KNOWLEDGE REPRESENTATION

Unit

Outline

- Knowledge Representation
  - Knowledge Based Agents
  - Formal logic
  - Connectives
  - Truth tables
  - Syntax
  - Semantics
  - Tautology
  - Knowledge Models
  - Validity
  - Well Formed Formula
- Propositional Logic
- Predicate Logic
  - FOPL
  - Interpretation
  - Quantification
  - Horn Clauses

Outline

- Inference
  - Rules of Inference
  - Unification
  - Resolution Refutation System
  - Answer Extraction from RRS
  - Rule based Deduction System
- **Statistical Reasoning**
  - **Probability and Bayes Theorem**
  - **Causal Networks**
  - **Reasoning in Belief Network**

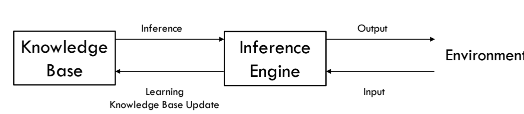
Knowledge Representation

- An area of AI whose fundamental goal is to represent knowledge in a manner that facilitates inferring or drawing conclusion from knowledge
- Analyses how to think formally, how to use symbol to represent a domain of discourse along with the function that allow inference about the objects

Knowledge Representation

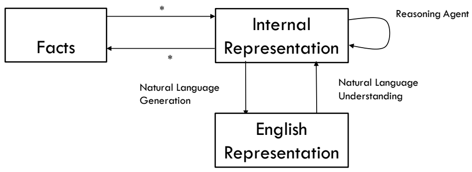
- Helps to address problems like:
  - How do we represent facts about the world?
  - How do we reason about them?
  - What representations are appropriate for dealing with the real world?
- Its objective is to express knowledge in a computer tractable form so that agent can perform well.

Knowledge Representation



Knowledge Representation

Figure: Mapping Facts and Representation



Knowledge Representation: Approaches

- A good system for knowledge representation should have
  - Representable Adequacy: Ability to represent all kind of knowledge that are needed in the domain
  - Inferential Adequacy: Ability to manipulate the representational structure in such a way as to derive new structures corresponding to new knowledge inferred from old
  - Inferential Efficiency: Ability to incorporate into the knowledge structure additional information that can be used to focus the attention of the inference mechanism in the most promising direction
  - Acquisitional Efficiency: Ability to acquire new information easily

Knowledge Representation: Types

- Simple Relational Knowledge
  - The simplest way to represent declarative facts is as a set of relations of the same sort used in database system
- Inheritable Knowledge
  - Structure must be designed to correspond to the inference mechanism that are desired
- Inferential Knowledge
  - Represents knowledge as formal logic
  - Based on reasoning from facts or from other inferential knowledge
  - Useless unless there is also an inference procedure that can exploit it
- Procedural (Imperative) Knowledge
  - Knowledge exercised in the performance of some task
  - Processed by an intelligent agent

Knowledge Representation: Issues

- Are any attributes of objects so basic that they have been occurred in almost every problem domain?
- Are there any important relationships that exist among attributes of objects
- At what level should knowledge be represented?
- How should sets of objects be represented?
- How can relevant parts be accessed when they are needed?

Knowledge Based Agent

- Knowledge Base: a set of sentences
- An agent having a knowledge base
- Each sentence in a knowledge base is expressed in a language called a knowledge representation language
- There must be a way to add new sentences to the knowledge base
- Logical Agents must infer from the knowledge base that has the information from the past or background knowledge

Knowledge Based Agent: Levels of Knowledge Base

- Knowledge Level
  - The most abstract level
  - Describes agent by saying what it knows
  - Example:
    - An intelligent taxi might know that the Bagmati Bridge connects Kathmandu with Lalitpur
- Logical Level
  - The level at which the knowledge is encoded into formal sentences
  - Example:
    - Joins(Bagmati bridge, Kathmandu, Lalitpur)
- Implementation Level
  - Physical representation of the sentences in the logical level
  - Example:
    - Objects, string, dams, etc.

Approaches of system building

13

- Declarative approach
  - Designing the representation language to make it easy to express the knowledge in the form of sentences
- Procedural approach
  - Encoded desired behaviour directly as program code

Logic

14

- Logic
- Syntax: Formal standard to express sentences so that the sentences are well formed
- Semantics: Has to do with the meaning of sentences
  - Defines the truth of the sentences with respect to respective possible world
- Connectives: Joins the different components of the sentence
- Model and Real World
- Entailment: the idea that a sentence follows logically from another sentence
  - Example: $\alpha \models \beta$,where α & β are sentences and β follows from α

Logic

15

- An inference algorithm that derives only entailed sentences is called sound or truth preserving
- Completeness is desirable
  - An inference algorithm is complete if it can derive any sentence that is entailed
- If knowledge base is true in the real world, then any sentence derived from the knowledge base by a sound inference procedure is also true in the real world

Logic

16

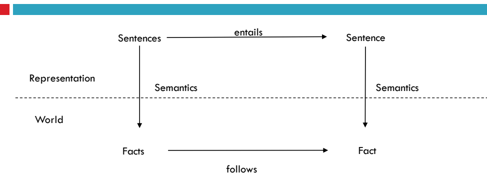


Figure: semantics map sentences in logic to fact in the world

Logic

17

- Example
- Knowledge Base
  - Socrates is a man
  - All men are Mortal
  - Äll men are kind
- Inference algorithm is applied to the above base
- Inferring “Socrates is Mortal”
- “Socrates is kind” follows the sentence “All men are Kind

Truth Table

18

P	Q	!P	P˅Q	P^Q
False	False	True	False	False
False	True	True	True	False
True	False	False	True	False
True	True	False	True	True

Tautology and Validity

19

- A notation used in formal logic which is always true and valid.
- Example: A OR (NOT A)
  I am eating food OR I am nor eating food
- If all the conditions for a statement is true its tautology
- Tautologies are also called valid sentences

Knowledge Models

20

- A model is a world in which a sentence is true under a particular interpretation
- There can be several models at once that have the same interpretations
- Types:
  - First order logic
  - Procedural Representation Model
  - Relational Representation Model
  - Hierarchical Representation Model
  - Semantic Nets

Knowledge Models: Types

21

- First Order Logic
  - First Order Predicate Calculus
  - Consists of objects, predicates on objects, connectives and quantifiers
  - Predicates are the relations between objects or properties of the objects
  - Connectives and quantifiers allow for universal sentences
  - Relations between objects can be true or false
- Procedural Representation Model
  - This model of knowledge representation encodes facts along with the sequence of operations for manipulation and processing of the facts
  - Expert systems are based on this model
  - It works best when experts follow set of procedures for problem solving
  - Example: doctor making diagnosis

Knowledge Models: Types

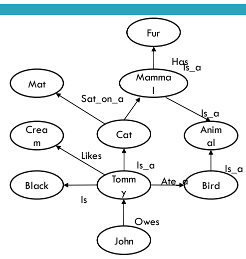
22

- Relational Representation Model
  - Collection of knowledge are stored in tabular form
  - Mostly used in commercial databases, relational databases
  - The information is manipulated with relational calculus use a language like SQL, Oracle, etc.
  - Its flexible way of storing information by not good for storing complex relationships
  - Problem arises when more than one subject area is attempted
  - A new knowledge base from scratch has to be built for each area of expertise
- Hierarchical Representation Model
  - Based on inherited knowledge and the relationship and shared attributes between objects

Knowledge Models: Types

23

- Semantic Nets
  - Semantic networks are an alternative to predicate logic as a form of knowledge representation
  - The idea is that we can store our knowledge in the form of graph with nodes representing objects in the world and are representing relationships between those objects



Propositional Logic

24

- It is declarative sentences which can either be true or false but not both or neither
- A Very simple logic
- A Mathematical model that allows us to reason about the truth or falsehood of logical expressions
- There are sentences and connectives to describe an expression
- Its syntax defines allowable sentences
- Example:
  - Is it raining?
  - Is 2+2=5?
- Logical Connectives in Propositional Logic
  - ^ : Conjunction (and)
  - ᵛ : Disjunction (or)
  - ˥ : Negation (not)
  - -> → : Implication (if…then…)
  - <-> ⇔ : Logical Equivalence (If and only If)

Propositional Logic: Truth Tables

A	˥A
T	F
F	T

A	B	A^B
F	F	F
F	T	F
T	F	F
T	T	T

A	B	AᵛB
F	F	F
F	T	T
T	F	T
T	T	T

A	B	A=>B
F	F	T
F	T	F
T	F	T
T	T	T

A	B	A<=> B
F	F	T
F	T	F
T	F	F
T	T	T

Propositional Logic

- Sentence Properties
  - T or F itself is a sentence
  - Individual Proposition symbols are sentences eg. P, Q, …
  - If s is a sentence, so is (s)
  - If S1 and S2 are sentences, so are: ˥S1, ˥S2, S1^S2, etc.
- Order of Precedence
  - ˥ : Negation (not)
  - ^ : Conjunction (and)
  - ᵛ : Disjunction (or)
  - -> ➔ : Implication (if…then…)
  - <-> ⇔ : Logical Equivalence (If and only If)

Propositional Logic

- Atomic Sentences
  - Single sentence
  - T, F, P, Q,…
    where, each symbol stands for proposition that can be true or false.
  - Example: P="Ram likes Rice"
    Q="Sita is woman"
- Complex Sentences
  - Sentences constructed from simple sentences using logical connectives
  - Example: P="It is hot today"
    Q="It is humid today"
    P^Q
    "It is hot and humid today"

Propositional Logic

- Unsatisfiable (Contradiction)
  - If all the sentences or statements are always false
  - Example: "There will be a clear sky during rainy day"
- Satisfiable
  - If at least one sentence in the knowledge base is true

Propositional Logic: Equivalence Laws

1. P =>Q Ξ ˥ P ᵛ Q
2. P <=> Q Ξ (P => Q) ^ (Q => P)
3. Distributive Laws
   A ^ (B ᵛ C) Ξ (A ^ B) ᵛ (A ^ C)
   A ᵛ (B ^ C) Ξ (A ᵛ B) ^ (A ᵛ C)
4. De-Morgan's Law
   ˥(A ^ B ^ C) Ξ (˥A) ᵛ (˥B) ᵛ (˥C)
   ˥(A ᵛ B ᵛ C) Ξ (˥A) ^ (˥B) ^ (˥C)

Propositional Logic: Inference Rules

1. Modus Ponen Rule
   Whenever any sentence of the form P=>Q and P are given, then the sentence Q can be inferred
   P=>Q, P
   Q
2. And Elimination
   A ^ B
   A|B
   sentence A or B can be inferred if A and B is given
3. And Introduction
   A, B, ……….N
   A^B^……^N
4. Or Introduction
   A, B, ……….N
   AᵛBᵛ……ᵛN
5. Double Negation Elimination
   ˥˥P
   P

Propositional Logic: Inference Rules

6. Unit Resolution
   $\dfrac{A \vee B,\ \neg A}{B}$
7. Modus Tollens
   $\dfrac{P \Rightarrow Q,\ \neg Q}{\neg P}$
8. Resolution Chaining
   $\dfrac{P \Rightarrow Q,\ Q \Rightarrow R}{P \Rightarrow R}$
   $\dfrac{\neg P \Rightarrow Q,\ Q \Rightarrow R}{\neg P \Rightarrow R}$

Propositional Logic

- The semantics defines the rules for determining the truth of sentences with respect to a particular model, i.e. semantic must specify how to compute the truth value of any sentence in a given model.

Propositional Logic: BNF Grammar

- Backus Normal Form or Backus Naur Form
- It's a notation technique for context free grammars often used to describe the syntax of languages used in computing
- BNF can be used in two ways:
  - To generate strings belonging to the grammar
  - To recognize strings belonging to the grammar

Normal Forms of Propositional Logic Sentences

1. Conjunctive (disjunction of conjunction of literals) Normal Form
   - In which a sentence is written as the conjunction of literals
     $(A \vee B) \Rightarrow Q$
     $\Xi\ \neg(A \vee B) \vee Q$
     $\Xi\ (\neg A \wedge \neg B) \vee Q$
     $\Xi\ (\neg A \vee Q) \wedge (\neg B \vee Q)$
2. Disjunctive (conjunction of disjunction of literals) Normal Form
   - In which a sentence is written as the disjunction of literals
     $(A \wedge Q) \vee (B \wedge Q)$

First Order Predicate Logic (FOPL)

- Propositional logic assumes that the world or system being modelled can be described in terms of fixed, known set of propositions
- This assumption can make it awkward or even impossible to specify many pieces of knowledge
- Example:
  - Consider a general sentence "if a person is rich then they have a nice car"
  - In propositional logic, we can generate rule for each person as
    - Bob_is_rich → Bob_has_a_nice_car
    - John_is_rich→John_has_a_nice_car
  - This seems to be an impractical way to represent knowledge, hence, generalization to represent this type of knowledge is a must

First Order Predicate Logic (FOPL)

- FOPL is a logic that gives us the ability to quantify over objects
- In FOPL, statements from a natural language like English are translated into symbolic structure composed of predicates, functions, variables, constants, quantifiers and logical connectives
- First Order Predicate Logic represents facts by separating classes and individuals and consider that world consists of different objects and relations between those objects

FOPL: Syntax

Sentence → AtomicSentence | (Sentence Connective Sentence) | Quantifier Variable,…Sentence | ˥Sentence

AtomicSentence → Predicate(Term,….) | Term = Term

Term → Function (Term,…) | Constant | Variable

Connective → ˥|˅|˄|=>|<=>

Quantifier → ∀| ∃

Constant → A| X| John| …

Variable → a | x | s| …

Predicate → Before | HasColor | Raining | …

Function → Mother | Leftleg | …

FOPL: Syntax

- Constant Symbols are the strings that will be interpreted as representing objects
- Variable Symbols are used as place holders for quantifying over objects
- Predicate symbols are used to denote properties of objects and relationship among them
- Function Symbols map the specified number of input objects to objects
- Quantifiers are used to quantify objects
  - Universal Quantifier represents for all
  - Existential Quantifier represents the existence of an object

FOPL: Variable Scope

- The scope of the variable is in the sentence to which the quantifier syntactically applies
- In a block structured programming language, a variable in a logical expression refers to the closest quantifier within whose scope it appears
- In a well formed formula all the variables should be properly introduced

Relation Between Quantifiers

- $\forall x \, \neg P \;\equiv\; \neg \, \exists x \, P$
- $\neg \forall x \, P \;\equiv\; \exists x \, \neg P$
- $\forall x P \equiv \neg \exists x \neg P$
- $\exists x P \equiv \neg \forall x \neg P$
- $\forall x \; P(x) \cap Q(x) \equiv \forall x \; P(x) \cap \forall x \; Q(x)$
- $\exists x \; P(x) \cup Q(x) \equiv \exists x \; P(x) \cup \exists x \; Q(x)$

Examples

- All birds can't fly
  $\forall x \; Bird(x)$
  OR
  $\neg(\exists x \; (Bird(x) \cap Fly(x))$
- Not all birds can fly
  $\neg(\forall x \; Bird(x)$
- If anyone can solve the problem then Raju can
  $\exists x Solves(x, problem) \rightarrow Solves(Raju, problem)$
- Try these
  - Nobody in electrical class is smarter than everyone in AI class
  - John hates all the people who don't hate themselves

Equality

- Can include equality as a primitive predicate in the logic or require it to be introduces and axiomitized as the identity relation
- Useful in representing certain types of knowledge
  - Example: Sita owns two cars
    $\exists x \exists y (Owns \; (Sita, x) \cap Owns \; (Sita, y) \cap Car(x) \cap Car(y) \cap \neg(x = y))$
- Try these:
  - There are exactly two purple flowers out of three
  - Everyone is married to exactly one person

Solution

- There are exactly two purple mushroom out of three
  $\exists x \exists y Mushroom(x)$
  $\cap\ Mushroom(y) \cap Purple(x)$
  $\cap\ Purple(y) \cap \neg(x = y)$
  $\cap\ \forall z (Mushroom(z)$
  $\cap\ Purple(z)$
  $\Rightarrow ((x = z) \cup (y = z)))$
- Everyone is married to exactly one person
  $\forall x \exists y Married(x, y)$
  $\cap\ \forall z (Married(x, z)$
  $\Rightarrow (y = z))$

Try few more

- Ram likes all kinds of food
- Anything anyone eats and is not killed by is food
- Rita eats samosa and is still alive
- Gita eats everything Rita eats
- Someone who hates something owned by another person will not love that person
- There is a barber in the town who shaves all men in the town who don't shaves themselves
- Everyone loves somebody
- No one likes everyone
- There is someone who is liked by everyone
- You can fool some of the people every time
- All employee earning Rs.200000|- or more per year pay taxes
- Some employee are sick today
- Nobody earns more than the chairman

Try few more: Solution

- Ram likes all kinds of food
- Anything anyone eats and is not killed by is food
- Rita eats samosa and is still alive
- Gita eats everything Rita eats
- Someone who hates something owned by another person will not love that person
- There is a barber in the town who shaves all men in the town who don't shaves themselves

$\forall x Food(x) \Rightarrow Likes(John, x)$
$\forall x \forall y Eats(x, y) \cap \neg Killedby(y, x)$
$\Rightarrow food(x)$
$Eats(Rita, Samosa) \cap Alive(Rita)$
$\forall x Eats(Rita, x) \Rightarrow Eats(Gita, x)$
$\exists x \exists y \exists z Owns(x, z) \cap Hates(y, z)$
$\Rightarrow Hates(y, x)$
$\exists x (Barber(x) \cap Intown(x)$
$\cap\ \forall y Man(y) \cap Intown(y)$
$\cap\ \neg Shaves(y, y) \Rightarrow Shaves(x, y)$

Try few more: Solution

- Everyone loves somebody
- No one likes everyone
- There is someone who is liked by everyone
- You can fool some of the people every time
- All employee earning Rs.200000|- or more per year pay taxes
- Some employee are sick today
- Nobody earns more than the chairman

$\forall x \exists y Loves(x, y)$
$\neg \exists x \forall y Likes(x, y) \equiv \forall x \exists y \neg Likes(x, y)$

$\exists y \forall x Likes(x, y)$
$\exists x \forall y Person(x) \cap Time(y)$
$\Rightarrow Canbefooled(x, t)$
$\forall x Employee(x)$
$\cap\ Earnmorethan(x, 200000)$
$\Rightarrow Paytax(x)$
$\exists x Employee(x) \Rightarrow Sick(x)$
$\forall x Employee(x)$
$\Rightarrow \neg Earnmorethan(x, Salary(Chairman))$

Horn Clause

- Disjunction of literals of which at most one is positive is Horn Clause

  $P1 \cap P2 \cap \cdots \cap Pn \Rightarrow Q$
  $\equiv \neg P1 \cup \neg P2 \cup \cdots \cup \neg Pn \cup Q$

- Clause with exactly one positive literals giving definite clause (fact)
- Horn clause with no positive literals can be written as an implication whose conclusion is the literal false
  $\neg x1 \cup \neg x2 \equiv x1 \cap x2 \Rightarrow False$

Horn Clause

<u>Reason for its importance</u>

- Every horn clause can be written as an implication whose premises is a conjunction of positive literals and whose conclusion is a single positive literal
  Example: $\neg L1 \cup \neg L2 \cup B$ can be written as $L1 \cap L2 \Rightarrow B$
- Inference with horn clauses can be done with the forward chaining and backward chaining
- Deciding entailment with horn clauses can be done in time that is linear in the size of knowledge base

Well Formed Formula

49

- A sentence that has all its variables properly introduced using quantifiers is a well formed formula
- Example:
  $\forall x P(x, y)$ is not a well formed formula where x is bounded as universal quantifier and y is free
  $\forall x \exists y Q(x, y)$ is a well formed formula where both x and y are bounded
- Notes:
  - Predicate can't be quantifiers
  - Constant can't be negative
  - Letter cases must be well considered

Inference in FOL

50

- If x is a parent of y, then x is older than y
- If x is the mother of y then x is a parent of y
- Devaki is the mother of Krishna
- Conclusion:
  Devaki is older than Krishna

<u>Mapping in FOL</u>

- $\forall x \forall y\ parent(x, y) \Rightarrow older(x, y)$
- $\forall x \forall y\ mother(x, y) \Rightarrow parent(x, y)$
- $mother(Devaki, Krishna)$
- Conclusion:
  older(Devaki,Krishna)

Inference Rules in FOL

51

- Universal Instantiation
  - If a person is a student, studies in KEC and studies AI, then he/she is a third year student
  - $\forall x student(x) \cap studiesin(x, KEC) \cap studies(x, AI) \Rightarrow thirdyearstudent(x)$
- Existential Instantiation
  - There must be a topper in KEC
  - $\exists x student(x) \cap studiesin(x, KEC) \cap topper(x)$
- Propositionization
  - All people are kind
    $\forall x person(x) \Rightarrow kind(x)$
    It can be inferred as
    $person(Ram) \Rightarrow kind(Ram)$

Inference Rules in FOL

52

- Generalized Modus Ponens
  - $\forall x student(x) \cap studieshard(x) \Rightarrow good\ student\ (x)$
  - $student(Arjun)$
  - $studieshard(Arjun)$
  - Conclusion:
    $goodstudent(Arjun)$
- Unification
  - $[knows(Sita, x),$
  -
  -
  -

Inference Rules in FOL

53

- Resolution
  - Produces proof by refutation (proof person or statement that is wrong)
  - Resolution can be applied to sentences in CNF (conjunctive normal form)
- Process of Resolution
  - Convert all sentences to CNF
  - Negate x
  - Add negate x to premises
  - Repeat until either a contradiction is detected or no progress is being made

CNF Conversion Process

54

1. Elimination of all implications with equivalence symbols
   - $P \rightarrow Q \equiv \neg P \cup Q$
   - $P <=> Q \equiv (\neg P \cup Q) \cap (\neg Q \cup P)$
2. Move ¬ inward (use De'Morgans law)
   - $\neg(P \cap Q) \equiv \neg P \cup \neg Q$
   - $\neg(P \cup Q) \equiv \neg P \cap \neg Q$
   - $\forall x\ \neg P\ \equiv\ \neg \exists x\ P$
   - $\neg \forall x\ P\ \equiv\ \exists x\ \neg P$
   - $\forall x P\ \equiv\ \neg \exists x \neg P$
   - $\exists x P\ \equiv \neg \forall x \neg P$
3. Standardize Variables
   - Rename variables if necessary so that all quantifiers have different variable assignments

CNF Conversion Process

55

4. Skolemization
   - The process of eliminating the existential quantifiers through a substitution process
   - The process requires that all such variables be replaced by short term functions, which can always assume a Skolen function, a correct value required for an existential quantifier variable
   - If leftmost quantifier in an expression is existential quantifier (∃), replace all occurrence of the variables that quantifies with an arbitrary constant not appearing elsewhere in the expression and delete the quantifier
     - Example: $\exists x \exists y \forall z P(x,y,z) \cup Q(x,y) \equiv \forall z P(a,b,z) \cup Q(a,b)$

CNF Conversion Process

56

4. Skolemization
   - If existential quantifier (∃) is preceded by universal quantifier (∀), replace the existentially quantified variable by a function symbol whose arguments are variable appearing in those universal quantifiers
   - Example:
     $\exists u \forall x \forall y \exists z\, P(f(u),x,y,z) \cup Q(x,y,z)$
     $\equiv\ \forall x \forall y \exists z\, P(f(a),x,y,z) \cup Q(x,y,z)$
     $\equiv\ \forall x \forall y\, P(f(a),x,y,f(x,y)) \cup Q(x,y,f(x,y))$
5. Drop all universal quantifiers
6. Distribute ^over ˅

Example: Given Premises

57

1. If x is on top of y, y support x
2. If x is above y and they are touching each other, x is on top of y
3. Everything is on top of another thing
4. A cup is above a book
5. A cup is touching a book

- Answer:

  Is the book supporting the cup?

Example: Solution

58

- $\forall x \forall y\ ontop(x,y) \Rightarrow supports\ (y,x)$
  $Implication\ Elimination$
  $\forall x \forall y \neg ontop(x,y) \cup supports\ (y,x)$
  $Drop\ \ \forall x\ and\ \forall y$
  $\neg ontop(x,y)$
  $\Rightarrow supports\ (y,x)$
- $\forall x,y\ above(x,y) \cap touch(x,y) \Rightarrow ontop(x,y)$
  $Implication\ Elimination$
  $\forall x,y\ \neg above(x,y) \cup \neg touch(x,y) \cup ontop(x,y)$
  $Drop\ \forall x,y$
  $\neg above(x,y) \cup \neg touch(x,y) \cup ontop(x,y)$

Example: Solution

59

- $\forall x,y\ ontop(x,y)$
  $Drop\ \forall x,y$
  $ontop(x,y)$
- $above(cup, book)$
- $touch(cup, book)$

Solution

60

Conclusion

- $supports(book, cup)$
  Let
  $\neg supports(book, cup)$
  $using\ second\ and\ fifth\ conditions$
  $\neg above(x,y) \cup \neg touch(x,y) \cup ontop(x,y)$
  $touch(cup, book)$
  $\neg above(x,y) \cup ontop(x,y)$
  $using\ fourth\ condition$
  $above(cup, book)$
  $ontop(x,y)$
  $using\ first\ condition$
  $\neg ontop\ (x,y) \cup supports(y,x)$
  $supports(book, cup)$
  $using\ assumed\ condition$
  $\neg supports(book, cup)$
  $Empty\ Clause$
  $Hence, the\ book\ is\ supporting\ the\ cup$

Try these

- Every American who sells weapon to hostile nation is a criminal. The country Iraq is an enemy of America. All of the missiles in Iraq were sold by George. George is an American.
  Prove:
  George is a Criminal
- All Pompeiians are Romans. All Romans were either loyal to Caesar or hated him. Everyone is loyal to someone. People only try to assassinate rulers they are not loyal to. Marcus tried to assassinate Caesar. Marcus was a Pompeian.
  Conclude:
  Did Marcus hare Caesar?

Forward Chaining

- One of the two main methods for reasoning using inference rules
- Can be described logically as repeated application of Modus Ponens
- It's a popular strategy of reasoning in expert system and production systems
- It starts with the available data and uses inference rules to extract more data until a goal is reached
- An inference engine using forward chaining searches the inference rules until it founds one where antecedent (If clause) is known to be true

Forward Chaining

- When it found if clause it can conclude or infer the consequent (then clause) to its data resulting in the addition of new information
- Example: (Animal Identification System)
  If X croaks and eats flies then it's a frog
  If X chirps and sings then it's a canary

If X is a frog then X is green
If X is a canary then X is yellow
goal: colour of pet given that it croaks and eat flies

- In above example first clause describing the animal that croaks and eat flies will be examined, i.e. first statement
- Then on the basis of consequent that it's a frog colour will be determined using third statement

Forward Chaining: Steps

- The rule base would be searched and the first suitable rule would be selected which would be an antecedent matched
- Now the consequent is added to the data
- The rule base is again searched, this selecting some other rule
- Steps 2 and onward is repeated until no more data can be inferred from the given information
- Hence this technique is also called data driven inference
- It is often referred as goal driven reasoning

Backward Chaining

- One of the two most commonly used method of reasoning with inference rules
- Backward chaining is used in logic programming, automated theorem provers, etc.
- Backward chaining starts with a list of goals or hypothesis and backward from consequent to antecedent to see if there is data available that will support any of the consequents
- An inference engine using this technique would search the inference rules until it finds one which has a consequent that matched desired goal
- If the antecedent of that rule is not known to be true, then it is added to the list of goals
- This technique is also based on Modus Ponens

Backward Chaining: Steps

- First consequent resulting the asked criteria are chosen
- Then the antecedents resulting from those statements are added as goals
- The second step is repeated until the desired result is achieved
- For the same example:
  - The rule base is searched and third and fourth rules are selected as those results to match the goal i.e. find the colour
  - Since it is not known that pet is frog so both the antecedents are added as goals
  - The rule base is searched again selecting the first two rules that matches the new goals just added to the list
  - The antecedent is known to be true for the first statement hence it can be concluded that pet is a frog not canary
  - Finally goal is determined i.e. the required colour of animal that croaks and eats flies is green

67

Structured Knowledge Representation

68

Knowledge

- Data
- Information
- Knowledge

- Tacit Knowledge
- Recorded Knowledge

- Knowledge Engineering

69

Representations and Mappings

- Solving Complex Problems is guided by requirement of large amount of knowledge and some mechanisms to manipulate that knowledge and create the solution to the Problem
- For that knowledge is to be represented for which the following points are to be considered
  - Facts: that we want to represent, i.e. the truth in the representing world
  - Representation: in some formal way

70

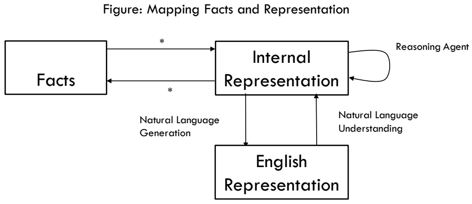
Representations and Mappings

- For structuring these entities one way is to think at two levels:
  - The knowledge level
  - The symbol level
- At knowledge level facts are described
- At symbol level objects represented at knowledge level are defined in terms of symbols that can be manipulated by programs

71

Representations and Mappings (Reviewing)

Figure: Mapping Facts and Representation



72

Representations and Mappings

- Rather than thinking of one level on top of another, focusing on facts, representations and on the two way mappings that must exist between them is more important
- These links are called Representation Mappings
- Forward Representation Mapping maps facts to representations
- Backward Representation Mapping maps representations to facts

Representations and Mappings

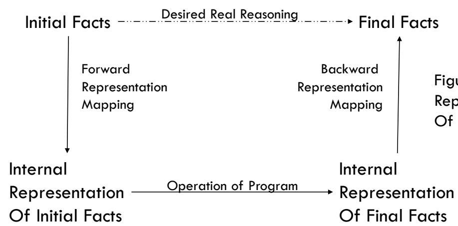


Figure: Representation Of Facts

Approaches to Knowledge Representation

For a good system following four properties are must
- Representational Adequacy: Ability to represent all kind of knowledge that are needed in the domain
- Inferential Adequacy: Ability to manipulate the representational structure in such a way as to derive new structures corresponding to new knowledge inferred from old
- Inferential Efficiency: Ability to incorporate into the knowledge structure additional information that can be used to focus the attention of the inference mechanism in the most promising direction
- Acquisitional Efficiency: Ability to acquire new information easily

Knowledge Representation: Types

- Simple Relational Knowledge
  - The simplest way to represent declarative facts is as a set of relations of the same sort used in database system
- Inheritable Knowledge
  - Structure must be designed to correspond to the inference mechanism that are desired
- Inferential Knowledge
  - Represents knowledge as formal logic
  - Based on reasoning from facts or from other inferential knowledge
  - Useless unless there is also an inference procedure that can exploit it
- Procedural (Imperative) Knowledge
  - Knowledge exercised in the performance of some task
  - Processed by an intelligent agent

Issues in Knowledge Representation

- Are any attributes of objects so basic that they have been occurred in almost every problem domain?
- Are there any important relationships that exist among attributes of objects
- At what level should knowledge be represented?
- How should sets of objects be represented?
- How can relevant parts be accessed when they are needed?

Semantic Networks

- Other than descriptive logic, the major system designed to organise and for reasoning
- Evolved from existential graphs, called the logic of the future
- Existential graphs uses a graphical notation of nodes and arcs
- Semantic networks provide for certain kinds of sentences is often more convenient but if we strip away the human interface issues, the underlying concept persist with objects, relations, quantification and so on

Semantic Networks

- Many variant of semantic net are available now a days
- Semantic nets are capable of representing individual objects, categories of objects and relationships among those objects
- A typical graphical notation displays object or categories names in ovals or boxes and connects them with labelled arcs
- Suitable for implementing inheritance and object oriented concepts
- Inverse Links: Example→ Brother of (x, y) = Has brother (y, x)

Semantic Networks

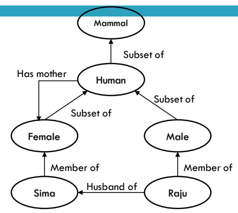


Figure: Semantic Network Example

Frames

- A frame is a collection of attributes (slots) and associated values and possibly constraints on values that describe some entity in the world.
- Frame system is build on a set of frames that are connected to each other by the virtue of fact that the value of an attribute of one frame may be another frame
- Generally, Frames are based on set theory

Frames

Example:

<u>Human</u>
Isa: Mammal
Cardinality: 6000000000
*Legs: 2
<u>Male</u>
Isa: Human
Cardinality: 4000000000
*Hair: Short

Conceptual Dependencies

- Theory of how to represent the kind of knowledge about events that is usually contained in natural language sentences
- Goal:
  - Concluding inferences from the sentences
  - Independent of the language in which the sentences were originally stated

Conceptual Dependencies

- Use conceptual primitives that can be combined to form the meaning of the word in any particular language
- Provides both a structure and a specific set of primitives for information construction

Conceptual Dependencies

- Example: I gave the man a book

P o R to man
I ←→ATRANS ←book ←
from I

- Where, the symbols have following meaning
  - Arrows indicates direction of dependency
  - Double arrow indicates two way link between actor and action
  - P indicates Past Tense, O indicates object case relation, R indicates reciepient case relation
  - ATRANS indicates transfer of possession(Primitive)

Conceptual Dependencies: Primitive Acts

- ATRANS: Transfer an abstract relationship, example: take
- PTRANS: Transfer of physical location of an object, example: come
- PROPEL: Application of physical force to an object, example: pull
- MOVE: Movement of body part by its owner, example: punch
- GRASP: Grasping of an object by an actor, example: clutch
- INGEST: Ingestion of an object by an animal, example: eat
- EXPEL: Expulsion of something from the body of an animal, example: spit
- MTRANS: Transfer of Mental Information, example: tell
- MBUILD: Building new information out of old, example: decide
- SPEAK: Production of sounds, example: say
- ATTEND: Focusing on sense organ toward a stimulus, example: Listen

Conceptual Dependencies: Set of tenses

- p: Past
- f: Future
- t : Transition
- t_s: Start Transition
- t_f: Finish Transition
- k: Continuing
- ?: Interrogative
- /: Negative
- nil: Present
- Delta: Timeless
- c: Conditional

Scripts

- A script is a structure that is used to describe the sequence of events in a particular context
- It consists of a set of slots
- Each slot is associated with some information describing the kind of values a slot may contain as well as a default value to be used if no other information is available
- Script seems to be similar to frames but these have more detailed information
- For example: refer to Page Number 286, Artificial Intelligence, Rich and Knight.

88

Statical Reasoning

Statistical Reasoning

- One of the most common characteristics of the human information available is its imperfection due to partial observability, non deterministic or combination of both
- An agent may not know what state it is in or will be after certain sequence of actions
- Agent can cope with these defects and make rational judgments and rational decisions to handle such uncertainty and draw valid conclusions

Statistical Reasoning

What is uncertainty?

- The lack of the exact knowledge that would enable us to reach a perfectly reliable conclusion
- Classical Logic permits only exact reasoning i.e. perfect knowledge always exists

IF A is true THEN A is not false and IF B is true THEN B is not false

- In Real world such clear cut knowledge could not be provided to systems

Statistical Reasoning

91

Sources of Uncertain Knowledge

- **Weak Implication:** Domain experts and knowledge engineer have rather painful or hopeless task of establishing concrete correlation between IF(Condition) and THEN(action) part of rules. Vague Data.
- **Imprecise Language :** NLP is ambiguous and imprecise. We define facts in terms of **often, sometimes, frequently, hardly ever.** Such can affect IF-THEN implication
- **Unknown Data:** incomplete and missing data should be processes to an approx. reasoning with this values
- **Combining the views of different experts:** Large system uses data from many experts

Statistical Reasoning

92

- The basic Concept of probability plays significant role in our life like we try to determine the probability of rain, prospect of promotion, likely hood of winning in Black Jack
- The probability of an event is the proportion of cases in which the event occurs (Good, 1959)
- Probability, mathematically, is indexed between 0 and 1
- Most events have probability index strictly between 0 and 1, which means that each event has at lease two possible outcomes: favorable outcome or success and unfavorable outcomes or failure

$$\text{P(success)} = \frac{\textit{The number of successes}}{\textit{The number of possible outcomes}}$$

$$\text{P}(failure) = \frac{\textit{The number of failure}}{\textit{The number of possible outcomes}}$$

Statistical Reasoning

93

- If s is the number of success and f is the number of failure then:

$$\text{P}(success) = \frac{s}{s+f}$$

$$\text{P}(failure) = \frac{f}{s+f}$$

and

$$p + q = 1$$

Statistical Reasoning

94

- Let us consider classical examples with a coin and a dice. If we throw a coin, the probability of getting a head will be equal to the probability of getting a tail. In a single throw, s = f = 1, and therefore the probability of getting a head (or a tail) is 0.5.
- Consider now a dice and determine the probability of getting a 6 from a single throw. If we assume a 6 as the only success, then s = 1 and f = 5, since there is just one way of getting a 6, and there are five ways of not getting a 6 in a single throw. Therefore, the probability of getting a 6 is

$$\text{P} = \frac{1}{1+5} = 0.1666$$

Likewise, the probability of not getting 6 is

$$\text{q} = \frac{5}{1+5} = 0.8333$$

Statistical Reasoning

95

- Above instances are for independent events i.e. mutually exclusive events which can not happen simultaneously
- In the dice experiment, the two events of obtaining a 6 and, for example, a 1 are mutually exclusive because we cannot obtain a 6 and a 1 simultaneously in a single throw. However, events that are not independent may affect the likelihood of one or the other occurring. Consider, for instance, the probability of getting a 6 in a single throw, knowing this time that a 1 has not come up. There are still five ways of not getting a 6, but one of them can be eliminated as we know that a 1 has not been obtained. Thus,

$$\text{p} = \frac{1}{1+(5-1)}$$

Statistical Reasoning

96

- Let A and B be two not mutually exclusive events, but occur conditionally on the occurrence of other.
- The probability of event A will occur if event B occurs is called conditional Probability

$$\text{p(A|B)} = \frac{\textit{the number of times A and B can occur}}{\textit{the number of times B can occur}}$$

The probability of both A and B will occur is called joint probability $(A \cap B)$

$\text{p(A|B)} = \frac{p(A \cap B)}{p(B)}$, the probability of A occurring given B has occurred

$\text{p(B|A)} = \frac{p(B \cap A)}{p(A)}$, the probability of B occurring given A has occurred

Statistical Reasoning

- Joint probability is commutative, thus

$$p(A \cap B) = p(B \cap A)$$

Therefore,

$$p(A \cap B) = p(B|A) * p(A)$$

Now the final equation becomes:

$p(A|B) = \frac{p(B|A) * p(A)}{p(B)}$ --(a)

Where:

p(A|B) is the conditional probability that event A occurs given event B has occurred

p(B|A)is the conditional probability that event B occurs given event A has occurred

p(A)is the probability of event A occurring   p(B)is the probability of event B occurring

The above equation (a) is known as **Bayesian Rule**

Statistical Reasoning

- For *n* number of mutually exclusive event B we have

$$p(A \cap B_1) = p(A|B_1) \times p(B_1)$$
$$p(A \cap B_2) = p(A|B_2) \times p(B_2)$$
$$\vdots$$
$$p(A \cap B_n) = p(A|B_n) \times p(B_n)$$

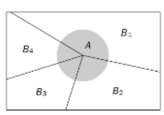
or when combined:

$$\sum_{i=1}^{n} p(A \cap B_i) = \sum_{i=1}^{n} p(A|B_i) \times p(B_i)$$

Statistical Reasoning

- Summed over an exhaustive list of events for Bi, we get :

$$\sum_{i=1}^{n} p(A \cap B_i) = p(A)$$



- $$p(A) = \sum_{i=1}^{n} p(A|B_i) \times p(B_i)$$

Statistical Reasoning

- If the occurrence of A depends on only two mutually exclusive events, i.e. B and NOT B, then above equation becomes

$$p(A) = p(A|B) \times p(B) + p(A|\neg B) \times p(\neg B)$$

- Similarly,

$$p(B) = p(B|A) \times p(A) + p(B|\neg A) \times p(\neg A)$$

- Substituting above equations in Bayesian Equation, We get:

$$p(A|B) = \frac{p(B|A) \times p(A)}{p(B|A) \times p(A) + p(B|\neg A) \times p(\neg A)}$$

Statistical Reasoning: Bayesian Networks

Why Bayesian Network???

- To represent the probabilistic relationship between two different classes
- To avoid dependencies between values of attributes by joint conditional probability distribution
- In Naïve Bayes classifier, attributes are conditionally independent

Statistical Reasoning: Bayesian Networks

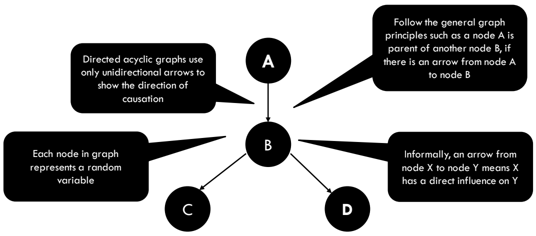
- Bayesian Network are also known as **Bayes Network, Belief Networks** and **Probabilistic Networks**
- A BN is defined by two parts, **Directed Acyclic Graph (DAG)** and **Conditional Probability Tables (CPT)**

Nodes → Random Variables

Arcs→ Indicates Probabilistic dependencies between nodes

Statistical Reasoning: Bayesian Networks

103



Statistical Reasoning: Bayesian Networks

104

A BN is a directed graph with the following properties:

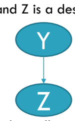
- **Nodes:** Set of Random Variables which may be discrete or continuous
- **Directed Links (Arcs) :** The real meaning od a link from node X to node Y is that X has a direct influence on Y
- Each node has a Conditional Probability Distribution $\mathbf{P}(X_i|Parents(X_i))$ that quantifies the effects that the parent have on the node
- The graph has no directed cycles

Statistical Reasoning: Bayesian Networks

105

A BN is a directed graph with the following properties (contd…)

- If an arc is drawn from Y to Z, then Y is a parent or immediate predecessor of Z, and Z is a descendant of Y



- Each variable is conditionally independent of its non-descendants in the graph, given its parents

Statistical Reasoning: Bayesian Networks

106

Incremental Network Construction:

1. ***Nodes***: First determine the set of variables that are required to model the domain. Now order them, $\{X_1, X_2, \ldots, X_n\}$. Any order will work, but the resulting network will be more compact if the variables are ordered such that causes precede effects
2. ***Links*** : for $i = 1\ to\ n\quad do$:
   1. Choose, from $X_1, \ldots, X_{n-1}$, a minimal set of parents for X_i such that equation $\mathbf{P}(X_i|X_{i-1}, \ldots, X_1)$= $\mathbf{P}(X_i|Parents(X_i))$ is satisfied
   2. For each parent insert a link from the parent to X_i
   3. CPTs: Write down the Conditional Probability Table, $\mathbf{P}(X_i|Parents(X_i))$

Statistical Reasoning: Bayesian Networks

107

Conditional Independence:

$\mathbf{P}(X_1, X_2, \ldots, X_n) = \mathbf{P}(X_n|X_{n-1}, \ldots, X_1)\ \mathbf{P}(X_{n-1}, \ldots, X_1)$

$= \mathbf{P}(X_n|X_{n-1}, \ldots, X_1)$

$\mathbf{P}(X_{n-1}, \ldots, X_1) \ldots \mathbf{P}(X_2|X_1)\mathbf{P}(X_1)$

$= \sum_{i=1}^{n} \mathbf{P}(X_i|Parents(X_i))$

A BN represents Conditional Independence

$\mathbf{P}(X_i|X_{i-1}, \ldots, X_1)$= $\mathbf{P}(X_i|Parents(X_i))$

Statistical Reasoning: Bayesian Networks

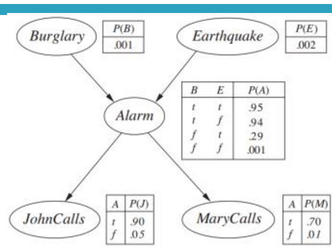
108

Example

- Burglar Alarm at Home
  - Fairly reliable at detecting a Burglary
  - Also Respond at times of Earthquake
- Two neighbors (John and Mary) on hearing Alarm calls you
  - John always calls when he hears the alarm, but sometimes confuses the telephone ringing with the alarm and calls then too
  - Mary likes aloud music and sometimes misses the alarm altogether

Statistical Reasoning: Bayesian Networks

109



Statistical Reasoning: Bayesian Networks

110

- Inference from Effect to cause; given Burglary, what is P(J|B)?

  P(J | B) = ?

  first calculate probability of Alarm ringing on burglary:

  P(A | B) = P(B)P(¬E)P(B∩ ¬E) + P(B)P(E)P(B∩E)

  P(A | B) = 1*(0.998)*(0.94) + 1*(0.002)*(0.95)

  P(A | B) = 0.94

  Now, Let us calculate P(J | B)

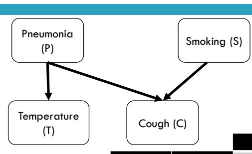
  P(J | B) = P(A | B)*P(J) + P(¬(A | B))*P(¬J)

  P(J | B) = (0.94) * (0.9) + (0.06) * (0.05) = 0.85
- Also calculate P(M|B) = ?

Bayesian Network

111

Pneumonia	
True	0.1
False	0.9



smoking	
True	0.2
False	0.8

	Temperature	
Pneumonia	Yes	No
Yes	0.9	0.1
No	0.2	0.8

		Cough	
Pneumonia	Smoking	True	False
True	Yes	0.95	0.05
True	No	0.8	0.2
False	Yes	0.6	0.4
False	No	0.05	0.95

Find:

a. P(C| S ^ P)

b. P(S | C)

Statistical Reasoning: Bayesian Networks

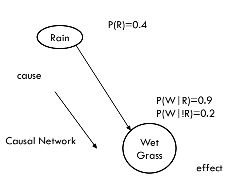
112

Benefits of BN:

- It can readily handle incomplete data sets
- It allows one to learn about causal relationships
- It readily facilitate use of prior knowledge
- It Provide a natural representation for conditional independence
- It is more complex to construct the graph

Example

113



- What is the probability that rain is the cause?

$$P(R|W) = \frac{P(W|R)P(R)}{P(W)} = \frac{P(W|R)P(R)}{P(W|R)P(R) + P(W|!R)P(!R)} = \frac{0.9X0.4}{0.9X0.4 + 0.2X0.6} = 0.75$$

References

114

- Russell, S. and Norvig, P., 2011, Artificial Intelligence: A Modern Approach, Pearson, India.
- Rich, E. and Knight, K., 2004, Artificial Intelligence, Tata McGraw hill, India.

115 Thank You

Any Queries?

Now, Search for yourself.