Semantic Nets, Frames, Scripts, CGs, CDs

Semantic Nets

What is a Semantic Net?

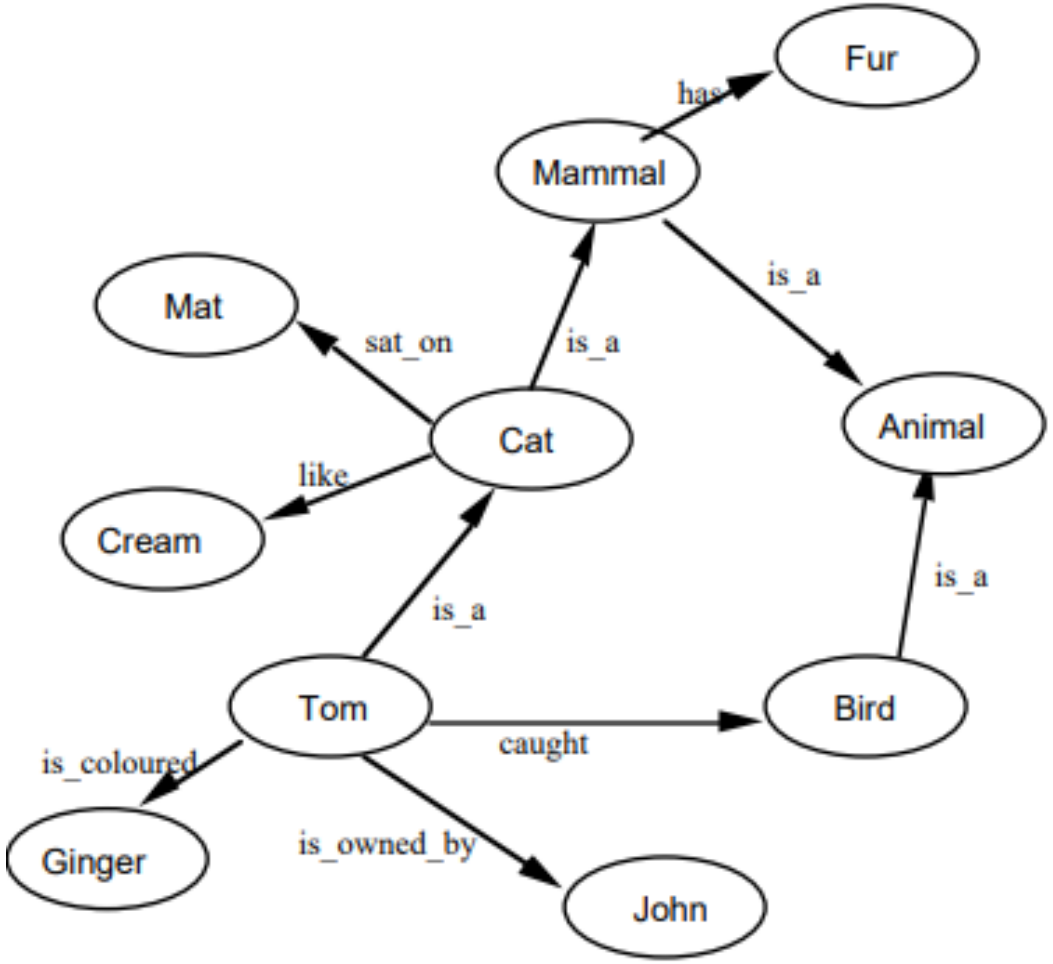
- Semantic nets are an alternative to predicate logic as a form of knowledge representation.
- The idea is that we can store our knowledge in the form of a graph, with nodes representing objects in the world, and arcs representing relationships between those objects.

Structure of a Semantic Net

- Nodes are sometimes referred to as objects
- Arcs as links or edges
- The links are used to express relationships
- Nodes are to represent physical objects, concepts, or situation

Example

- Tom is a cat.
- Tom caught a bird.
- Tom is owned by John.
- Tom is ginger in colour.
- Cats like cream.
- The cat sat on the mat.
- A cat is a mammal.
- A bird is an animal.
- All mammals are animals.
- Mammals have fur.



Semantic Nets vs Predicate Logic

- It is argued that this form of representation is closer to the way humans structure knowledge by building mental links between things than the predicate logic we considered earlier.
- Note in particular how all the information about a particular object is concentrated on the node representing that object, rather than scattered around several clauses in logic.

Links in Semantic Nets

- Two types of commonly used links are:
  - IS-A, and
  - A-KIND-OF

IS-A Link

- IS-A means "is an instance of' and refers to a specific member of a class
  - A class is related to the mathematical concept of a set in that it refers to a group of objects
    - For example, {3, eggs, blue, tires, art}

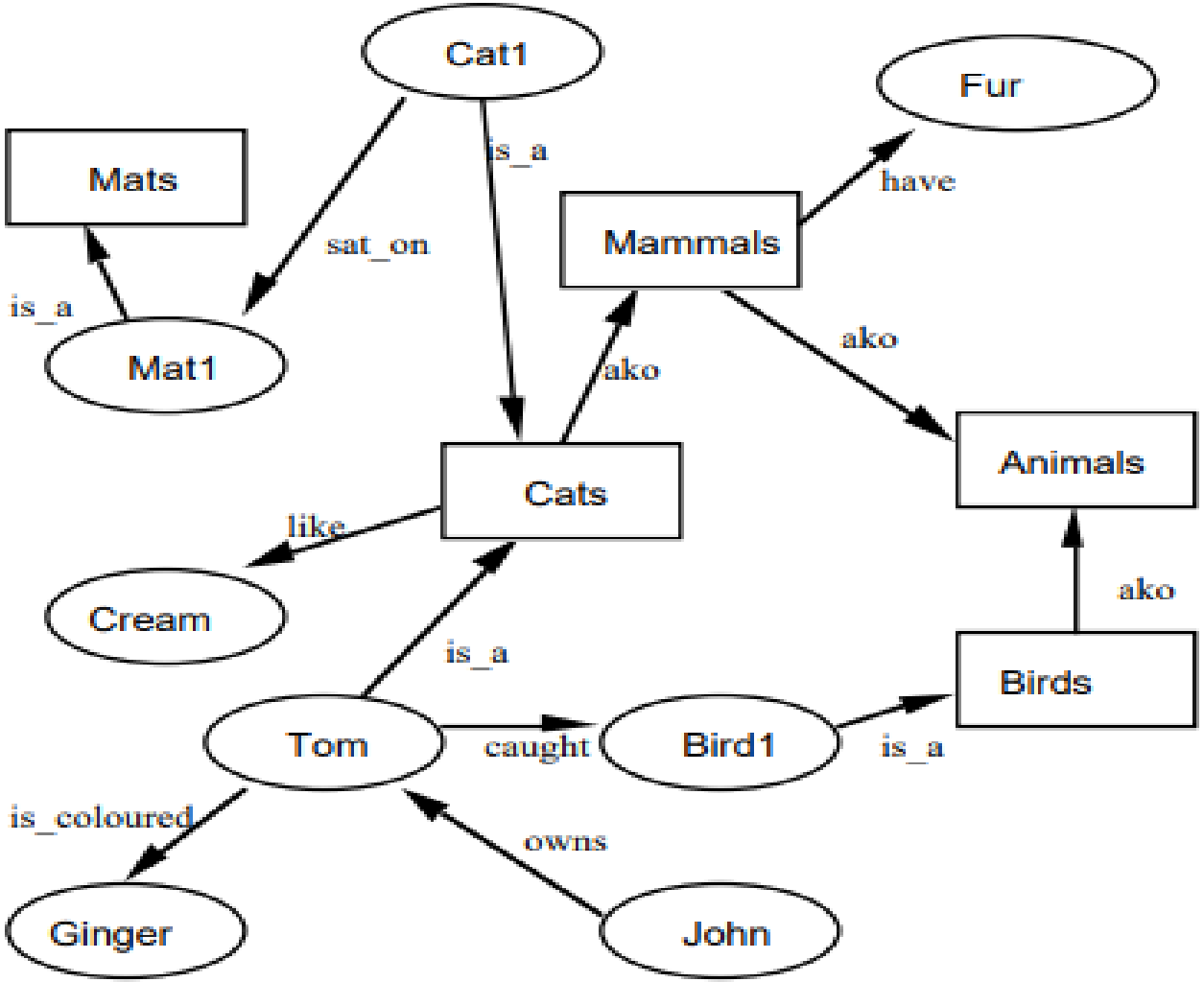
A-KIND-OF (AKO) Link

- The link AKO is used here to relate one class to another
  - AKO relates generic nodes to generic nodes while the IS-A relates an instance or individual to a generic class
  - The more general class that an AKO arrow points to is called a superclass
  - AKO points from a subclass to a class

Objects in a Class

- The objects in a class have one or more attributes in common
  - Each attribute has a value
  - The combination of attribute and value is a property
  - For example, a blimp has attributes of size, weight, shape, and color.
    - The value of the shape attribute is ellipsoidal

A Semantic Net with IS-A and A-KIND-OF (AKO) Links



Limitations of Semantic Nets

- One of the drawbacks of semantic network is that the links between the objects represent only binary relations. For example, the sentence Run(ChennaiExpress, Chennai,Bangalore,Today) cannot be asserted directly.
- There is no standard definition of link names.

Frames

What is a Frame?

- Frames are useful for simulating common-sense knowledge, which is a very difficult area for computers to master
- The basic characteristic of a frame is that it represents related knowledge about a narrow subject that has much default knowledge
- A frame system would be a good choice for describing a mechanical device, for example a car
- A frame is analogous to a record structure, corresponding to the fields and values of a record are the slots and slot fillers of a frame
- A frame is basically a group of slots and fillers that defines a stereotypical object

Flexibility in Frames

- Slots in a frame can contain
  - information for choosing a frame in a situation
  - relationship between this and other frames
  - procedures to carry out after various slots filled
  - default information to use when input is missing
  - blank slots — left blank unless required for a task
  - other frames, which gives a hierarchy

Frames vs Semantic Nets

- While semantic nets are basically a two-dimensional representation of knowledge, frames add a third dimension by allowing nodes to have structures
- The frame contrasts with the semantic net, which is generally used for broad knowledge representation
- Just as with semantic nets, there are no standards for defining frame-based systems

Example

- The car is the object, the slot name is the attribute, and the filler is the value

Slots	Fillers
manufacturer	General Motors
model	Chevrolet Caprice
year	1979
transmission	automatic
engine	gasoline
tires	4
color	blue

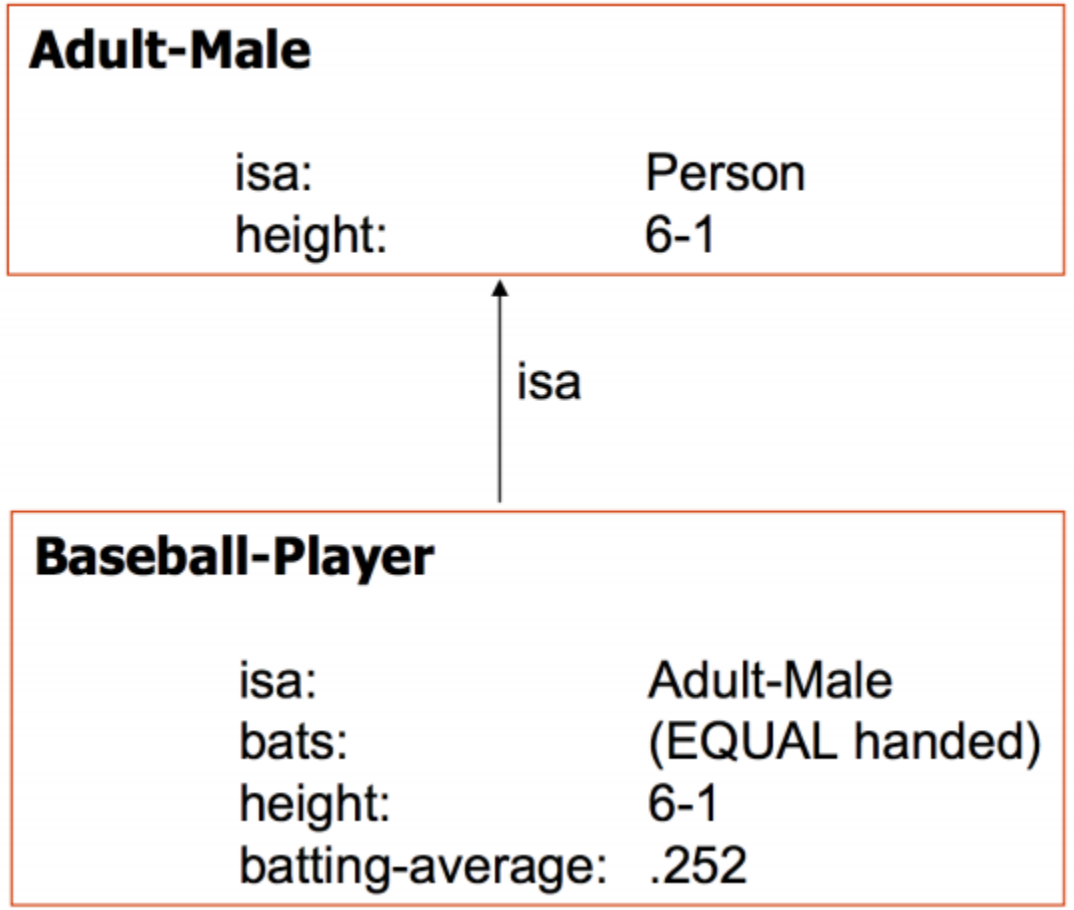
Frames

- Frames are generally designed to represent either generic or specific knowledge
- The slots may also contain procedures attached to the slots, called procedural attachments
  - The if-needed type is executed when a filler value is needed but none are initially present or the default value is not suitable
    - Defaults are often used to represent common-sense knowledge
  - The if-added type is run for procedures to be executed when a value is to be added to a slot
  - An if-removal type is run whenever a value is to be removed from a slot
- Slot fillers may also contain relations, e.g., a-kind-of and is-a relations

A Generic Frame for Property

Slots	Fillers
name	property
specialization_of	a_kind_of object
types	(car, boat, house) if-added: Procedure ADD_PROPERTY
owner	default: government if-needed: Procedure FIND_OWNER
location	(home, work, mobile)
status	(missing, poor, good)
under_warranty	(yes, no)

Example: Frame Hierarchy



Car Frame - A Generic Subframe of Property

Slots	Fillers
name	car
specialization_of	a-kind-of property
types	(sedan, sports, convertible)
manufacturer	(GM, Ford, Toyota)
location	mobile
wheels	4
transmission	(manual, automatic)
engine	(gasoline, hybrid gas/electric)

An Instance of a Car Frame

Slots	Fillers
name	John's car
specialization_of	is_a car
manufacturer	GM
owner	John Doe
transmission	automatic
engine	gasoline
status	good
under_warranty	yes

Scripts

What is a Script?

- A *script* is a structure that prescribes a set of circumstances which could be expected to follow on from one another.
- It is similar to a thought sequence or a chain of situations which could be anticipated.
- It could be considered to consist of a number of slots or frames but with more specialised roles.

Scripts are Beneficial

- Events tend to occur in known runs or patterns.
- Causal relationships between events exist.
- Entry conditions exist which allow an event to take place
- Prerequisites exist upon events taking place. *E.g.* when a student progresses through a degree scheme or when a purchaser buys a house.

Components of a Script

- Entry Conditions: These must be satisfied before events in the script can occur.
- Results: Conditions that will be true after events in script occur.
- Props: Slots representing objects involved in events.
- Roles: Persons involved in the events.
- Track: Variations on the script. Different tracks may share components of the same script.
- Scenes: The sequence of events that occur. Events are represented in conceptual dependency form.

Example: Robbing a Bank

- Getting a gun.
- Hold up a bank.
- Escape with the money.

Props

- Gun, G
- Loot, L
- Bag, B
- Get away car, C

Roles

- Robber, *S*.
- Cashier, *M*.
- Bank Manager, *O*.
- Policeman, *P*.

Entry Conditions

- *S* is poor.
- *S* is destitute.

Results

- *S* has more money.
- *O* is angry.
- *M* is in a state of shock.
- *P* is shot.

Scenes

- obtaining the gun,
- robbing the bank and
- the getaway.

Script: ROBBERY *Track: Successful Snatch*

Props:
G = Gun,
L = Loot,
B= Bag,
C = Get away car.

Roles:
R = Robber
M = Cashier
O = Bank Manager
P = Policeman.

Entry Conditions:
R is poor.
R is destitute.

Results:
R has more money.
O is angry.
M is in a state of shock.
P is shot.

Scene 1: Getting a gun

R PTRANS R into Gun Shop
R MBUILD R choice of G
R MTRANS choice.
R ATRANS buys G

(go to scene 2)

Scene 2 Holding up the bank

R PTRANS R into bank
R ATTEND eyes M, O and P
R MOVE R to M position
R GRASP G
R MOVE G to point to M
R MTRANS "Give me the money or ELSE" to M
P MTRANS "Hold it Hands Up" to R
R PROPEL shoots G
P INGEST bullet from G
M ATRANS L to M
M ATRANS L puts in bag, B
M PTRANS exit
O ATRANS raises the alarm

(go to scene 3)

Scene 3: The getaway

M PTRANS C

Advantages of Scripts

- Ability to predict events.
- A single coherent interpretation may be build up from a collection of observations.

Disadvatages of Scripts

- Less general than frames.
- May not be suitable to represent all kinds of knowledge.

Conceptual Graphs

What is a Conceptual Graph (CG)?

- CG capture nuances in natural language whilst being able to be implemented in computer software.
- CG were devised by John Sowa from philosophical, psychological, linguistic, and artificial intelligence foundations in a principled way.

Concepts and Relations

- CG are based upon the following general form:

- This may be read as: “The relation of a Concept-1 is a Concept-2”.
- The direction of the arrows assists the direction of the reading.
- If the arrows were pointing the other way, then the reading would be the same except that Concept-1 and Concept-2 would exchange places (i.e. “The relation of Concept-2 is a Concept-1”).
- As an alternative to the above ‘display’ form, the graphs may be written in the following ‘linear’ text-based form:
- [Concept_1] -> (relation) -> [Concept_2].

Another Example

- [Funding_Request] -> (initiator) -> [Employee].
- The example graph reads as “The initiator of a funding request is an employee”.
- This may create readings that may sound long-winded or ungrammatical, but is a useful mnemonic aid in constructing and interpreting any graph.
- It is easier to state “An employee initiates a funding request”

Referents

- Concepts can have referents, which refers to a particular instance, or individual, of that concept.
- For example consider the concept:
  - [Employee: Simon].
- This reads as “The employee known as Simon”.
- The referent is a conformity to the type label in a concept.
- This example shows that Simon conforms to the type label ‘Employee’.

Generic Referents

- A concept that appears without an individual referent has a generic referent.
- Such ‘generic concepts’ should be denoted as [Type_Label: *].
- Writing [Type_Label] is merely a convenient shorthand.
- Generic concepts may take up an individual referent.

Unique Identifier

- A unique identifier can be used to make a concept distinct.
- Thereby the generic concept [Funding_Request] might become [Funding_Request: #1234] with respect to [Employee: Simon].
- This would yield:
  - [Funding_Request: #1234] -> (initiator) -> [Employee: Simon].
- If there are two or more employees with the name Simon we would need to make them distinct from one another e.g. [Employee: Simon#122014].

The Type Hierarchy

- In CG, type labels belong to a type hierarchy.
- Thereby: Manager < Employee. (“A manager is an employee”)
- This means Manager is a more specialised type of the type Employee i.e. Manager is a subtype of Employee.
- Similarly, the remainder of the hierarchy may be:
  - Employee < Person.
  - Person < Animal.
  - Animal < Entity.
  - Entity < T.
- The type denoted as ‘T’ means the universal supertype. It has no supertypes and is therefore the most general type.

Supertypes

- A subtype can have more than one immediate supertype.
- For example, consider the concept [Animal], which has a more detailed set of supertypes than indicated above.
- This concept has a type label Animal which may be defined as:
  - Animal < Animate, Mobile_Entity, Physical_Object, ¬Machine.
  - Animate < Entity.
  - Mobile_Entity < Entity.
  - Physical_Object < Entity.
- An animal therefore inherits the characteristics of being animate, a mobile-entity, and a physical object, but it is not a machine (¬ means ‘not’).

Conceptual Dependencies

Conceptual Dependency Theory

- Conceptual Dependency originally developed to represent knowledge acquired from natural language input.
- The goals of this theory are:
  - To help in the drawing of inference from sentences.
  - To be independent of the words used in the original input.
    - That is to say: *For any 2 (or more) sentences that are identical in meaning there should be only one representation of that meaning.*

Representation

- Sentences are represented as a series of diagrams depicting actions using both abstract and real physical situations.
- The agent and the objects are represented.
- The actions are built up from a set of primitive acts which can be modified by tense.

Primitives

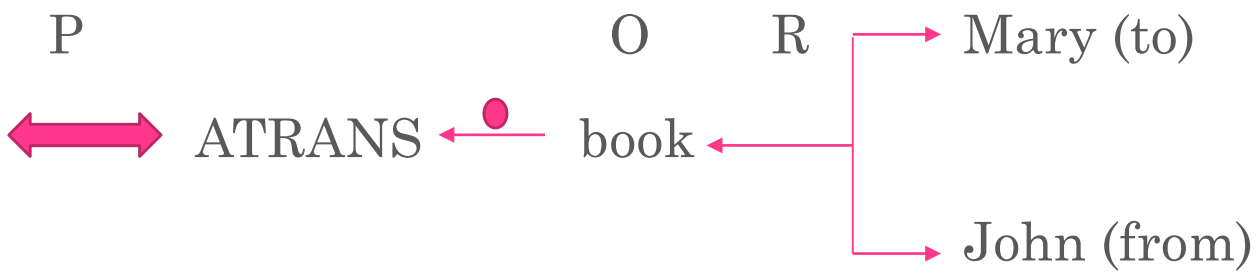
- ATRANS: Transfer of an abstract relationship. e.g. give.
- PTRANS: Transfer of the physical location of an object. e.g. go.
- PROPEL: Application of a physical force to an object. e.g. push.
- MTRANS: Transfer of mental information. e.g. tell.
- MBUILD: Construct new information from old. e.g. decide.
- SPEAK: Utter a sound. e.g. say.
- ATTEND: Focus a sense on a stimulus. e.g. listen, watch.
- MOVE: Movement of a body part by owner. e.g. punch, kick.
- GRASP: Actor grasping an object. e.g. clutch.
- INGEST: Actor ingesting an object. e.g. eat.
- EXPEL: Actor getting rid of an object from body. e.g. ????.

Primitive Conceptual Categories

- PP: Real world objects.
- ACT: Real world actions.
- PA: Attributes of objects.
- AA: Attributes of actions.
- T: Times.
- LOC: Locations.

How do we connect these things together?

- Example: John gave Mary a book.



- Arrows indicate the direction of dependency.
- Double arrows indicate two-way relation between the actor (PP) and action (ACT).

Relationships

- Letters above indicate certain relationships:
- O: object-case relation
- R: recipient-case relation
- P: past tense
- I: instrument e.g. eat with a spoon.
- D: destination e.g. going home.

Tenses

- P: Past
- F: Future
- T: Transition
- Ts: Start transition
- Tf: Finished transition
- K: Continuing
- ?: Interrogative
- /: Negative
- Nil: Present
- Delta: Timeless
- C: Conditional
- The absence of any modifier implies the present tense.