Semantic Networks, Frames, Scripts and Rules

Semantic Networks

Network Knowledge Representation

- “Traditional” knowledge representation is formal logic
- Network knowledge representation originated in 1960s with psychologists and linguists
- Observations of recall time for concepts
- Closely related concepts were recalled faster than more distantly related

A. Collins and M.R. Quillian (1969) Retrieval time from semantic memory, *Journal of Verbal Learning and Verbal Behaviour* *:240-247

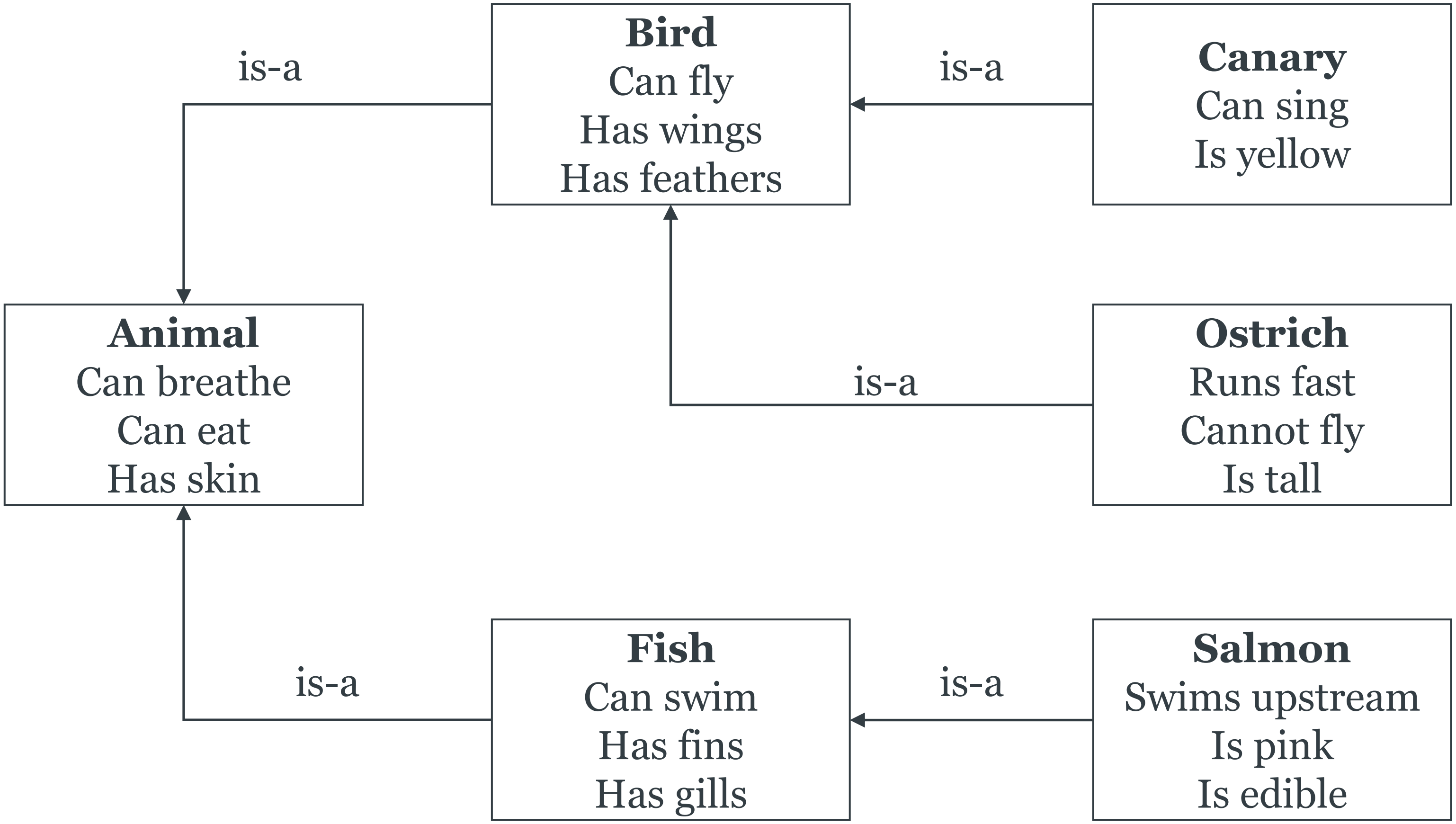
Semantic Networks

- A semantic network is a structure for representing knowledge as a pattern of interconnected nodes and arcs
- Nodes in the net represent concepts of entities, attributes, events, values
- Arcs in the network represent relationships that hold between the concepts

Semantic Networks

- Semantic networks can show inheritance
  - Relationship types – is-a, has-a
- Semantic Nets - visual representation of relationships
- Can be combined with other representation methods

Semantic Networks



Bird
Can fly
Has wings
Has feathers

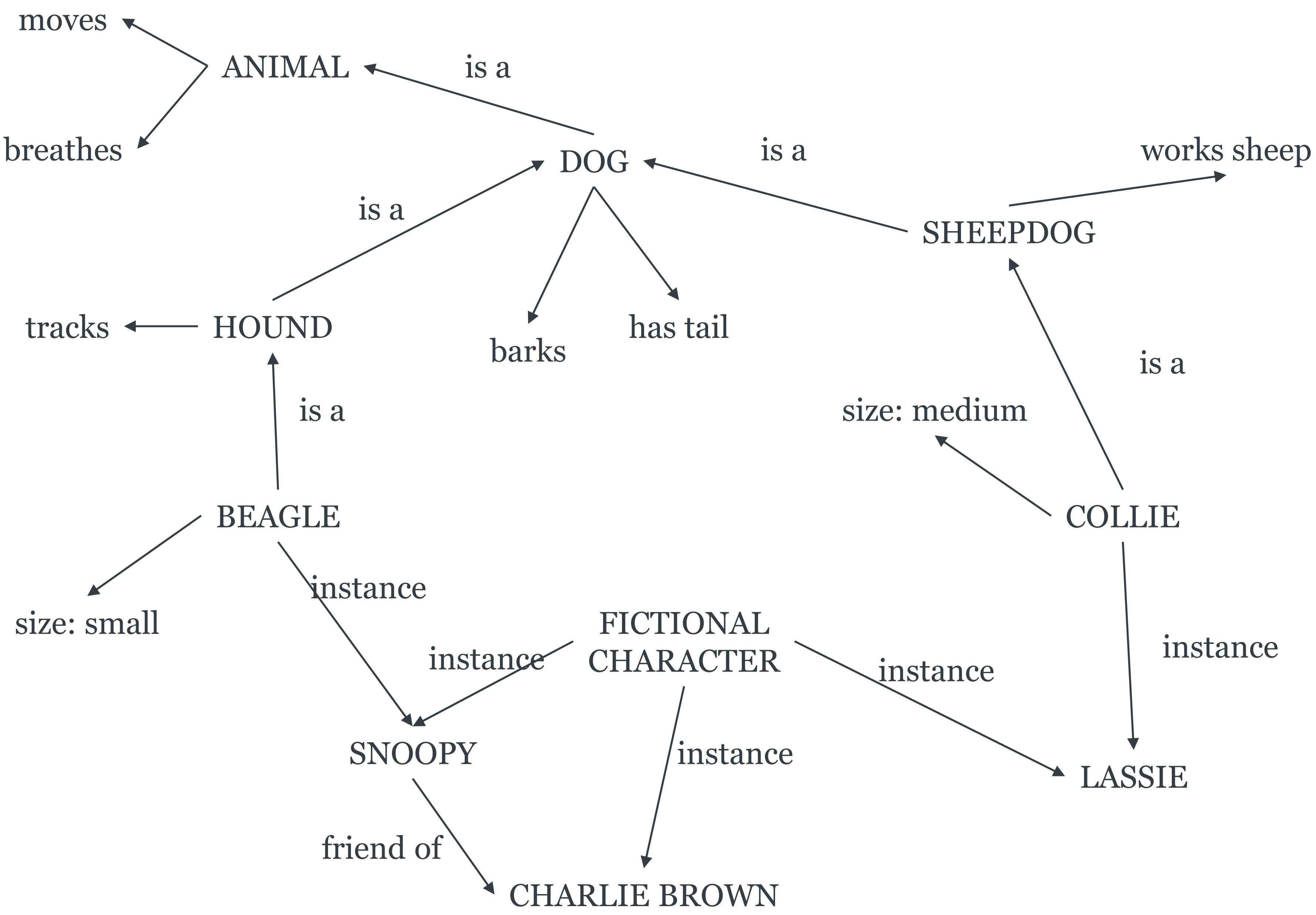
Canary
Can sing
Is yellow

Animal
Can breathe
Can eat
Has skin

Ostrich
Runs fast
Cannot fly
Is tall

Fish
Can swim
Has fins
Has gills

Salmon
Swims upstream
Is pink
Is edible



moves
ANIMAL
is a
breathes
DOG
is a
works sheep
is a
SHEEPDOG
tracks
HOUND
barks
has tail
is a
is a
size: medium
BEAGLE
COLLIE
instance
size: small
FICTIONAL CHARACTER
instance
instance
instance
SNOOPY
instance
LASSIE
friend of
CHARLIE BROWN

Semantic Networks

- What does or should a node represent?
  - A class of objects?
  - An instance of an class?
  - The canonical instance of a class?
  - The set of all instances of a class?

Semantic Networks

- Semantics of links that define new objects and links that relate existing objects, particularly those dealing with ‘intrinsic’ characteristics of a given object
- How does one deal with the problems of comparison between objects (or classes of objects) through their attributes?
  - Essentially the problem of comparing object instances
- What mechanisms are there are to handle quantification in semantic network formalisms?

Transitive inference, but…

- Clyde is an elephant, an elephant is a mammal: Clyde is a mammal.

- The US President is elected every 4 years, Bush is US President: Bush is elected every 4 years
- My car is a Ford, Ford is a car company: my car is a car company

Network knowledge representation

- Many types of network KR
  - Conceptual Graphs (Sowa)
  - Semantic Networks (Quillian)
  - Conceptual Dependency Theory (Schank)
  - (and the Semantic Web…)
- Close correspondence with other KR techniques (logic, frames, scripts, etc)
  - A different way of viewing knowledge

Frames

Frames

- A *frame* is a knowledge representation formalism based on the idea of a frame of reference.
- A frame is a data structure that includes all the knowledge about a particular object
- Frames organised in a hierarchy Form of object-oriented programming for AI and ES.
- Each frame describes one object
- Special terminology

M. Minsky (1974) A Framework for Representing Knowledge,
MIT-AI Laboratory Memo 306

Frames

- There are two types of frame:
  - Class Frame
  - Individual or Instance Frame
- A frame carries with it a set of *slots* that can represent objects that are normally associated with a subject of the frame.

Frames

- The slots can then point to other slots or frames. That gives frame systems the ability to carry out inheritance and simple kinds of data manipulation.
- The use of procedures - also called *demons* in the literature - helps in the incorporation of substantial amounts of procedural knowledge into a particular frame-oriented knowledge base

Frame-based model of semantic memory

- Knowledge is **organised** in a data structure
- Slots in structure are instantiated with particular values for a given instance of data
- …translation to OO terminology:
  - frames == classes or objects
  - slots == variables/methods

General Knowledge as Frames

DOG

Fixed
- legs: 4

Default
- diet: carnivorous
- sound: bark

Variable
- size:
- colour:

COLLIE

Fixed
- breed of: DOG
- type: sheepdog

Default
- size: 65cm

Variable
- colour:

```
MAMMAL:
    subclass: ANIMAL
    has_part:     head

ELEPHANT
    subclass: MAMMAL
    colour:       grey
    size:         large

Nellie
    instance: ELEPHANT
    likes:    apples
```

Logic underlies Frames

- $\forall x\ \mathrm{mammal}(x) \Rightarrow \mathrm{has\_part}(x, \mathrm{head})$
- $\forall x\ \mathrm{elephant}(x) \Rightarrow \mathrm{mammal}(x)$

- $\mathrm{elephant}(\mathrm{clyde})$
  $\therefore$
  $\mathrm{mammal}(\mathrm{clyde})$
  $\mathrm{has\_part}(\mathrm{clyde}, \mathrm{head})$

```
MAMMAL:
    subclass: ANIMAL
    has_part:    head
    *furry:          yes

ELEPHANT
    subclass: MAMMAL
    has_trunk:   yes
    *colour:  grey
    *size:    large
    *furry:      no

Clyde
    instance: ELEPHANT
    colour:      pink
    owner:       Fred

Nellie
    instance: ELEPHANT
    size:        small
```

- Can represent subclass and instance relationships (both sometimes called ISA or “is a”)
- Properties (e.g. colour and size) can be referred to as slots and slot values (e.g. grey, large) as slot fillers
- Objects can inherit all properties of parent class (therefore Nellie is grey and large)
- But can inherit properties which are only typical (usually called default, here starred), and can be overridden
- For example, mammal is typically furry, but this is not so for an elephant

Multiple Inheritance

- Situation can be complicated by multiple inheritance, where object or class may have more than one parent class.
- May result in some conflict: for example if Nellie is both an elephant and a circus animal
- From elephant we would expect Nellie’s habitat to be the jungle, but from circus animal we would expect it to be a tent
- Could set further precedence order to resolve this – or might need further class for Circus-elephant

The Nixon Diamond

- All Quakers are pacifists
- All Republicans are not pacifists
- Nixon is a Republican
- Nixon is a Quaker

R. Reiter and G. Criscuolo (1981). On interacting defaults. In Proceedings of the Seventh International Joint Conference on Artificial Intelligence (IJCAI'81), pages 94-100.

Prototypical Situations

- Many situations are prototypical
- Prototypical situations share a common set of attributes
- We can use the powerful inheritance capabilities of the frames representation to help represent these situations

- Provide a concise, structural representation of knowledge in a natural manner
- Frame encompasses complex objects, entire situations or a management problem as a single entity
- Frame knowledge is partitioned into slots
- Slot can describe declarative knowledge or procedural knowledge
- Hierarchy of Frames: Inheritance

Capabilities of Frames

- Ability to clearly document information about a domain model; for example, a plant's machines and their associated attributes
- Related ability to constrain allowable values of an attribute
- Modularity of information, permitting ease of system expansion and maintenance
- More readable and consistent syntax for referencing domain objects in the rules

Capabilities of Frames

- Platform for building graphic interface with object graphics
- Mechanism to restrict the scope of facts considered during forward or backward chaining
- Access to a mechanism that supports the inheritance of information down a class hierarchy
- Used as underlying model in standards for accessing KBs (Open Knowledge Base Connectivity - OKBC)

Summary

- Frames have been used in conjunction with other, less well-grounded, representation formalisms, like production systems, when used to build to pre-operational or operational expert systems
- Frames cannot be used efficiently to organise ‘a whole computation

Scripts

Scripts

- Knowledge representation scheme describing a stereotypical sequence of events, and goals and plans of actors concerned
- Expectation-driven programming
- Allows inference about implicit cause and effect relationships when interpreting situations
- Builds on Conceptual Dependency Theory

R.C. Schank and R. Abelson (1977) Scripts, Plans, Goals and Understanding

Script Elements

- Entry Conditions
  - What are the descriptors of the world that must be true for the script to be called?
- Props
  - What objects make up the content of the script?
- Roles
  - What actions are performed by the participants in the script?
- Scenes
  - Temporal decomposition of script into meaningful episodes
- Results
  - What are the outcomes following termination of the script?

Script: RESTAURANT

Props:
Tables
Menu
F = Food
Bill
Money

Roles:
S = Customer
W = Waiter
C = Chef
M = Cashier
O = Owner

Entry conditions:
S is hungry
S has money

Results:
S has less money
O has more money
S is not hungry
S is pleased (optional)

Scenes:
Entering
Ordering
Eating
Exiting

Scene 1: Entering

S PTRANS S into restaurant
A ATTEND eyes to tables
S MBUILD where to sit
S PTRANS S to table
S MOVE S to sitting position

Scene 2: Ordering (Menu on table)

S PTRANS menu to S
S MTRANS food list to CP(S)
S MBUILD choice of F
S MTRANS signal to W
W PTRANS W to table
S MTRANS ‘I want F’ to W
W PTRANS W to C
W MTRANS (ATRANS F) to C
C DO (prepare F script)
…

Scripts and Natural Language Understanding

- Scripts are used to organise a knowledge base in terms of the situations that are to be understood
- “Jack went to a restaurant. He decided to order steak. He sat there and waited for a long time. Finally, he got angry and left”
  - What was Jack waiting for?
  - Why did he get angry?
- Effective in limited domains, can be inflexible in more general domains

Production Rules

Production Rules

- Condition-Action Pairs
  - IF this condition (or premise or antecedent) occurs, THEN some action (or result, or conclusion, or consequence) will (or should) occur
  - IF the traffic light is red AND you have stopped, THEN a right turn is OK

Production Rules

- Each production rule in a knowledge base represents an autonomous chunk of expertise
- When combined and fed to the inference engine, the set of rules behaves synergistically
- Rules can be viewed as a simulation of the cognitive behaviour of human experts
- Rules represent a model of actual human behaviour
- Predominant technique used in expert systems, often in conjunction with frames

Forms of Rules

- IF premise, THEN conclusion
  - IF your income is high, THEN your chance of being audited by the Inland Revenue is high
- Conclusion, IF premise
  - Your chance of being audited is high, IF your income is high

Forms of Rules

- Inclusion of ELSE
  - IF your income is high, OR your deductions are unusual, THEN your chance of being audited is high, OR ELSE your chance of being audited is low
- More complex rules
  - IF credit rating is high AND salary is more than £30,000, OR assets are more than £75,000, AND pay history is not "poor,"
    THEN approve a loan up to £10,000, and list the loan in category "B.”
- Action part may have more information: THEN "approve the loan" and "refer to an agent"

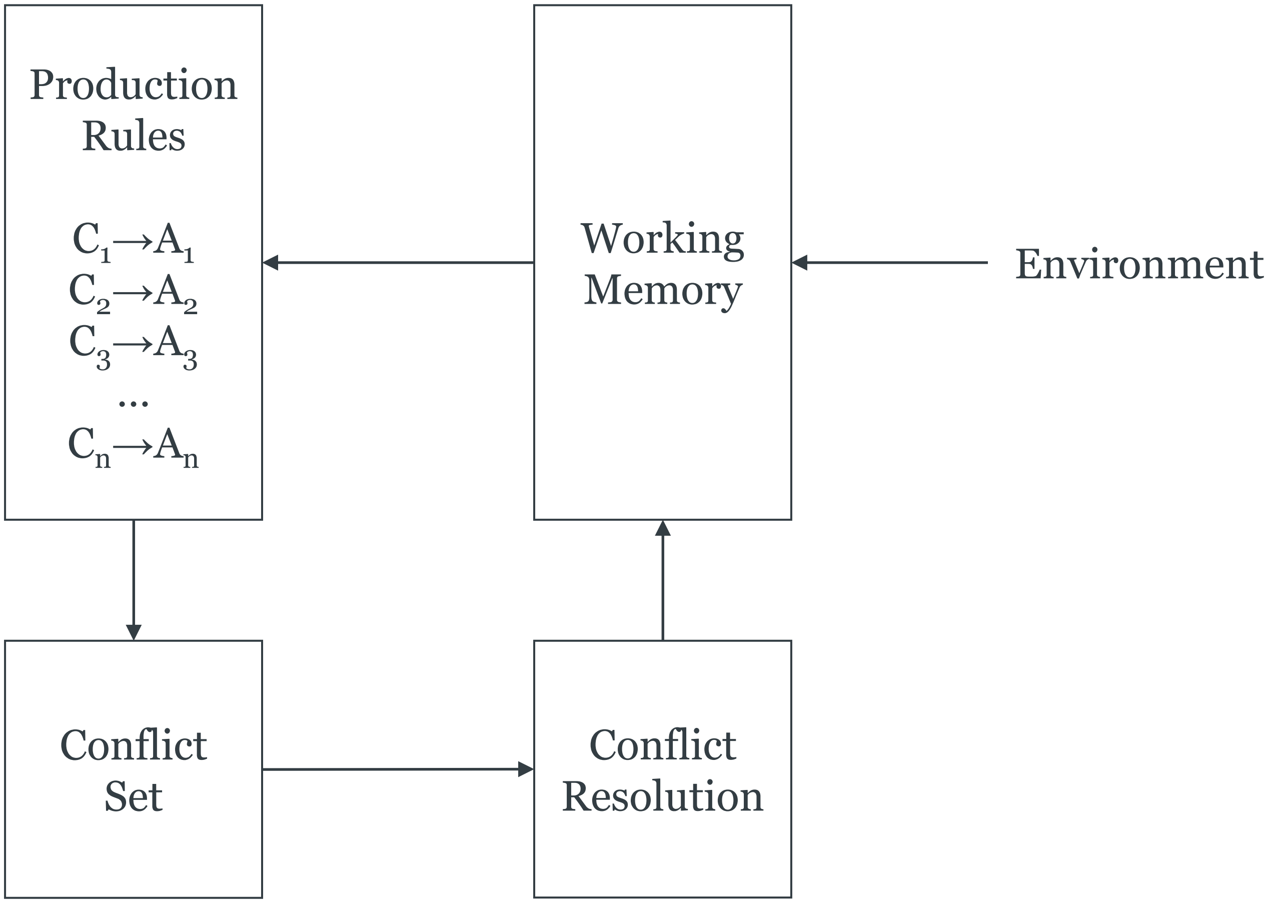
Characteristics of Rules

	First Part	**Second Part**
Names	Premise Antecedent Situation IF	Conclusion Consequence Action THEN
Nature	Conditions, similar to declarative knowledge	Resolutions, similar to procedural knowledge
Size	Can have many IFs	Usually only one conclusion
Statement	AND statements	All conditions must be true for a conclusion to be true
	OR statements	If any condition is true, the conclusion is true

Rule-based Inference

- Production rules are typically used as part of a *production system*
- Production systems provide pattern-directed control of the reasoning process
- Production systems have:
  - Productions: set of production rules
  - Working Memory (WM): description of current state of the world
  - Recognise-act cycle

Production Systems



Production Rules

$C_1 \rightarrow A_1$
$C_2 \rightarrow A_2$
$C_3 \rightarrow A_3$
…
$C_n \rightarrow A_n$

Working Memory

Environment

Conflict Set

Conflict Resolution

Recognise-Act Cycle

- Patterns in WM matched against production rule conditions
- Matching (activated) rules form the **conflict set**
- One of the matching rules is selected (conflict resolution) and **fired**
  - Action of rule is performed
  - Contents of WM updated
- Cycle repeats with updated WM

Conflict Resolution

- Reasoning in a production system can be viewed as a type of search
  - Selection strategy for rules from the conflict set controls search
- Production system maintains the conflict set as an **agenda**
  - Ordered list of **activated rules** (those with their conditions satisfied) which have not yet been executed
  - Conflict resolution strategy determines where a newly-activated rule is inserted

Salience

- Rules may be given a precedence order by assigning a **salience value**
- Newly activated rules are placed in the agenda above all rules of lower salience, and below all rules with higher salience
  - Rule with higher salience are executed first
- Conflict resolution strategy applies between rules of the same salience
- If salience and the conflict resolution strategy can' t determine which rule is to be executed next, a rule is chosen at random from the most highly ranked rules

Conflict Resolution Strategies

- Depth-first: newly activated rules placed above other rules in the agenda
- Breadth-first: newly activated rules placed below other rules
- Specificity: rules ordered by the number of conditions in the LHS (simple-first or complex-first)
- Least recently fired: fire the rule that was last fired the longest time ago
- Refraction: don’t fire a rule unless the WM patterns that match its conditions have been modified
- Recency: rules ordered by the timestamps on the facts that match their conditions

Salience

- Salience facilitates the modularisation of expert systems in which modules work at different levels of abstraction
- Over-use of salience can complicate a system
  - Explicit ordering to rule execution
  - Makes behaviour of modified systems less predictable
- Rule of thumb: if two rules have the same salience, are in the same module, and are activated concurrently, then the order in which they are executed should not matter

Common Types of Rules

- Knowledge rules, or declarative rules, state all the facts and relationships about a problem
- Inference rules, or procedural rules, advise on how to solve a problem, given that certain facts are known
- Inference rules contain rules about rules (metarules)
- Knowledge rules are stored in the knowledge base
- Inference rules become part of the inference engine

Major Advantages of Rules

- Easy to understand (natural form of knowledge)
- Easy to derive inference and explanations
- Easy to modify and maintain
- Easy to combine with uncertainty
- Rules are frequently independent

Major Limitations of Rules

- Complex knowledge requires many rules
- Builders like rules (hammer syndrome)
- Search limitations in systems with many rules