Logica descriptivă

Logicile descriptive (description logic DL) sunt notații concepute cu scopul de a ușura descrierea definițiilor și a proprietăților categoriilor (în comparație cu reprezentarea prin cadre).
Accentul se pune pe aspectele declarative ale reprezentării orientat obiect, revenind la concepte precum predicate și implicații din FOL.

In FOL reprezentăm categorii de obiecte prin intermediul predicatelor simple: Company(x), Boat(x), Mother(x). Pentru a reprezenta construcții/tipuri mai interesante precum "un vânător-colecționar" sau "un om ai cărui copii sunt de sex feminin" avem nevoie de predicate cu o structură internă compusă.
De exemplu, dacă am avea predicatul compus Hunter&Gatherer(x) ne-am aștepta ca pentru orice x pentru care Hunter&Gatherer(x) este adevărat, predicatele Hunter(x) și Gatherer(x) să fie adevărate.
Această legătură este definită de înțelesul atribuit predicatului compus Hunter&Gatherer.
Putem avea un nume simplu ca abreviere a unui concept compus.
De exemplu, Teenager este sinonim cu PersonWithAgeBetween13and19.

Avem nume de categorii (Hunter, Teenager) ce descriu clase de obiecte de bază și nume relaționale (Age) ce descriu obiecte ce sunt părți/proprietăți ale altor obiecte. In DL ne referim la primul tip prin "concept" și la al 2-lea tip prin "rol" (în reprezentarea cu cadre sunt numite cadre/slot-uri).
Hunter&Gatherer este o specializare a lui Hunter. In DL, majoritatea raționamentelor efectuate este centrată în jurul stabilirii automate a relației de generalizare.
Rolurile pot avea mai multe filler-e (spre deosebire de slot-urile cadrelor).
Pe lângă concepte și roluri, în DL avem și constante (johnSmith).

Un limbaj descriptiv

In DL avem două tipuri de simboluri — logice, cu înțeles predefinit
 nonlogice, ce depind de aplicație

4 tipuri de simboluri logice — punctuație "[", "]", "(", ")"
 întregi pozitivi 1, 2, 3 ...
 operatori pentru formare de concepte: "ALL", "EXISTS", "FILLS", "AND"
 conectori $\sqsubseteq$, $=$, $\rightarrow$

3 tipuri de simboluri nonlogice
 concepte atomice - încep cu majusculă
 Person, FatherOfOnlyGirls
 - există un concept atomic special Thing
 roluri - scrise precum conceptele atomice dar precedate de :
 :Age , :Mother
 constante - încep cu literă mică table17, johnSmith

In DL există 4 tipuri de expresii sintactice valide: constante, roluri, concepte și propoziții.

c pentru constante
r pentru roluri
d, e pentru concepte
a pentru concepte atomice

Mulțimea de concepte din DL satisface următoarele:
- orice concept atomic este un concept;
- dacă r rol și d concept, atunci [ALL r d] concept;
- dacă r rol, n întreg pozitiv, atunci [EXISTS n r] concept;
- dacă r rol, c constantă, atunci [FILLS r c] concept;
- dacă $d_1, \ldots, d_n$ concepte, atunci [AND $d_1 \ldots d_n$] concept.

Există 3 tipuri de propoziții în DL:

- dacă d_1, d_2 concepte, atunci $(d_1 \sqsubseteq d_2)$ este propoziție;
- dacă d_1, d_2 concepte, atunci $(d_1 = d_2)$ propoziție;
- dacă c constantă, d concept, atunci $(c \rightarrow d)$ propoziție.

O bază de cunoștințe în DL este o colecție de propoziții.

Constantele reprezintă indivizi în domeniul aplicației; conceptele atomice reprezintă categorii sau clase de indivizi; rolurile reprezintă relații binare între indivizi.

Înțelesul unui concept complex derivă din părțile constituente.

De exemplu [EXISTS n r] reprezintă clasa de indivizi ce satisfac relația r în raport cu cel puțin alți n indivizi.

[EXISTS 1 :Child] reprezintă persoanele care nu sunt fără copii.

Dacă c reprezintă un individ, [FILLS r c] reprezintă indivizii ce se află în relație r cu c.

[FILLS :Cousin george] reprezintă pe cineva al cărui văr este George

Dacă conceptul d reprezintă o clasă de indivizi, [ALL r d] reprezintă indivizii care sunt în relația r cu indivizi din clasa reprezentată de d.

[ALL :Employee UnionMember] descrie companiile ai căror angajați sunt membri de sindicat.

Conceptul [AND $d_1 \ldots d_n$] reprezintă orice este descris de $d_1, \ldots, d_n$.

```
[AND Wine
     [FILLS :Color red]
     [EXISTS 2 :GrapeType]]
```

– reprezintă un vin de culoare roșie, ce are cel puțin 2 tipuri de struguri în el.

```
[AND Company
     [EXISTS 7 :Director]
     [ALL :Manager [AND Women
                        [FILLS :Degree phD]]]
     [FILLS :MinSalary $20/hour]]
```

În DL, propozițiile pot fi adevărate sau false (ca în FOL).

d_1, d_2 concepte, c constantă

$(d_1 \sqsubseteq d_2)$ spune că d_1 este subsumat (inclus) lui d_2, adică toți indivizii ce satisfac d_1, satisfac d_2.

(Surgeon $\sqsubseteq$ Doctor) orice chirurg este doctor

$(d_1 = d_2)$ spune că d_1 este echivalent cu d_2, adică indivizii ce satisfac d_1 sunt exact aceia care satisfac d_2
(adică $(d_1 \sqsubseteq d_2)$ și $(d_2 \sqsubseteq d_1)$ sunt adevărate)

$(c \rightarrow d)$ spune că individul indicat de c satisface descrierea exprimată de conceptul d.

Interpretări în DL

O interpretare $\mathfrak{I}$ în DL este (ca și în FOL) o pereche $\langle D, I \rangle$, unde D este o mulțime nevidă de obiecte, numită domeniul interpretării, iar I este o mapare a interpretării ce asignează un înțeles simbolurilor nonlogice din DL, astfel încât:

1. pentru orice constantă c, $I[c] \in D$;
2. pentru orice concept atomic a, $I[a] \subseteq D$;
3. pentru orice rol r, $I[r] \subseteq D \times D$.

Mulțimea $I[d]$ se numește extensie dacă d este un concept:

- $I[\text{Thing}] = D$;
- $I[[\text{ALL } r\ d]] = \{x \in D \mid \forall y$ cu $\langle x, y \rangle \in I[r]$, atunci $y \in I[d]\}$
- $I[[\text{EXISTS } n\ r]] = \{x \in D \mid$ există cel puțin n indivizi y a.î. $\langle x, y \rangle \in I[r]\}$;
- $I[[\text{FILLS } r\ c]] = \{x \in D \mid \langle x, I[c] \rangle \in I[r]\}$;
- $I[[\text{AND } d_1 \ldots d_n]] = I[d_1] \cap \ldots \cap I[d_n]$.

Adevărul într-o interpretare

Propoziția $(c \rightarrow d)$ este adevărată în $\mathcal{J}$ dacă obiectul denotat de c este în extensia lui d;
Propoziția $(d \sqsubseteq d')$ este adevărată dacă extensia lui d este o submulțime a extensiei lui d';
Propoziția $(d = d')$ este adevărată dacă extensia lui d este identică cu extensia lui d'.

$\mathcal{J} = \langle D, I \rangle$ propoziția α este adevărată în $\mathcal{J}$, și scriem $\mathcal{J} \models \alpha$, dc

$\mathcal{J} \models (c \rightarrow d)$ dnd $I[c] \in I[d]$;
$\mathcal{J} \models (d \sqsubseteq d')$ dnd $I[d] \subseteq I[d']$;
$\mathcal{J} \models (d = d')$ dnd $I[d] = I[d']$.

d, d' concepte
c constantă

Implicația logică

Fie S o mulțime de propoziții în DL și α o propoziție.
S implică logic α $(S \models \alpha)$ dnd pentru orice interpretare $\mathcal{J}$ în care $\mathcal{J} \models S$, avem $\mathcal{J} \models \alpha$.
Spunem că propoziția α este validă logic $(\models \alpha)$ dacă este dedusă logic de mulțimea vidă.
În DL sunt două tipuri de bază de raționament: dacă o constantă c satisface sau nu un concept d; dacă un concept d este inclus într-un alt concept d'.

$KB \models (c \rightarrow d)$ $\qquad$ $KB \models (d \sqsubseteq d')$

Propoziția ([AND Doctor Female] $\sqsubseteq$ Doctor) este validă - indiferent de extensiile asignate de o interpretare $\mathcal{J}$ lui Doctor și Female, extensia conceptului AND este o submulțime a extensiei lui Doctor.
Propoziția (john $\rightarrow$ Thing) este validă - indiferent de I[john], aceasta aparține lui D, care este extensia lui Thing.

În cazurile tipice, implicațiile depind de propoziții din KB. De exemplu dacă în KB avem propoziția (Surgeon $\sqsubseteq$ Doctor) atunci avem implicația
KB $\models$ ([AND Surgeon Female] $\sqsubseteq$ Doctor).
Ajungem la aceeași concluzie și dacă în loc de (Surgeon $\sqsubseteq$ Doctor) avem
(Surgeon $=$ [AND Doctor [FILLS :Specialty surgery]]).
Dacă KB ar fi vidă, nu am avea relația de subsumare
KB $\models$ ([AND Surgeon Female] $\sqsubseteq$ Doctor) deoarece putem alege o interpretare $\mathcal{I}$ în care conceptul AND să nu fie submulțime a extensiei lui Doctor (de exemplu, dacă $\mathcal{I}$[Doctor] $= \emptyset$ iar $\mathcal{I}$[Surgeon] $=$ $\mathcal{I}$[Female] $= \{1, 2, 3\}$).

Determinarea implicațiilor

Dată o bază de cunoștințe KB, dorim să determinăm dacă KB $\models \alpha$ pentru α de forma:
– $(c \rightarrow d)$ unde c constantă și d concept
– $(d \sqsubseteq e)$ unde d, e concepte
[KB $\models (d = e)$ dnd KB $\models (d \sqsubseteq e)$ și KB $\models (e \sqsubseteq d)$]

Simplificarea KB

Se poate demonstra că implicația de tip subsumare nu este afectată de prezența propozițiilor de forma $(c \rightarrow d)$ din KB.
Adică KB $\models (d \sqsubseteq e)$ dnd KB' $\models (d \sqsubseteq e)$, unde KB' este KB din care se elimină toate propozițiile de forma $(c \rightarrow d)$.
Deci pentru întrebări de tip subsumare, presupunem că KB nu conține propoziții $(c \rightarrow d)$.

Mai mult, putem înlocui propozițiile de tipul $(d \sqsubseteq e)$ din KB cu propoziții de forma ($d =$ [AND e a]), unde a este un concept atomic nou nefolosit în altă parte.

Se consideră următoarele restricții în KB:

- partea stângă a conectorului = este un concept atomic, altul decât Thing
- fiecare atom apare în partea stângă a unei propoziții o dată (exac
  (astfel de propoziții definesc conceptele atomice)
- propozițiile de tipul = sunt aciclice. Adică excludem o KB care conține $\{(d_1 = [AND\ d_2 \dots]), (d_2 = [ALL\ r\ d_3]), (d_3 = [AND\ d_1 \dots$

Cu aceste restricții, pentru a determina dacă $KB \models (d \sqsubseteq e)$ vom parcurge următorii pași:

1. folosind definițiile (=) din KB, scriem d și e într-o formă normalizată specială;
2. determină dacă fiecare parte a conceptului e normalizat este reprezentată de o parte a conceptului normalizat d. Căutăm de fapt o relație structurală între două concepte normalizate.
   De exemplu, dacă e conține componenta $[ALL\ r\ e']$ atunci d trebuie să conțină un $[ALL\ r\ d']$, cu $d' \sqsubseteq e'$.

Normalizarea

Această preprocesare simplifică pasul de potrivire a structurilor conceptelor.

Normalizarea se aplică pe rând câte unui concept și se face pe pași:

1. extinderea definițiilor – orice concept atomic ce apare în stânga conectorului = este înlocuit de definiția lui.

   (Surgeon = [AND Doctor [FILLS :Specialty surgery]]) în KB

   conceptul [AND ... Surgeon ...] se extinde la

   [AND ... [AND Doctor [FILLS :Specialty surgery]] ...]

2. nivelarea operatorilor AND

   $[AND \dots [AND\ d_1 \dots d_n] \dots]$ devine

   $[AND \dots d_1 \dots d_n \dots]$

3. combinarea operatorilor ALL

[AND ... [ALL r d_1] ... [ALL r d_2] ...] devine

[AND ... [ALL r [AND d_1 d_2]] ...].

4. combinarea operatorilor EXISTS

[AND ... [EXISTS n_1 r] ... [EXISTS n_2 r] ...] devine

[AND ... [EXISTS n r] ...] unde $n = \max(n_1, n_2)$.

5. Thing – în anumite situații eliminat ca argument al conceptului AND: Thing, [ALL r Thing] și AND fără argumente.

6. eliminarea expresiilor ce sunt duplicate ale altor expresii în cadrul aceluiași concept AND

Acești pași se aplică în mod repetat în orice ordine și la orice nivel al operatorilor ALL și AND. Normalizarea se încheie când nu mai este aplicabil niciun pas.

AND fără componente AND cu o componentă

Rezultatul normalizării poate fi Thing sau un concept atomic sau un concept de forma

[AND a_1 ... a_m
[FILLS r_1 c_1] ... [FILLS $r_{m'}$ $c_{m'}$]
[EXISTS n_1 s_1] ... [EXISTS $n_{m''}$ $s_{m''}$]
[ALL t_1 e_1] ... [ALL $t_{m'''}$ $e_{m'''}$]]

unde $a_1 \ldots a_m$ sunt concepte atomice (altele decât Thing), r_i, s_i, t_i sunt roluri, c_i constante, n_i întregi pozitivi și e_i sunt concepte normalizate.

Exemplul 1 Considerăm următoarea KB:

WellRoundedCo = [AND Company [ALL :Manager [AND B-School Grad
[EXISTS 1 :TechnicalDegree]]]]

HighTechCo = [AND Company [FILLS :Exchange nasdaq]
[ALL :Manager Techie]]

Techie = [EXISTS 2 :TechnicalDegree]

Să se normalizeze conceptul [AND WellRoundedCo HighTechCo]

$\xrightarrow{1}$ [AND [AND Company [ALL :Manager [AND B-SchoolGrad
[EXISTS 1 :TechnicalDegree]]]
2
3
[AND Company [FILLS :Exchange nasdaq]
[ALL :Manager [EXISTS 2 :TechnicalDegree]]]

$\xrightarrow{2,3}$ [AND Company [ALL :Manager [AND [AND B-SchoolGrad
2
[EXISTS 1 :TechnicalDegree]]
4
[EXISTS 2 :TechnicalDegree]]]
6
Company
[FILLS :Exchange nasdaq]]

$\xrightarrow{2,4,6}$ [AND Company
[ALL :Manager [AND B-SchoolGrad [EXISTS 2 :TechnicalDegree]]]
[FILLS :Exchange nasdaq]]

<u>Procedura de verificare a relației structurale</u>

INPUT: d și e concepte normalizate
d este [AND $d_1 \dots d_m$]
e este [AND $e_1 \dots e_{m'}$]

OUTPUT: DA sau NU după cum $KB \models (d \sqsubseteq e)$ sau nu.

Se returnează DA dnd pentru orice componentă e_j, $1 \le j \le m'$, există o componentă d_i, $1 \le i \le m$ a.î. d_i se potrivește cu e_j după cum urmează:

1. dacă e_j este un concept atomic, d_i trebuie să fie identic cu e_j;
2. dacă e_j este de forma [FILLS r c], d_i trebuie să fie identic cu e_j
3. dacă e_j este de forma [EXISTS n r], d_i trebuie să fie de forma [EXISTS n' r] cu $n' \ge n$; dacă $n = 1$ d_i poate să fie și de forma [FILLS r c] pentru orice constantă c;
4. dacă e_j este de forma [ALL r e'], d_i trebuie să fie de forma [ALL r d'] cu $d' \sqsubseteq e'$.

Exemplul 2

d [AND Company
   [ALL :Manager B-SchoolGrad]
   [EXISTS 1 :Exchange]]

1 4

d' [AND Company
   [ALL :Manager [AND B-SchoolGrad [EXISTS 2 :TechnicalDegree]]]
3 [FILLS :Exchange nasdaq]]

Avem deci $d' \sqsubseteq d$.

Satisfacerea unui concept

Ne interesează dacă $KB \models (b \to e)$, unde b constantă și e concept

Dacă KB conține $(b \to d)$ și $KB \models (d \sqsubseteq e)$ atunci $KB \models (b \to e)$

Am putea colecta toate propozițiile de forma $(b \to d_i)$ din KB și returna DA (adică $KB \models (b \to e)$) dacă conceptul $[AND\ d_1 \dots d_n] \sqsubseteq e$.

Dar în felul acesta putem pierde inferențe necesare. De exemplu, dacă KB conține:

joe → Person
canCorp → [AND Company
   [ALL :Manager Canadian]
   [FILLS :Manager joe]]

putem deduce $KB \models$ (joe → Canadian)

In general, pentru a vedea dacă un individ satisface o descriere, trebuie să propagăm ce știm despre alți indivizi înainte de a verifica relația de subsumare.

Acest lucru se poate face printr-o procedură de tip "înlănțuire înainte".

Dacă presupunem că nu există termeni EXISTS în niciun concept din KB, procedura este:

1. fie S lista de perechi (b,d), b constantă din KB și d este versiunea normalizată a conceptului $[AND\ d_1' \ldots d_n']$ cu $(b \to d_i')$ în KB.

2. găsește două constante b_1 și b_2 a.î. $(b_1, d_1) \in S$, $(b_2, d_2) \in S$, $[FILLS\ r\ b_2]$ și $[ALL\ r\ e]$ sunt părți ale lui d_1, dar $KB \not\models (d_2 \sqsubseteq e$

3. dacă nu există b_1 și b_2, exit. Altfel, înlocuiește (b_2, d_2) din S cu (b_2, d_2') unde d_2' este normalizarea conceptului $[AND\ d_2\ e]$ și mergi la pasul 2.

Procedura calculează cel mai specific concept d a.î. $KB \models (b \to d)$, pentru b constantă din KB.
Acum, pentru a testa dacă $KB \models (b \to e)$ trebuie să testăm dacă $KB \models (d \sqsubseteq e)$.

În cazul în care avem termeni de forma $[EXISTS\ 1\ r]$, vom folosi lanțurile de roluri

$[AND \ldots [ALL\ r_1 \ldots [AND \ldots [ALL\ r_k\ a] \ldots] \ldots] \ldots]$

$\tau = r_1 \cdot r_2 \cdots r_k$ s.n. lanț de roluri

Dacă b constantă, r_1, r_2 roluri $b \cdot r_1 \cdot r_2$ reprezintă un individ (nenumit) ce se află în relație r_2 cu un individ ce se află în relație r_1 cu b.
Dacă τ este vid, $b \cdot \tau$ este b.
Procedura se extinde acum cu pași adiționali:
- găsește o constantă b, un lanț de roluri τ (posibil vid) și un rol r astfel încât $(b \cdot \tau, d_1) \in S$ și $(b \cdot \tau \cdot r, d_2) \in S$ (dacă nu există atunci d_2 se ia Thing), unde $[EXISTS\ 1\ r]$ și $[ALL\ r\ e]$ sunt componente ale conceptului d_1, dar $KB \not\models (d_2 \sqsubseteq e)$.
- dacă se găsește $(b \cdot \tau, d_1)$, dacă $(b \cdot \tau \cdot r, d_2) \in S$ atunci se înlocuiește cu $(b \cdot \tau \cdot r, d_2')$, unde d_2' este normalizarea conceptu[lui] $[AND\ d_2\ e]$. Dacă $(b \cdot \tau \cdot r, d_2) \notin S$ atunci se adaugă $(b \cdot \tau \cdot r, d_2'$
Repetă.

Se pornește cu o proprietate a individului b·τ și se deduce ceva nou în legătură cu individul (anonim) b·τ·r. Într-un final această informație nouă poate duce la obținerea unei informații noi cu privire la un individ cu nume.

De exemplu, dacă în KB avem:

$$\overset{b\cdot\tau}{ellen} \rightarrow \overbrace{[\text{AND } [\text{EXISTS } 1 \ \overset{r}{:\text{Child}}] \ [\text{ALL } \overset{r}{:\text{Child}} \ \overbrace{[\text{AND } [\text{FILLS } :\text{Pediatrician } marianne] \ [\text{ALL } :\text{Pediatrician Scandinavian}]]}^{e}]]}^{d_1}$$

$$(b\cdot\tau\cdot r, d_2) = (ellen{:}\text{Child}, \text{Thing})$$

$$KB \not\models (\text{Thing} \sqsubseteq e)$$

în S se va adăuga $(ellen{:}\text{Child}, [\text{AND } [\text{FILLS } :\text{Pediatrician } marianne] \ [\text{ALL } :\text{Pediatrician Scandinavian}]])$

De aici se deduce (marianne → Scandinavian) (cazul fără EXISTS).

Cazul termenilor [EXISTS n r], n>1 se tratează la fel ca pentru n=1. Nu este necesară crearea a n indivizi anonimi diferiți, deoarece toți ar "produce" aceleași proprietăți în întâlnirea înainte.