First-Order Logic

Knowledge Interchange Format (KIF)

Professor Richard Fikes

CS222
Winter 2003

Computer Science Department
Stanford University

KR Language Components

◆ A logical formalism
  ▪ Syntax for wffs
  ▪ Vocabulary of logical symbols  (e.g., AND, OR, NOT, =>, <=>)
  ▪ Interpretation semantics for the logical symbols
    E.g., “(=> A B)” is true if and only if B is true or A is false.

◆ An ontology
  ▪ Vocabulary of non-logical symbols
    > Relations, functions, constants
  ▪ Definitions of non-primitive symbols
    E.g., (=> (Bachelor ?x) (AND (Man ?x) (Unmarried ?x)))
  ▪ Axioms restricting the interpretations of primitive symbols
    E.g., (=> (Person ?x) (Gender (Mother ?x) Female))

◆ A proof theory
  ▪ Specification of the reasoning steps that are logically sound
    E.g., From “(=> S1 S2)” and “S1”, conclude “S2”.
Knowledge Interchange Format

Interlingua for Multi-Use Knowledge

Language 1  Language 2  ...  Language n

K I F

KB Library

◆ Knowledge Interchange Format (KIF)
  ‣ First-order symbolic logic with equality
  ‣ Includes numbers, lists, and strings
  ‣ Linear ASCII syntax
  ‣ In the process of becoming an ISO standard

Conceptualization

◆ Set of objects about which knowledge is being expressed
  ‣ Objects can be –
    > Concrete        Clyde, my car
    > Abstract        Justice, 2
    > Primitive       Resister
    > Composite       Electric circuit
    > Fictional       Sherlock Holmes
  ‣ Objects always in the KIF conceptualization
    > Real and complex numbers
    > All finite lists of objects
    > Words
    > ASCII characters
    > Finite strings of ASCII characters

◆ Set of relations and functions on the objects
Conceptualization

- Set of objects about which knowledge is being expressed
- Set of relations and functions on the objects
  - Relation
    > Entity having an extension that is a set of finite sequences of objects
      E.g., Parent: {<Richard Earl> <Richard Polly> <Debbie Don> ... }
  - Relation \( R \) is functional on a sequence \( \langle O_1, ..., O_n \rangle \) just in case there is exactly one object \( V \) in the conceptualization such that \( \langle O_1, ..., O_n, V \rangle \) is in the extension of \( R \)
  - Function
    > Relation \( R \) such that for every sequence \( \langle O_1, ..., O_n, V \rangle \) in the extension of \( R \), \( R \) is functional in \( \langle O_1, ..., O_n \rangle \)
      E.g., \( +: \{\langle 1 \ 3 \ 4 \rangle \langle 17 \ 23 \ 40 \rangle \langle 2 \ 7 \ 10 \ 12 \ 31 \rangle \} \)
    > The last element of each sequence in the extension of a function is referred to as the value of that sequence

Blocks World

Objects - a, b, c, d, e, table
Blocks World

- **Objects**
  - a, b, c, d, e, table

- **Relations**
  - Above: \{〈a b〉 〈a c〉 〈b c〉 〈d e〉\}
  - Clear: \{〈a〉 〈d〉\}
  - Table: \{〈c〉 〈e〉\}

- **Functions**
  - On: \{〈a b〉 〈b c〉 〈d e〉\}

KIF Syntax

- **Knowledge Base** – Collection of sentences
- **Sentence** – Expression denoting a statement
- **Term** – Expression denoting an object
- **Word** – Letter or digit followed by any number of other legal word characters
  - **Object Variable**
    - Word beginning with “?”
      - E.g., ?x, ?The-First-Murderer
  - **Sequence Variable**
    - Word beginning with “@”
      - E.g., @x, @The-Other-Murderers
  - **Sentence Operator**
    - not, and, or, implies, iff, for all, exists
  - **Constant**
    - All other words
      - E.g., Fred, Block-A, Justice
Term Syntax

- **Constants**
  E.g., Joe, 2

- **Object variables**
  E.g., ?x

- **Function Terms**
  \( (\text{function constant} \ <\text{term}>^* \ [\text{sequence variable}]) \)
  E.g., (plus 2 3) (Father-Of Richard) (plus 4 ?x @Other-Addends)

  Denotes the object denoted by the “value” of the function with the given arguments

Sentence Syntax

- **Atomic Sentences**
  \( (\text{relation constant} \ <\text{term}>^* \ [\text{sequence variable}]) \)
  E.g, (Parent Richard Earl) (Clear A) (Set-Partition Set1 @Sets)

  Equations
  > \( (= <\text{term} > <\text{term}>) \)
  E.g, (= (Father Richard) Earl) (= A B)

- **Logical Sentences**
  \( (\text{not} <\text{sentences}>) \)
  \( (\text{and} <\text{sentence}>^*) \)
  \( (\text{or} <\text{sentence}>^*) \)
  \( (\Rightarrow <\text{sentence}> <\text{sentence}>) \)
  \( (\Leftrightarrow <\text{sentence}> <\text{sentence}>) \)

- **Quantified Sentences**
  \( (\text{forall} <\text{variable}>^*) <\text{sentence}> \)
  \( (\text{exists} <\text{variable}>^*) <\text{sentence}> \)
Interpretations

◆ An Interpretation consists of:
  ◆ Universe of discourse (O)
    > The objects and relations in a conceptualization
  ◆ Extension function (ext)
    > Maps a relation into a set of sequences of objects in O
  ◆ Semantic value function (σ)
    > Maps a term to the object in O it denotes
  ◆ Truth value function (τ)
    > Maps a sentence to either “True” or “False

Semantic Value of a Term

◆ Semantic value of a constant
  ◆ σ(<constant>) = <object or relation in O>

◆ Semantic value of a variable
  ◆ σ(<object variable>) = <object in O>
  ◆ σ(<sequence variable>) = (<object in O>)

◆ Semantic value of a function term
  ◆ σ((fn term1 ... term_n)) =
    If σ(fn) is functional in ⟨σ(term1) ... σ(term_n)⟩ –
      Then the object V such that ⟨σ(term1) ... σ(term_n) V⟩ is in set ext(σ(fn))
      Else σ(fn)
  ◆ σ((fn term1 ... term_n @var)) =
    If σ(fn) is functional in ⟨σ(term1) ... σ(term_n) | σ(@var)⟩ –
      Then the object V such that ⟨σ(term1) ... σ(term_n) | σ(@var) V⟩ is in set ext(σ(fn))
      Else σ(fn)
Truth Value of Atomic Sentences

- **Relational sentences**
  \[ \tau((\text{rel} \ \text{term}_1 \ldots \text{term}_n)) = \]
  > true when \( \langle \sigma(\text{term}_1), \ldots, \sigma(\text{term}_n) \rangle \) is a member of set \( \text{ext}(\sigma(\text{rel})) \)
  > false otherwise

  \[ \tau((\text{rel} \ \text{term}_1 \ldots \text{term}_n \ \text{@var})) = \]
  > true when \( \langle \sigma(\text{term}_1), \ldots, \sigma(\text{term}_n), \sigma(\text{@var}) \rangle \) is a member of set \( \text{ext}(\sigma(\text{rel})) \)
  > false otherwise

- **Equations**
  \[ \tau(= \ \text{term}_1 \ \text{term}_2) = \]
  > true when \( \sigma(\text{term}_1) \) and \( \sigma(\text{term}_2) \) are the same object
  > false otherwise

Truth Value of Logical Sentences

- **Negations**
  \[ \tau(\text{not} \ \text{sent}) = \]
  > true when \( \tau(\text{sent}) \) is false
  > false otherwise

- **Conjunctions**
  \[ \tau(\text{and} \ \text{sent}_1 \ldots \text{sent}_n) = \]
  > true when \( \tau(\text{sent}_i) \) is true for all \( i = 1, \ldots, n \)
  > false otherwise

- **Disjunctions**
  \[ \tau(\text{or} \ \text{sent}_1 \ldots \text{sent}_n) = \]
  > true when \( \tau(\text{sent}_i) \) is true for some \( i = 1, \ldots, n \)
  > false otherwise
Truth Value of Logical Sentences

◆ Implications
  ‣ \( \tau((\implies \text{antecedent consequent})) = \)
    \> true when \( \tau(\text{antecedent}) \) is false or when \( \tau(\text{consequent}) \) is true
    \> false otherwise
  ‣ Note: \( \tau((\implies a c)) = \tau((\text{or} \ (\neg a) \ c)) \)

◆ Logical equivalences
  ‣ \( \tau((\iff \text{sent}_1 \text{ sent}_2)) \) =
    \> true when \( \tau(\text{sent}_1) \) is the same as \( \tau(\text{sent}_2) \)
    \> false otherwise
  ‣ Note: \( \tau((\iff s_1 \text{ sent}_2)) = \tau((\text{and} \ (\implies s_1 \text{ sent}_2) \ (\implies s_2 \text{ sent}_1))) \)

Truth Value of Quantified Sentences

◆ I-variant interpretations
  ‣ Let \( I = \langle O, \text{ext}, \sigma, \tau \rangle \) be an interpretation and each of \( v_1,\ldots,v_n \) be variables.
  ‣ An interpretation \( I' = \langle O, \text{ext}, \sigma', \tau \rangle \) is an I-variant on \( v_1,\ldots,v_n \) if \( \sigma' \) differs from \( \sigma \) at most in what it assigns to one or more of \( v_1,\ldots,v_n \).

◆ Existentially quantified sentences
  ‣ \( \tau((\exists \text{ (} v_1,\ldots,v_n \text{) sent})) = \)
    \> true when \( \tau(\text{sent}) \) is true under some I-variant on \( v_1,\ldots,v_n \)
    \> false otherwise

◆ Universally quantified sentences
  ‣ \( \tau((\forall \text{ (} v_1,\ldots,v_n \text{) sent})) = \)
    \> true when \( \tau(\text{sent}) \) is true under every I-variants on \( v_1,\ldots,v_n \)
    \> false otherwise
Digital Circuit $C_1$

Russell and Norvig, Figure 8.1

Domain Conceptualization

- **Objects**
  - Circuits
  - Signals
  - Gate types
  - Terminals
  - Gates
  - Signal values

- **Relations**
  - Connected: $\langle$terminal$\rangle$ $\langle$terminal$\rangle$
  - Terminal: $\langle$terminal$\rangle$
  - ...

- **Functions**
  - Type: $\langle$gate$\rangle$ $\rightarrow$ $\langle$gate type$\rangle$
  - In: $\langle$index$\rangle$ $\langle$gate$\rangle$ $\rightarrow$ $\langle$input terminal$\rangle$
  - Out: $\langle$index$\rangle$ $\langle$gate$\rangle$ $\rightarrow$ $\langle$output terminal$\rangle$
  - Signal: $\langle$terminal$\rangle$ $\rightarrow$ $\langle$signal value$\rangle$
Electronic Circuit Domain Theory

- Connected terminals have the same signal
  
  ```
  (=> (Connected ?t1 ?t2)
      (and (Terminal ?t1) (Terminal ?t2) (= (Signal ?t1) (Signal ?t2))))
  ```

- Relation “Connected” is commutative
  
  ```
  (<=> (Connected ?t1 ?t2) (Connected ?t2 ?t1))
  ```

- The signal at a terminal is either on or off
  
  ```
  (<=> (Terminal ?t) (or (Signal ?t On) (Signal ?t Off)))
  ```

- A gate has at least 1 input terminal and 1 output terminal
  
  ```
  (=> (Gate ?g) (and (Terminal (In ?g 1)) (Terminal (Out ?g 1))))
  ```

OR and AND Gates

- OR gate’s output is off when both of its inputs are off
  
  ```
  (=> (Type ?g OR)
      (and (Gate ?g)
          (<=> (Signal (Out 1 ?g) Off)
          (and (Signal (In 1 ?g) Off) (Signal (In 2 ?g) Off))))))
  ```

- AND gate’s output is on when both of its inputs are on
  
  ```
  (=> (Type ?g AND)
      (and (Gate ?g)
          (<=> (Signal (Out 1 ?g) On)
          (and (Signal (In 1 ?g) On) (Signal (In 2 ?g) On))))))
  ```
XOR and NOT Gates

- **XOR gate's output is on when its inputs are different**

  => (Type ?g XOR)
  (and (Gate ?g)
   (Terminal (In 2 ?g))
   (<=> (Signal (Out 1 ?g) On)
    (not (= (Signal (In 1 ?g)) (Signal (In 2 ?g))))))

- **NOT gate's output is different from its inputs**

  => (Type ?g NOT)
  (and (Gate ?g)
   (not (Signal (Out 1 ?g) (Signal (In 1 ?g))))))

Circuit C₁ Representation

- **Gates**
  (Type X₁ XOR)  (Type X₂ XOR)
  (Type A₁ AND)  (Type A₂ AND)
  (Type O₁ OR)

- **Connections**
  (Connected (Out 1 X₁) (In 1 X₂))  (Connected (In 1 C₁) (In 1 X₁))
  (Connected (Out 1 X₁) (In 2 A₂))  (Connected (In 1 C₁) (In 1 A₁))
  (Connected (Out 1 A₂) (In 1 O₁))  (Connected (In 2 C₁) (In 2 X₁))
  (Connected (Out 1 A₁) (In 2 O₂))  (Connected (In 2 C₁) (In 2 A₁))
  (Connected (Out 1 X₂) (Out 1 C₁))  (Connected (In 3 C₁) (In 2 X₂))
  (Connected (Out 1 O₁) (Out 2 C₁))  (Connected (In 3 C₁) (In 1 A₂))
Family Relationships

- **One’s uncle is** –
  - The brother of one’s father or mother, or
  - The husband of one’s aunt
- \( (\iff \text{uncle ?y}) \ (\exists ?x \ (\text{uncle-of ?y ?x})) \)
- \( (\iff \text{has-uncle ?x ?y}) \ (\text{uncle-of ?y ?x})) \)
- \( (\iff \text{uncle-of ?x ?y}) \)
  - \( (\exists ?z \ (\or \ (\and \ (\or \ (\text{mother-of ?z ?y}) \ (\text{father-of ?z ?y})) \ (\text{brother-of ?x ?z})) \ (\and \ (\text{aunt-of ?z ?y}) \ (\text{husband-of ?x ?z}))) \)

- **A widow is a woman who has lost her husband by death and has not remarried**
- **???