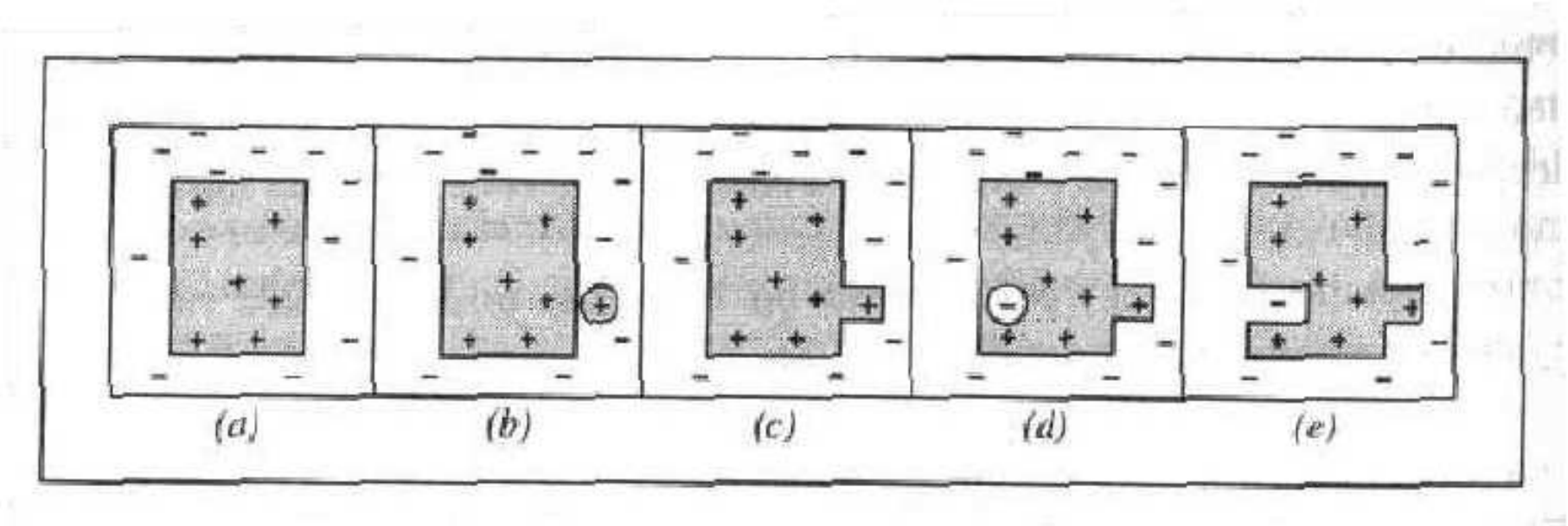
Háttértudás

A hipotézis illeszkedési lehetőségei



A hipotézis és a példahalmaz lehetséges illeszkedési viszonyai

A hipotézis illeszkedésének folyamata



A hipotézis a pozitív példák befogadásával és a negatív példák kizárásával alakul

Az illeszkedés még nem minden

Az illeszkedés pontossága: erről volt szó az előbbiekben

A hipotézis formája: pl. hány literált tartalmaz

Statisztikai szignifikancia: milyen regularitást emel ki a hipotézis

Információs tartalom: a tanulandó fogalom a priori valószínűségét is figyelembe veszi

Az induktív logikai programozás helye

Gépi tanulás

- evolúciós tanulás
- konnekcionista tanulás
- induktív tanulás (fogalmakra)
  - attribútum-érték tanulók (propozionális tanulók)
    - döntési listák
    - if-then szabályok
  - reláció-tanulók
    - elsőrendű logikai tanulók
    - Horn-klóz tanulók: **induktív logikai programozás**

Az ILP-sajátosságai (1.): a θ tartalmazási háló

A keresési tér struktúrálására a θ tartalmazási reláció szerint részben rendezést definiálunk. A reláció definíciója:

Behelyettesítés: $\theta = \{X_1/t_1, \ldots, X_k/t_k\}$

θ-tartalmazás: Legyen c, c' (logikai) program-klózok, θ egy behelyettesítés. A c θ-tartalmazza c'-t, ha $c\theta \subseteq c'$.

Vegyük észre, hogy a θ-tartalmazásból következik $c \models c'$, de ez fordítva nem igaz. Erre példa:

$$c = \mathit{list}([V \mid W]) \leftarrow \mathit{list}(W),\ c' = \mathit{list}([X, Y \mid Z]) \leftarrow \mathit{list}(Z)$$

Az ILP-sajátosságai (2.): Mire jó a θ-tartalmazás?

- A változóátnevezés erejéig egyedi legkisebb felső és legnagyobb alsó elemek lesznek
- Szintaktikai a definíció, nem támaszkodik a háttértudásra
- A keresési teret vághatjuk, ha c inkonzisztens, mert
  - Ha $c > c'$ és $\mathcal{B} \cup \{c\} \models e$, akkor $\mathcal{B} \cup \{c'\} \models e$ is igaz.
- A keresési teret vághatjuk, ha c nem teljes, mert
  - Ha $c > c'$ és $\mathcal{B} \cup \{c\} \not\models e$, akkor $\mathcal{B} \cup \{c'\} \not\models e$ is igaz.

Az ILP két alaplépése

Most már láthatjuk az ILP két alapvető lépését:

Az ILP két alaplépése

Most már láthatjuk az ILP két alapvető lépését:

Szaturáció: a példák telítése, azaz általánosítása: a bottom-up lépés

Az ILP két alaplépése

Most már láthatjuk az ILP két alapvető lépését:

Szaturáció: a példák telítése, azaz általánosítása: a bottom-up lépés

Redukció: lefelé utazni a finomítási gráfon: a top-down lépés

Az ILP sajátosságai (3.): Egy szaturációs technika (1.)

Keressük meg a legfelső elemet: $lgg(c_1, c_2)$

- Termek esetében:
  - $lgg(t, t) = t$,
  - $lgg(f(s_1, \ldots, s_n), f(t_1, \ldots, t_n)) = f(lgg(s_1, t_1), \vdots, lgg(t_n, s_n))$,
  - $lgg(f(s_1, \ldots, s_n), f(t_1, \ldots, t_m)) = V\ (V \in Vars)$, ha $f \neq g$,
  - $lgg(s, t) = V\ (V \in Vars)$, ha $s \neq t$ és legalább egyikük változó
- Atomok esetében:
  - $lgg(p(s_1, \ldots, s_n), p(t_1, \ldots, t_n)) = p(lgg(t_1, s_1), \ldots, p(t_n, s_n))$
  - máskülönben definiáltlan
- Nem-atomi literálok esetében:
  - Ha $L_1 = \overline{A_1}$ és $L_2 = \overline{A_2}$, akkor $lgg(L_1, L_2) = \overline{lgg(A_1, A_2)}$.
  - máskülönben definiáltlan

Az ILP sajátosságai (4.): Egy szaturációs technika (2.)

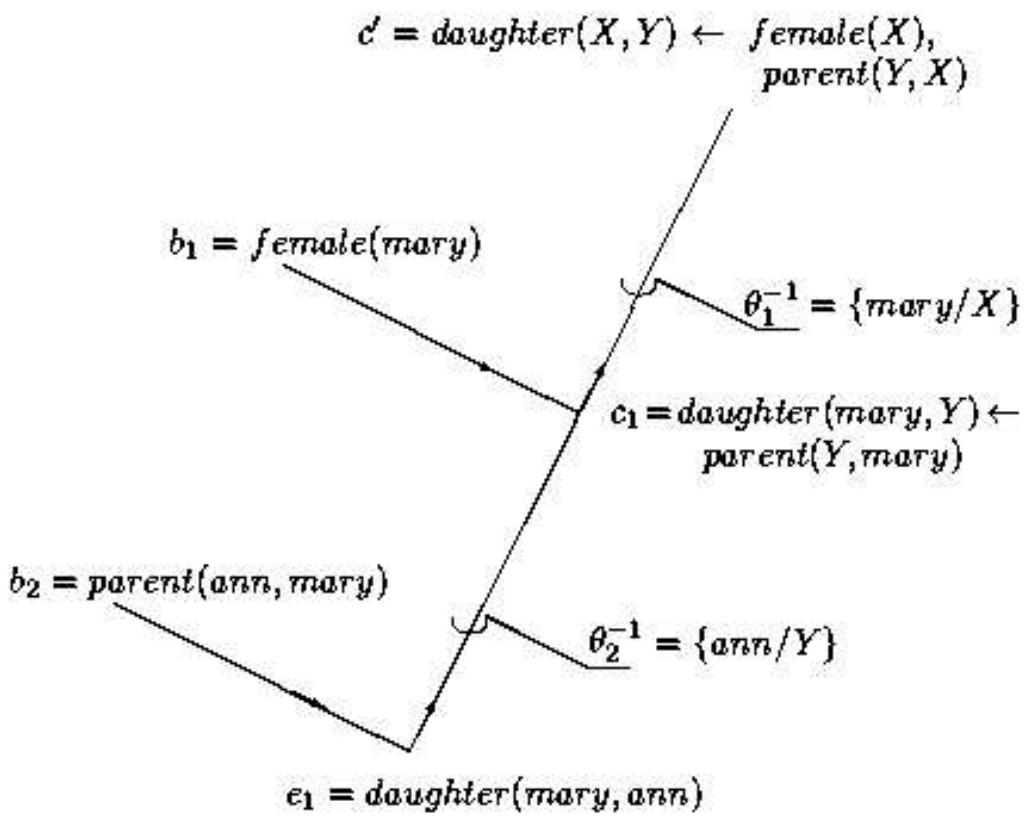
- Klózok esetében:
  - Legyen $c_1 = \{L_1, \ldots, L_n\}$ és $c_2 = \{K_1, \ldots, K_m\}$.
    $lgg(c_1, c_2) = \{L_{ij} = lgg(L_i, K_j) \mid L_i \in c_q, K_j \in c_2$ és $lgg(L_i, K_j)$ definiált$\}$.

Vegyük figyelembe a háttértudást is:

- $rlgg(A_1, A_2) = lgg((A_1 \leftarrow K), (A_2 \leftarrow K))$, ahol K a grounded háttértudást jelöli.

Az ILP sajátosságai (5.): Egy másik szaturációs technika

Egy másik szaturációs technika az inverz rezolúció:



Az ILP sajátosságai...

Sok technikát lehetne még ismertetni a szimbolikus tanulás és a tételbizonyítás témaköréből,

Az ILP sajátosságai...

Sok technikát lehetne még ismertetni a szimbolikus tanulás és a tételbizonyítás témaköréből, de . . .

lássuk inkább a gyakorlatot!

Az ILP-rendszerek felosztása

Egy predikátum vs. egész elmélet tanulása Egy predikátum (több klóza), avagy több predikátumból álló elmélet.

Batch vs. inkrementális tanulás A tanuláis folyamat során a példákat egyenként, vagy egyszerre használja.

Interaktív A felhasználó felügyeli az általánosítási folyamatot és módosíthatja az osztályozást is.

Kiinduló hipotézis Meglévő elmélet revíziójára használható.

Két gyakoribb típus:

- Nem-interaktív batch rendszerek kiinduló hipotézis nélkül
- Inkrementális interaktív elmélet-módosító rendszerek

ILP a gyakorlatban: az ALEPH rendszer

- A Learning Engine for Proposing Hypotheses
- Fejlesztő: Ashwin Srinivasan (Oxford Univ.)
- Folyamatos fejlesztés 1993 óta (Legutolsó módosítás 2004. okt. 10-én.)
- Korábbi neve: P-Prolog
- Használható Prolog rendszerek:
  - YAP: egy nagyon gyors Prolog-implementáció
  - SWI-Prolog: a méltán népszerű LGPL-licenszű Prolog
- Non-profit célokra szabadon használható
- Ténylegesen használják is (biológiai és NLP-alkalmazások)

Az ALEPH képességei

- A fogalom- és tényleíró nyelv a Prolog
- Részletesen specifikálható nyelvi bias
- Inkrementális és batch-tanulás
- Interaktív és nem-interaktív tanulás
- Több predikátum együttes tanulása (kísérleti)
- Kiegészítések: osztályozó, döntési fa, korlátok és predikátum-szignatúrák tanulása
- 10 keresési bias és 12 beépített értékelés
- Jól bővíthető (Saját függvények értékelésre, heurisztikákra, megjelenítésre, hipotézis-validálásra vagy pruningra; saját kényszerek stb.)
- Non-grounded példák megadása (a pozitív még kísérleti stádiumban)
- Nyílt kód, tisztességes dokumentáció

Egyszerű példa az ALEPH (és az ILP) működésére

Célunk: A `grandfather/2` predikátum megtanulása.

Adott: A magyar fejedelmek szülői relációi az első magyar királyig, valamint a férfitagok neme.

Nyelvi bias: Két predikátum

- `parent/2`
- `male/1`

Az egyszerű példa: A háttértudás (1.)

```
parent(imre,istvan).
parent(istvan,geza).
parent(geza,taksony).
parent(taksony,zoltan).
parent(zoltan,arpad).
parent(sarolt,istvan).
parent(gizella,imre).
male(istvan).
male(geza).
male(arpad).
male(zoltan).
male(taksony).
male(imre).
```

Az egyszerű példa: A háttértudás (2.)

```
:-modeh(1,grandfather(+person,+person)).
:-modeb(1,male(+person)).
:-modeb(*,parent(+person,+person)).
:-modeb(*,parent(+person,-person)).
:-determination(grandfather/2,parent/2).
:-determination(grandfather/2,male/1).
:-set(i,2).
:-set(clauselength,4).
:-set(newvars,1).
:-set(minpos,2).
```

Az egyszerű példa: A példák

Pozitív példák (`grandfather.f`)

```
grandfather(imre,geza).
grandfather(istvan,taksony).
grandfather(geza,zoltan).
grandfather(taksony,arpad).
```

A negatív példák (`grandfather.n`)

```
grandfather(imre,istvan).
grandfather(gizella,istvan).
grandfather(imre,gizella).
grandfather(imre,sarolt).
grandfather(taksony,zoltan).
```

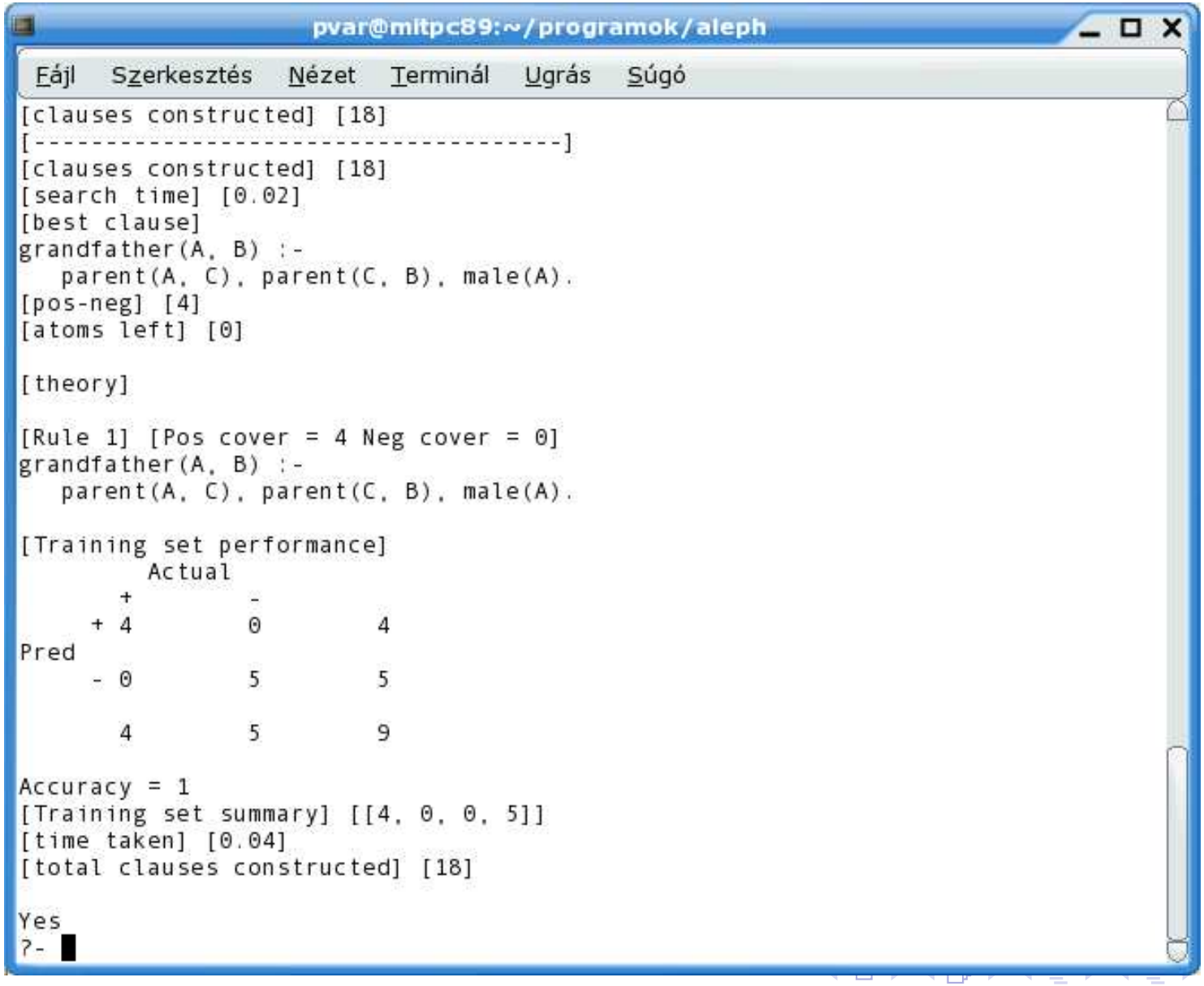
Az egyszerű példa: az ALEPH rendszer

pvar@mitpc89:~/programok/aleph

Fájl Szerkesztés Nézet Terminál Ugrás Súgó

```
pvar@mitpc89:~> cd programok/aleph/
pvar@mitpc89:~/programok/aleph> pl
Welcome to SWI-Prolog (Multi-threaded, Version 5.2.13)
Copyright (c) 1990-2003 University of Amsterdam.
SWI-Prolog comes with ABSOLUTELY NO WARRANTY. This is free software,
and you are welcome to redistribute it under certain conditions.
Please visit http://www.swi-prolog.org for details.

For help, use ?- help(Topic). or ?- apropos(Word).

?- [aleph].
%   library(pce) loaded into pce 0.13 sec, 162,224 bytes
%  library(broadcast) compiled into broadcast 0.16 sec, 188,500 bytes
%  library(time) compiled into time 0.00 sec, 2,612 bytes


A L E P H
Version 5
Last modified: Sun Oct 10 06:59:50 BST 2004

Manual: http://www.comlab.ox.ac.uk/oucl/groups/machlearn/Aleph/index.html

% aleph compiled 0.64 sec, 773,500 bytes

Yes
?- read_all(grandfather).
% grandfather.b compiled 0.00 sec, 3,328 bytes
[consulting pos examples] [grandfather.f]
[consulting neg examples] [grandfather.n]

Yes
?- induce.
```

Az egyszerű példa: az eredmények



Az ALEPH paraméterezése: a nyelvi bias

`determination/2`: Melyik klóz törzsében melyik klózt lehet felhasználni.

`mode/2`: A klózra hányszor és milyen szignatúrával lehet hivatkozni.

- Ha nem egész elméletet keresünk, akkor csak egy predikátumra vonatkozó `determination` deklarációk lesznek érvényesek.
- A szignatúra specifikációjának szintaxisa:
  - **+T**: bemenő paraméter
  - **-T**: kimenő paraméter
  - **#T**: konstans paraméter
- Rekurzív definícióra is lehetséges.

Az ALEPH paraméterezése: alapvető keresési bias-ok

`set(clauselength,+V)`: Felső korlát a klózban szereplő literálok számára (tehát a fej is számít)

Az ALEPH paraméterezése: alapvető keresési bias-ok

`set(clauselength,+V)`: Felső korlát a klózban szereplő literálok számára (tehát a fej is számít)

`set(depth,+V)`: Milyen mélységű SLD-rezolúcióba szabad belemenni a hipotézis ellenőrzésekor

Az ALEPH paraméterezése: alapvető keresési bias-ok

`set(clauselength,+V)`: Felső korlát a klózban szereplő literálok számára (tehát a fej is számít)

`set(depth,+V)`: Milyen mélységű SLD-rezolúcióba szabad belemenni a hipotézis ellenőrzésekor

`set(explore,+V)`: Kikapcsolja a pruning-ot és kevésbé mohón keres.

Az ALEPH paraméterezése: alapvető keresési bias-ok

`set(clauselength,+V)`: Felső korlát a klózban szereplő literálok számára (tehát a fej is számít)

`set(depth,+V)`: Milyen mélységű SLD-rezolúcióba szabad belemenni a hipotézis ellenőrzésekor

`set(explore,+V)`: Kikapcsolja a pruning-ot és kevésbé mohón keres.

`set(i,+V)`: Milyen mélységű változók szerepelhetnek a klózban (a fej változóinak mélysége egy, sít.)

Az ALEPH paraméterezése: alapvető keresési bias-ok

`set(clauselength,+V)`: Felső korlát a klózban szereplő literálok számára (tehát a fej is számít)

`set(depth,+V)`: Milyen mélységű SLD-rezolúcióba szabad belemenni a hipotézis ellenőrzésekor

`set(explore,+V)`: Kikapcsolja a pruning-ot és kevésbé mohón keres.

`set(i,+V)`: Milyen mélységű változók szerepelhetnek a klózban (a fej változóinak mélysége egy, sít.)

`set(minpos,+V)`: Minimum hány pozitív példát kell lefednie a hipotézis-klóznak (nagyon hasznos a ground clause-ok kizárására).

Az ALEPH paraméterezése: alapvető keresési bias-ok

`set(clauselength,+V)`: Felső korlát a klózban szereplő literálok számára (tehát a fej is számít)

`set(depth,+V)`: Milyen mélységű SLD-rezolúcióba szabad belemenni a hipotézis ellenőrzésekor

`set(explore,+V)`: Kikapcsolja a pruning-ot és kevésbé mohón keres.

`set(i,+V)`: Milyen mélységű változók szerepelhetnek a klózban (a fej változóinak mélysége egy, sít.)

`set(minpos,+V)`: Minimum hány pozitív példát kell lefednie a hipotézis-klóznak (nagyon hasznos a ground clause-ok kizárására).

`set(newvars,+V)`: Hány egzisztenciálisan kötött változót vezethessen be klóz törzsében.

Az ALEPH paraméterezése: egyszerűbb keresési stratégiák

`set(search,bf)`: Egyszerű szerkezetű klózokat preferálja (ez az alapértelmezett stratégia)

Az ALEPH paraméterezése: egyszerűbb keresési stratégiák

`set(search,bf)`: Egyszerű szerkezetű klózokat preferálja (ez az alapértelmezett stratégia)

`set(search,id)`: Iteratívan mélyülő keresés a `clauselength` korlátig

Az ALEPH paraméterezése: egyszerűbb keresési stratégiák

`set(search,bf)`: Egyszerű szerkezetű klózokat preferálja (ez az alapértelmezett stratégia)

`set(search,id)`: Iteratívan mélyülő keresés a `clauselength` korlátig

`set(search,rls)`: Szimulált lehűtés (a paraméterei részletesen beállíthatók, pl. kezdőhőmérséklet, generálási valószínűség klózhossz függvényében stb.)

Az ALEPH paraméterezése: egyszerűbb keresési stratégiák

`set(search,bf)`: Egyszerű szerkezetű klózokat preferálja (ez az alapértelmezett stratégia)

`set(search,id)`: Iteratívan mélyülő keresés a `clauselength` korlátig

`set(search,rls)`: Szimulált lehűtés (a paraméterei részletesen beállíthatók, pl. kezdőhőmérséklet, generálási valószínűség klózhossz függvényében stb.)

`set(search,heuristic)`: Egyszerű heurisztikus keresés

Az ALEPH paraméterezési: egyszerűbb értékelések

`set(evalfn,accuracy)`: Csak a példákra figyel: $P/)P + N)$. (Ez az alapértelmezett.)

`set(evalfn,compression)`: Figyelembe veszi a klóz hosszát (literálok számát) is.

Ezen kívül statisztikai értékelő függvények, valamint ILP-publikációkban megjelent függvények is rendelkezésünkre állnak.

Mikor használható az ILP?

Szabad diszkusszió...

Mikor használható az ILP? – a saját véleményem

Mikor használható az ILP? – a saját véleményem

- Ember által is értelmezhető szabályokat akarunk

Mikor használható az ILP? – a saját véleményem

- Ember által is értelmezhető szabályokat akarunk
- ... avagy illeszkedni szeretnénk már létező szabályok struktúrájához.

Mikor használható az ILP? – a saját véleményem

- Ember által is értelmezhető szabályokat akarunk
- ... avagy illeszkedni szeretnénk már létező szabályok struktúrájához.
- A megtanulandó regularitások intenzionálisan definiált predikátumokra támaszkodnak.

Mikor használható az ILP? – a saját véleményem

- Ember által is értelmezhető szabályokat akarunk
- ... avagy illeszkedni szeretnénk már létező szabályok struktúrájához.
- A megtanulandó regularitások intenzionálisan definiált predikátumokra támaszkodnak.
- Nagy vonalakban látható a megtanulandó szabályok "struktúrája"

Mikor használható az ILP? – a saját véleményem

- Ember által is értelmezhető szabályokat akarunk
- ... avagy illeszkedni szeretnénk már létező szabályok struktúrájához.
- A megtanulandó regularitások intenzionálisan definiált predikátumokra támaszkodnak.
- Nagy vonalakban látható a megtanulandó szabályok "struktúrája"
- ... csak éppen "sok van belőlük"

Mikor használható az ILP? – a saját véleményem

- Ember által is értelmezhető szabályokat akarunk
- ... avagy illeszkedni szeretnénk már létező szabályok struktúrájához.
- A megtanulandó regularitások intenzionálisan definiált predikátumokra támaszkodnak.
- Nagy vonalakban látható a megtanulandó szabályok "struktúrája"
- ... csak éppen "sok van belőlük"

"sok kicsi, ismert alakú", sőt léteznek ILP-módszerek zajos adatokra is...

Egy életszagúbb példa (1.)

alphabetic_past_data.f

```
past([a, b, a, n, d, o, n],[a, b, a, n, d, o, n, e, d]).
past([a, b, e, t],[a, b, e, t, t, e, d]).
past([a, b, o, u, n, d],[a, b, o, u, n, d, e, d]).
past([a, b, s, o, r, b],[a, b, s, o, r, b, e, d]).
past([a, c, c, e, p, t],[a, c, c, e, p, t, e, d]).
past([a, c, c, o, m, p, a, n, y],[a, c, c, o, m, p, a, n,
past([a, c, c, o, m, p, l, i, s, h],[a, c, c, o, m, p, l,
past([a, c, c, o, u, n, t],[a, c, c, o, u, n, t, e, d]).
past([a, c, c, u, s, e],[a, c, c, u, s, e, d]).
past([a, c, h, e],[a, c, h, e, d]).
past([a, c, h, i, e, v, e],[a, c, h, i, e, v, e, d]).
```

Egy életszagúbb példa: részlet a szabályokból

```
[Rule 1] [Pos cover = 688 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[e,d]).
[Rule 2] [Pos cover = 17 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[t,e,d]).
[Rule 4] [Pos cover = 440 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[d]).
[Rule 8] [Pos cover = 10 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[r,e,d]).
[Rule 10] [Pos cover = 20 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[]).
[Rule 13] [Pos cover = 11 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[g,e,d]).
[Rule 26] [Pos cover = 29 Neg cover = 0]
```

```
past(A,B) :-
   suff(A,B,[],[p,e,d]).
[Rule 30] [Pos cover = 2 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[l,e,d]).
[Rule 33] [Pos cover = 2 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[t]).
[Rule 38] [Pos cover = 5 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[n,e,d]).
[Rule 43] [Pos cover = 7 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[m,e,d]).
[Rule 64] [Pos cover = 5 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[b,e,d]).
[Rule 88] [Pos cover = 3 Neg cover = 0]
past(A,B) :-
   suff(A,B,[],[d,e,d]).
```

Ajánlott irodalom

- N. Lavrac and S. Dzeroski. *Inductive Logic Programming*: Techniques and Applications. Ellis Horwood, New York, 1994. (`http://www-ai.ijs.si/SasoDzeroski/ILPBook/`)
- Saso Dzeroski and Nada Lavrac, editors *Relational Data Mining* Springer, Berlin, 2001
- Stuart J. Russell, Peter Norvig: *Mesterséges intelligencia modern megközelítésben* Panem Kiadó, 1999. (az új kiadás magyar fordítása előkészületben) *18. fejezet*

Induktív logikai programozás

Köszönöm a figyelmet, várom a kérdéseiteket...