FIRST ORDER LOGIC (FOL)

6.1 Tujuan Instruksional

Mahasiswa dapat menjelaskan kelebihan FOL sebagai media untuk merepresentasikan pengetahuan.

Mahasiswa dapat mengkonversi bahasa natural ke dalam bentuk FOL.

Mahasiswa dapat menggunakan konsep-konsep tentang: sintaks, semantik, interpretasi, kuantifikator, entailment, unifikasi, dan skolemisasi.

Mahasiswa dapat menggunakan aturan inferensi dalam FOL.

Mahasiswa dapat memastikan tingkat validitas / penerimaan (soundness) dan kelengkapan (completeness) sebagai hasil inferensi dalam FOL.

Mahasiswa dapat menerapkan teknik resolusi dalam FOL.

Mahasiswa dapat menggunakan FOL dalam melakukan pencarian, termasuk dengan menggunakan heuristik.

6.2 First Order Logic

6.2.1 Sintaks

Pada bagian ini akan dibahas mengenai berbagai simbol dan alfabet yang digunakan dalam mendeskripsikan FOL secara umumnya. Perlu diingat bahwa dalam FOL dapat saja digunakan jenis simbol yang berbeda, tentu saja dengan syarat ada konsensus pemakaiannya.

6.2.1.1 Alfabet

- Simbol logika: simbol-simbol dengan arti standar dalam penalaran logika tetap berlaku dalam FOL, seperti AND, OR, NOT, IMPLIKASI, BI-IMPLIKASI. Namun dengan FOL dapat ditambahkan kuantifikator eksistensial dan universal dalam penyusunan kalimat logika. Sehingga dapat dimodelkan relasi antar obyeknya.
- Simbol non-logika, dapat terdiri atas:
    - Konstanta / Obyek
        - Predikat: menggambarkan relasi (biasanya teridentifikasi dengan kata kerja). Dapat bernilai benar / salah.
        - Fungsi: predikat yang memberikan suatu hasil tertentu dari relasi yang terjadi.
    - Variabel: merupakan sebuah identifikator, biasanya dilambangkan dengan huruf kecil, a … z, untuk melambangkan sebuah obyek yang belum merupakan sebuah konstanta.

Konvensi yang biasa digunakan untuk menuliskan predikat, fungsi ataupun variabel:

- Predikat, dituliskan dengan awal huruf kapital, dan biasanya merupakan sebuah kata kerja.
- Fungsi, dituliskan dengan huruf kapitas semua, dan biasanya merupakan sebuah kata kerja.
- Variabel, dituliskan dengan sebuah huruf kecil, dan digunakan untuk menggambarkan sebuah yang belum pasti, dalam suatu domain.

6.2.1.2 Terms

Sebuah term adalah sebuah konstanta atau variabel atau fungsi yang berisi term di dalamnya, dituliskan F(t1, t2,…)

6.2.1.3 Formula Atomik

Formula atomik bernilai awal default "FALSE", atau jika telah memiliki relasi antar term-nya, akan menggunakan predikat untuk menyatakan relasi tersebut, dituliskan dengan P(t1, t2, …).

6.2.1.4 Literal

Sebuah literal adalah sebuah formula atomik (literal positif) atau negasinya (literal negatif). Literal yang tidak memiliki variabel, disebut dengan Ground Literal.

6.2.1.5 Klausa

Sebuah klausa adalah gabungan (dengan operator OR / disjungsi) antara beberapa literal. Sebuah Klausa HORN, adalah klausa yang memiliki paling banyak satu buah literal positif. Klausa Definite, adalah klausa HORN yang persis memiliki satu literal dan berupa literal positif. Perlu diperhatikan bahwa relasi dalam sebuah klausa HORN akan bernilai benar secara implikasi.

$A \rightarrow B$, akan bernilai sama dengan $\neg A \vee B$.

$A \wedge B \rightarrow False$, akan bernilai sama dengan $\neg A \vee \neg B$.

6.2.1.6 Formula

Sebuah formula adalah jika memenuhi salah satu dari:

- Formula atomik
- Memiliki relasi negasi, konjungsi, disjungsi, implikasi, dan bi-implikasi dengan formula atomik lainnya.
- Menggunakan kuantifikator universal ($\forall$, universally quantified formula) atau eksistensial ($\exists$, existentially quantified formula).

Jika ada variabel dalam sebuah formula, namun term dalam formula tidak dikaitkan dengan variabel tersebut, maka akan disebut dengan "formula bebas". Jika semua variabel terikat di dalam formula, maka formula akan disebut sebagai formula tertutup (closed), dan jika tidak ada satupun variabel yang terikat dalam formula akan disebut sebagai formula terbuka (opened).

Sebuah formula yang memiliki keterhubungan “disjungsi” antar klausanya, akan disebut sebagai “clausal form formula”. Perlu diperhatikan bahwa setiap formula ekuivalen dengan bentuk klausalnya, ingat bahwa relasi OR, hanya memerlukan satu kondisi benar.

6.2.2 Substitusi

- Diberikan sebuah term “s”, maka hasil substitusi variabel x dalam term lain di dalam s, yang diberi nama “t”, s[t/x], akan memberi hasil:
  - t, jika x adalah variabel dari x.
  - y, jika y adalah variabel selain x.
  - F(s1[t/x] s2[t/x] … sn[t/x]), jika s adalah F(s1, s2, …, s3).
- Diberikan sebuah formula A, hasil dari substitusi sebuah term “t”, di dalam formula A untuk variabel x, A[t/x], adalah:
  - FALSE, jika A FALSE.
  - P(t1[t/x], t2[t/x], …, tn[t/x]), jika A adalah P(t1, t2, …, tn).
  - (B[t/x] AND C[t/x]), jika A adalah B AND C, demikian pula dengan konektor lainnya.
  - ($\forall$x B), jika A adalah $\forall$x B, demikian pula untuk kuantifikator $\exists$.
  - ($\forall$y B[t/x]), jika A adalah $\forall$y B, dan y berbeda dengan x. Demikian pula den kuantifikator $\exists$.

Subtitusi [t/x] dapat dianggap sebagai sebuah pemetanaan dari satu term ke term lainnya dan dari satu formula ke formula lainnya. Dapat pula didefinisikan kesetaraan (similarity) [t1/x1, t2/x2, …, tn/xn], dimana t1, t2, …, tn adalah term, dan x1, x2, …, xn adalah variabel, sebagai pemetaan dengan subtitusi yang dilakukan secara simultan untuk x1 oleh t1, x2 oleh t2, … , atau xn oleh tn. Jika term t1, … , tn adalah variabel, maka substitusi yang dilakukan disebut sebagai varian alfabet, dan jika t1, …, tn adalah konstanta, maka substitusinya akan disebut sebagai substitusi bebas (ground substitution).

6.2.3 Unifikasi

- Diberikan dua buah substitusi S = [t1/x1, …, tn/xn] dan V = [u1/y1, …, um/ym], maka komposisi untuk S dan V, disimbolkan dengan S*V, adalah substitusi yang diperoleh dengan:
  - Menerapkan V ke dalam t1, …, tn. Substitusi dengan properti seperti ini disebut sebagai konkatenasi, dan
  - Menambahkan pasangan uj/yj, dengan yj tidak berada dalam himpunan {x1, …, xn}.

Sebagai contoh: [G(x,y)/z * [A/x, B/y, C/w, D/z] akan memberikan hasil [G(A, B)/z, A/x, B/y, C/w]. Dalam hal ini D/z sudah disubstitusikan juga dalam B(A, B). Komposisi adalah suatu operasi yang bersifat asosiatif namun tidak komutatif.

- Sebuah himpunan dalam bentuk f1, …, fn akan dapat diunifikasikan, jika ada sebuah substitusi S sehingga f1*S = f2*S = ... = fn*S. Dengan demikian S adalah sumber unifikasi untuk himpunan tersebut. Sebagai contoh {P(x, F(y), B), P(x, F(B), B)} dapat diunifikasikan dengan [A/x B/y] dan juga dengan [B/y].

- Most General Unifier (MGU) adalah sebuah himpunan dengan bentuk f1, …, fn; yang adalah sebuah substitusi dalam S yang mengunifikasikan himpunan ini sehingga dapat disubstitusikan pula dengan T, yang mengunifikasikan himpunan ini dengan substitusi V, sehingga S*V = T. Hasil dari penerapan MGU ke dalam bentuk f1, …, fn, disebut sebagai Most General Instance (MGI).

  Berikut adalah beberap contoh:

FORMULAE	MGU	MGI
(P x), (P A)	[A/x]	(P A)
(P (F x) y (G y)), (P (F x) z (G x))	[x/y x/z]	(P (F x) x (G x))
(F x (G y)), (F (G u) (G z))	[(G u)/x y/z]	(F (G u) (G y))
(F x (G y)), (F (G u) (H z))	Not Unifiable	
(F x (G x) x), (F (G u) (G (G z)) z)	Not Unifiable	

Contoh terakhir menarik untuk dianalisis: terlihat pada awalnya bahwa (G u) harus mengsubstitusikan x, kemudian (G z) juga harus mengsubstitusikan x. Dan pada akhirnya terlihat bahwa x dan z adalah ekuivalen. Maka akan diperlukan x → (G z) dan x → z yang bernilai benar untuk kedua hal tersebut. Dan ini hanya akan bernilai benar jika z dan (G z) ekuivalen. Hal ini tidak akan pernah terjadi jika term terbatas. Untuk mengenali kasus seperti ini, yang tidak memungkinkan adanya unifikasi, yaitu z tidak dapat digantikan dengan (G z), karena z juga terjadi di dalam (G z), maka diperlukan adanya Uji Keberadaan (Occur Test). Kebanyak implementasi bahasa pemrograman Prolog menerapkan unifikasi, namun masih belum menerapkan Uji Keberadaan, karena alasan efisiensi.

Proses terjadinya Most General Unifier, dapat digambarkan melalui algoritma Unifikasi, dengan pseudocode seperti berikut:

```
FUNCTION Unify WITH PARAMETERS form1, form2, and assign RETURNS MGU,
where form1 and form2 are the forms that we want to unify, and assign
is initially nil.

1. Use the Find-Difference function described below to determine the
   first elements where form1 and form2 differ and one of the elements
   is a variable. Call difference-set the value returned by Find-
   Difference. This value will be either the atom Fail, if the two
   forms cannot be unified; or null, if the two forms are identical; or
   a pair of the form (Variable Expression).

2. If Find-Difference returned the atom Fail, Unify also returns Fail
   and we cannot unify the two forms.

3. If Find-Difference returned nil, then Unify will return assign as
   MGU.

4. Otherwise, we replace each occurrence of Variable by Expression in
   form1 and form2; we compose the given assignment assign with the
   assignment that maps Variable into Expression, and we repeat the
   process for the new form1, form2, and assign.
```

```
FUNCTION Find-Difference WITH PARAMETERS form1 and form2 RETURNS pair,
where form1 and form2 are e-expressions.

1. If form1 and form2 are the same variable, return nil.

2. Otherwise, if either form1 or form2 is a variable, and it does not
   appear anywhere in the other form, then return the pair (Variable
   Other-Form), otherwise return Fail.

3. Otherwise, if either form1 or form2 is an atom then if they are the
   same atom then return nil otherwise return Fail.

4. Otherwise both form1 and form2 are lists.
   Apply the Find-Difference function to corresponding elements of the
   two lists until either a call returns a non-null value or the two
   lists are simultaneously exhausted, or some elements are left over
   in one list.
```

6.2.4 Semantik

Sebelum dibicarakan lebih lanjut mengenai penalaran dan pembuktian, perlu dibahas terlebih dahulu mengenai nilai / arti kalimat (semantik) dalam FOL.

Sebuah struktur atau konseptualisasi untuk sebuah bahasa L adalah M=(U,I), dimana:

- U adalah sebuah himpunan yang tidak kosong, disebut sebagai domain atau semesta pembicaraan (Universe of Discourse) bagi M.
- I adalah sebuah interpretasi setiap lambang dalam bahasa L dengan pemetaan: I(F) : U x U x … x U, dan setiap simbol predikat P dalam L, sebagai himpunan bagian dari U x U x U.

Himpunan fungsi atau predikat yang diperoleh dengan cara seperti di atas, disebut sebagai Dasar Penerapan(Functional Basis) atau Dasar Relasi (Relational Basis) terhadap konseptualisasi dalam domain yang berlaku.

Diberikan sebuah bahasa L, dan sebuah konseptualisasi (U,I), sebuah alokasi (assignment) adalah pemetaan sebuah variabel L ke dalam U. Sebuah variant X, dari alokasi s adalah alokasi yang identik untuk setiap s dalam semesta pembicaraan, kecuali pada tempat-tempat dimana x berbeda.

Diberikan sebuah konseptualisasi M=(U,I), dan sebuah alokasi s, maka akan mudah untuk memperluas pemetaan s di dalam U untuk setiap term t, s(t), dengan menggunakan induksi terhadap struktur di dalam term terkait.

Maka berlaku:

- M akan berlaku benar untuk formula A melalui pemetaan s, jika:
    - A adalah formula atomik, katakan P(t1, …, tn), dan (s(t1), …, s(tn)) berada di dalam interpretasi I(P).
    - A adalah (NOT B), dan konseptualisasi M tidak berlaku benar untuk B dalam pemetaan s.
    - A adalah (B OR C), dan M berlaku benar untuk s, atau M berlaku benar untuk C dalam s. Demikian pula dengan konektor lainnya.
    - A adalah ($\forall$x B), dan M berlaku benar untuk B pada semua varian x dalam s.
    - A adalah ($\exists$x B), dan M berlaku benar untuk B pada beberapa varian x dalam s.
- Formula A berlaku benar di dalam M, jika ada sebuah alokasi s sehingga M berlaku benar untuk A di dalam s.
- Formula A berlaku benar, jika ada sebuah struktur bahasa L untuk interpretasi M yang berlaku benar di dalam M.
- Formula A valid atau benar secara logika di dalam M, jika M berlaku benar untuk A dengan nilai apapun untuk s. M dalam keadaan seperti ini akan disebutkan sebagai model untuk A.
- Formula A valid atau benar secara logika, jika untuk seiap struktur bahasa L, interpretasi M, alokasi s, maka M berlaku benar untuk A dalam alokasi s.

Beberapa dari definisi di atas, dapat disejajarkan dengan himpunan formula GAMMA:

- Formula A disebut memiliki konsekuensi logis GAMMA di dalam interpretasi M, jika M berlaku benar untuk A terhadap semua alokasi s, yang juga berlaku benar untuk semua formula dalam GAMMA.

- Formula A disebut memiliki konsekuensi logis GAMMA jika untuk setiap struktur bahasa L, interpretasi M, A adalah sebuah konsekuensi logis untuk GAMMA di dalam M. Pada saat “A adalah sebuah konsekuensi logis untuk GAMMA”, maka disebut pula “GAMMA entails A”.

Formula A dan B disebut ekuivalen secara logis, jika A adalah sebuah konsekuensi logis untuk B, dan B adalah konsekuensi logis untuk A.

Contoh “DUNIA BALOK”.

Dalam contoh berikut akan didefinisikan sebuah bahasa L untuk sebuah domain pembicaraan. Akan dibahas mengenai sebuah semesta pembicaraan mengenai BALOK, mengikuti gambar berikut ini:

```
                                      +--+
                                      |a |
                                      +--+
                                      |e |
   +--+                               +--+
   |a |                               |c |
   +--+     +--+                      +--+
   |b |     |d |      ======>         |d |
   +--+     +--+                      +--+
   |c |     |e |                      |b |
 ---------------                --------------------
```

Dari gambar dapat dilihat dua keadaan yang berlaku benar. Sebelah kiri adalah keadaan saat ini, dan sebelah kanan adalah tujuan yang diinginkan. Sebuah robot akan melakukan tranformasi ini.

Untuk mendeskripsikan dunia ini, dapat digunakan sebuah struktur universe (domain), U = {a, b, c, d, e}, dan dengan himpunan predikat P = {ON, ABOVE, CLEAR, TABLE}, yang memiliki arti sebagai berikut:

- ON: (ON (x,y)), jika x tepat berada di atas y. Interpretasi untuk ON pada keadaan di sebelah kiri adalah {(a, b), (b, c), (d, e)} dan untuk keadaan di sebelah kanan adalah {(a, e), (e,c), (c, d), (d, b)}.

- ABOVE: (ABOVE(x, y)), jika x berada di atas y, tanpa harus langsung. Interpretasi untuk ABOVE, pada keadaan di sebelah kiri adalah {(a, b), (a, c), (b, c), (d, e)} dan untuk keadaan di sebelah kanan adalah {(a, e), (a, c), (a, d), (a, b), (e,c), (e, d), (e, b), (c, d), (c, b), (d, b)}.

- CLEAR: (CLEAR(x)), jika tidak ada sesuatu apapun di atas x. Interpretasi untuk CLEAR pada keadaan sebelah kiri adalah {(a), (d)}, dan di sebelah kanan adalah {(a)}.

- TABLE: (TABLE(x)), jika x tepat berada di atas meja. Interpretasi untuk TABLE pada keadaan di sebelah kiri adalah {(c), (e)}, dan di sebelah kanan adalah {(b)}.

Contoh formula yang bernilai benar sesuai dengan dunia BLOK di atas(untuk keadaan kiri dan kanan), adalah:

- ON(x, y) mengimplikasi ABOVE(x,y)

- (ON(x, y) AND ABOVE (y, z)) mengimplikasi ABOVE (x, z)

- ABOVE(x, y) mengimplikasi (NOT(ABOVE (y,x))

- CLEAR(x), if $\neg\exists$y (ON (y, x))

- TABLE(x), if $\neg\exists$y (ON (x, y))

Perlu diperhatikan bahwa ada hal-hal yang tidak dapat kita katakan tentang dunia BLOK, dengan hanya mengandalkan fungsionalitas dan predikat dasar, tanpa menggunakan ekualitas. Dalam hal ini, tidak dapat dikatakan bahwa sebuah balok berada di atas (ON) paling banyak satu balok lainnya. Untuk itu perlu dikatakan bahwa: "ON(x, y) AND ON(x,z) → ON (y,z)". Dengan demikian digunakanlah sebuah logika dengan ekualitas (kesamaan maksud).

Tidak semua formula yang benar dalam keadaan di sebelah kiri, juga berlaku benar untuk dunia di sebelah kanan, dan sebaliknya. Sebagai contoh, sebuah formula yang benar untuk keadaan di sebelah kiri, namun tidak benar untuk di sebelah kanan adalah ON(a, b). Pengambilan kesimpulan dengan hanya berdasarkan salah satu keadaan saja dapat menghasilkan kontradiksi. Sebagai contoh ABOVE(b, c) benar di kiri, namun ABOVE(c, b) benar di kanan, dan jika digabungkan akan menghasilkan kontradiksi mengenai posisi b dan c. Dengan demikian hal apapun yang dihasilkan dari penggambaran keadaan dalam dunia BLOK hanya akan berlaku benar, jika hanya satu bagian saja yang dibicarakan dalam semesta pembicaraan. Untuk dapat membicarakan mengenai dua dunia secara simultan (dengan kondisi yang berbeda), diperlukan penggambaran situasi yang lebih dalam, dikenal dengan istilah Situation Calculus.