Study Material

Table of Contents

Module No.	Module Name	Content
Module 2	Knowledge Representation Scheme and Reasoning Concept	Semantic Networks

1. Introduction to Knowledge Representation and Semantic Networks

Knowledge representation (KR) is a critical area in both cognitive science and artificial intelligence (AI). In cognitive science, KR explores how people store and process information, while in AI, its main objective is to store knowledge so that programs can process it to achieve human-like intelligence.

There are various ways to represent knowledge, including predicate logic, semantic networks, extended semantic nets, frames, and conceptual dependency. Today, we will focus on **semantic networks**.

A **semantic network**, or semantic net, is a **graph structure used for representing knowledge** about objects, people, concepts, and the specific relationships between them. Its syntax is straightforward: it is composed of **labeled nodes and links**, forming a directed graph.

- **Nodes** correspond to concepts, facts, or objects.
- **Arcs** (or links) represent relations or associations between two concepts.

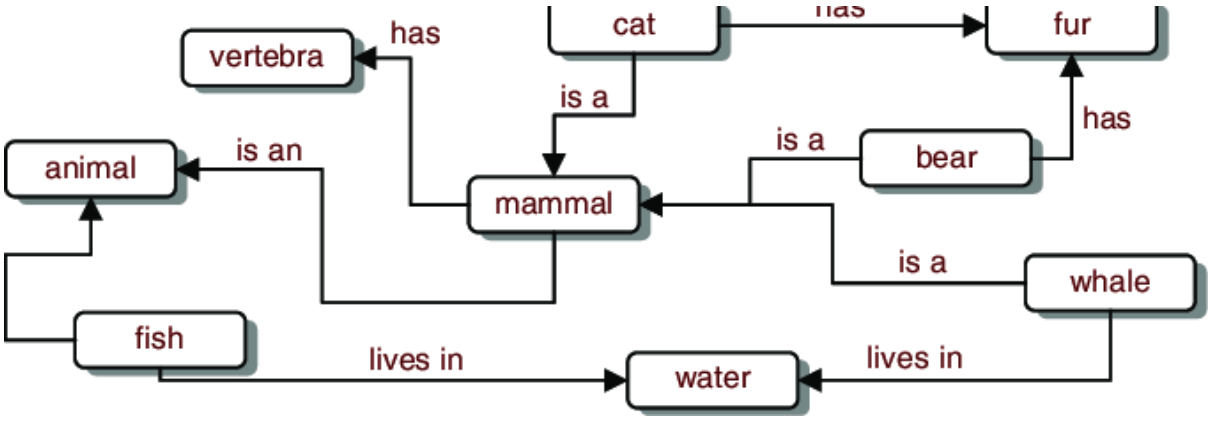
Semantic networks have a long history, with earlier versions used in philosophy, psychology, and linguistics, predating their computer implementations in AI and machine translation. A modern example is the **Giant Global Graph of the Semantic Web**, which is a large semantic network.

Semantic networks offer a **declarative graphic representation** that can both represent knowledge and support automated reasoning systems.

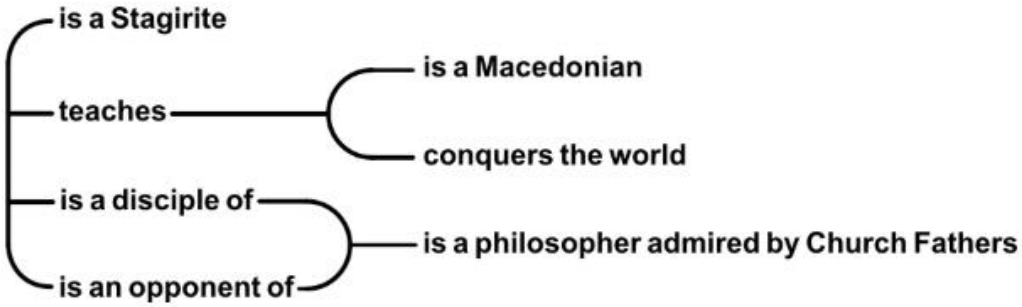
2. Types of Semantic Networks

The sources identify six common kinds of semantic networks, each with a distinct emphasis:

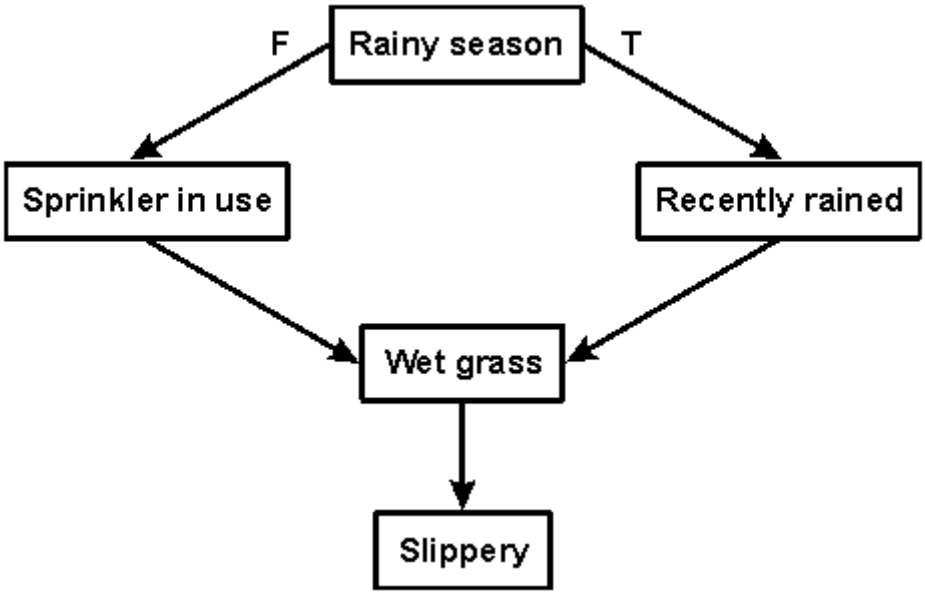
1. **Definitional Networks**: Emphasize the subtype or is-a relation, supporting inheritance.

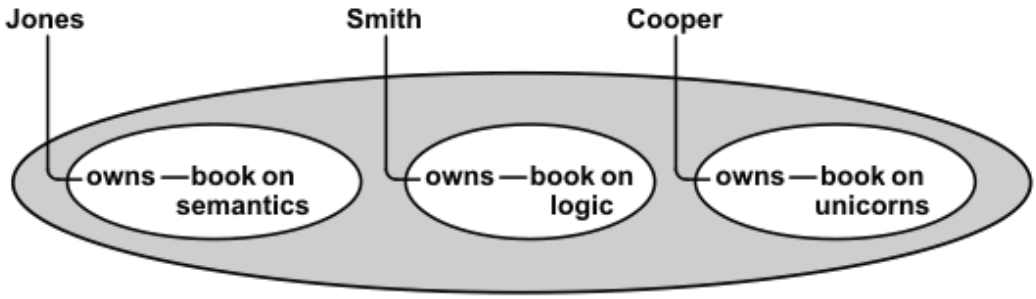


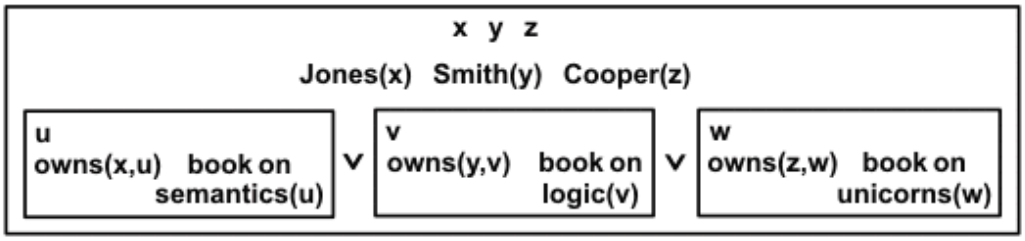
2. **Assertional Networks**: Designed to assert propositions, often assumed to be contingently true.



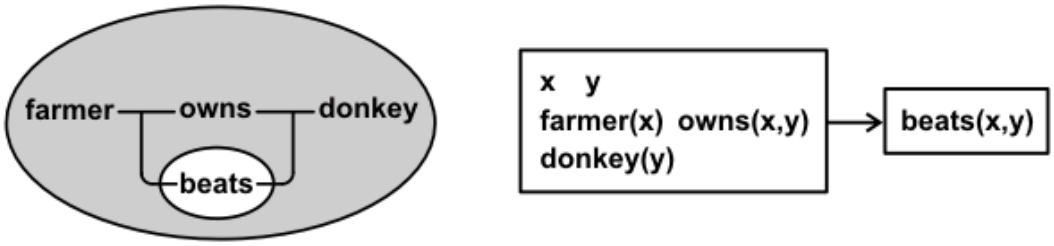
3. **Implicational Networks**: Use implication as the primary relation, representing beliefs, causality, or inferences.



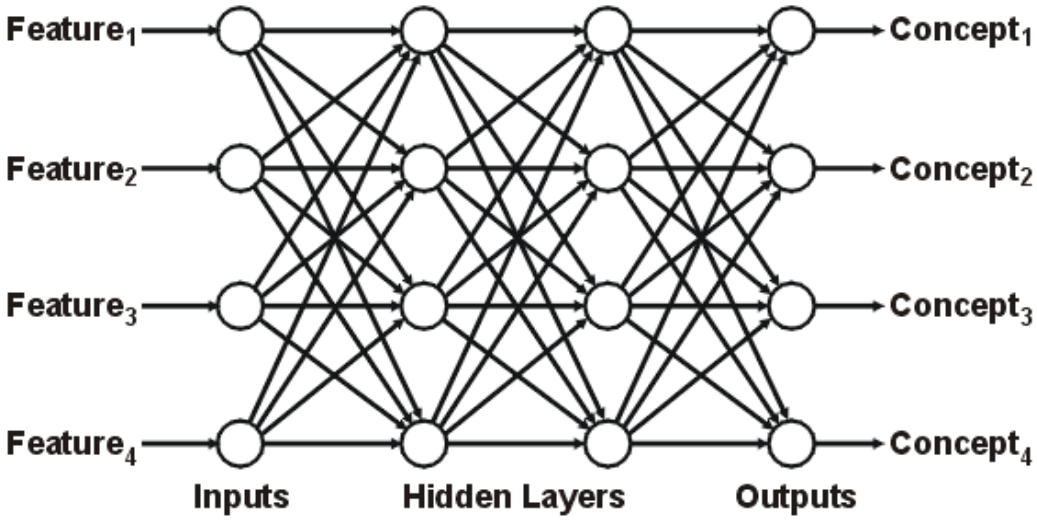




4. **Executable Networks**: Include mechanisms like message passing or attached procedures to perform inferences.



5. **Learning Networks**: Build or extend representations by acquiring knowledge from examples, modifying network structure or numerical weights.



6. **Hybrid Networks**: Combine two or more of the above techniques.

We will explore the first two types in more detail, as they lay foundational understanding, and then briefly touch upon the others.

3. Definitional Networks

Definitional networks prioritize the **subtype or is-a relationship** between a concept type and a newly defined subtype. They form a **generalization or subsumption hierarchy** that enables the powerful rule of **inheritance**.

Inheritance is a mechanism that allows knowledge to be stored at the highest possible level of abstraction, reducing the size of the knowledge base. It facilitates the inference of information

and is a natural tool for representing taxonomically structured information. Through inheritance, all members and sub-concepts of a concept share common properties. For example, properties attached to a class are inherited by all its subclasses and members.

- **Example**: If "Living_thing" has the property "breathe" and "Human" is an "isa" (subclass) of "Living_thing", then "Human" inherits the property "breathe".

Historically, the earliest known semantic network was the **Tree of Porphyry**, drawn in the 3rd century AD. This tree illustrates Aristotle's method of defining categories by specifying a genus (general type) and differentiae (distinguishing features for subtypes).

- **Illustration**: Substance with material differentia is Body; with immaterial differentia is Spirit. LivingThing inherits material Substance from Body and adds animate. Human inherits sensitive animate material Substance and adds rational. This directly reflects the concept of inheritance.

In modern AI, **description logics**, such as those derived from **KL-ONE**, embody these principles. KL-ONE defines concepts like Truck and TrailerTruck as subtypes of Vehicle.

- **Nodes**: Generic concepts (e.g., Truck), individual concepts (e.g., specific instance 18), and primitive types (e.g., Integer).
- **Links**:
    - **Double-line arrows**: subtype-supertype links (e.g., TrailerTruck to Truck).
    - **Arrows with a circle**: Represent roles (e.g., Truck has UnloadedWt).
    - **v/r notation**: Indicates value restrictions or type constraints on role values (e.g., NumberOfWheels restricted to 18 for TrailerTruck).

Most definitional networks, like the Tree of Porphyry and KL-ONE, are subsets of classical **first-order logic (FOL)** and belong to **monotonic logics**, meaning new information only adds to provable theorems without deleting or modifying old information.

Challenges can arise with **nonmonotonic reasoning**, especially with **conflicting defaults**. The "Nixon diamond" example shows how Quakers are by default pacifists and Republicans are by default not pacifists, leading to a conflict for Nixon who is both. Similarly, a RoyalElephant might override the default gray color for Elephant. Modern systems often use systematic methods for **belief revision** to ensure consistency.

Description Logic OWL is widely used in the Semantic Web for knowledge representation, with the EL subset aligning with Aristotle's syllogisms.

4. Assertional Networks

Assertional networks are designed to **assert propositions**, and the information within them is typically assumed to be **contingently true**, unless explicitly marked otherwise. These networks can model conceptual structures underlying natural language semantics.

Historically, **Gottlob Frege** (1879) developed a tree notation for first-order logic, the Begriffsschrift. **Charles Sanders Peirce** (1880, 1885) developed an algebraic notation and later a graphic notation known as **existential graphs (EG)**. **Hans Kamp** (1981) later invented **discourse representation structures (DRS)** for natural language semantics, which are formally equivalent to Peirce's EGs.

- **Initial Limitations**: Early relational graphs (and dependency graphs) were limited to representing only **conjunction ($\land$)** and the **existential quantifier ($\exists$)**. This is similar to the expressive power of **RDF** in the Semantic Web, where blank nodes correspond to existential quantifiers.
- **Overcoming Limitations**: Peirce's "brilliant discovery" in 1897 was introducing an **oval** to enclose and negate a graph or subgraph. This allowed combinations of ovals with conjunction and existential quantifiers to express all logical operators (~, ∨, ⊃, ∀).
    - **Example**: The implication "If a farmer owns a donkey, then he beats it" can be represented with a nest of two ovals in EG or two boxes linked by arrows in DRS. These translate to a formula using universal quantifiers and implication: (∀x)(∀y)((farmer(x) ∧ donkey(y) ∧ owns(x,y)) ⊃ beats(x,y)).
    - **Example (Disjunction)**: "Either Jones owns a book on semantics, Smith owns a book on logic, or Cooper owns a book on unicorns" can be represented in both EG and DRS, mapping to a formula with existential quantifiers and disjunction.

In linguistics, **Lucien Tesnière** (1959) developed **dependency grammar** graph notations. Influenced by this, **Roger Schank** (1975) shifted the emphasis to **conceptual dependencies (CD)**, using primitive acts and conceptual categories to represent sentence meanings.

- **Conceptual Dependency Theory**: Developed by Schank (1973-1975) to represent the meaning of natural language sentences. It helps in drawing inferences and is language-independent.
- **Conceptual Primitives**: CD uses a specific set of primitives rather than words to build representations. Examples include:
  - **ATRANS**: Transfer of an abstract relationship (e.g., give)
  - **PTRANS**: Transfer of physical location (e.g., go)
  - **INGEST**: Ingesting an object (e.g., eat)
  - **MTRANS**: Transfer of mental information (e.g., tell)
- **Conceptual Categories**:
  - **ACT**: Actions (CD primitives)
  - **PP**: Objects (picture producers)
  - **AA**: Modifiers of actions (action aiders)
  - **PA**: Modifiers of PPs (picture aiders)
- **CD Representation Rules**:
  - **Rule 1**: PP <=> ACT (actor causes an event), e.g., "John ran" -> John <=> PTRANS.
  - **Rule 5**: PP <=> PA (PP described by a PA), e.g., "John is fat" -> John <=> weight (> 80).
- **Inferences with Primitives**: General inferences are stored with each primitive act. For example, from "John killed Mike", one can infer "Mike is dead". For INGEST, inferences include: object no longer in original form, actor less hungry (if edible), actor's health bad (if toxic), PTRANS inferred for object's position change. For ATRANS, inferences about change of ownership.
- **Problems with CD**: It's difficult to reconstruct original sentences from CD representations, representations can be complex, and finding the correct primitive for verbs can be challenging.

Propositional semantic networks emerged in the 1970s to correctly represent the **scope of logical operators**, using explicit nodes for propositions.

- **SNePS (Semantic Network Processing System)** by Stuart Shapiro (1971): Uses nodes (e.g., M1-M5) to represent distinct propositions, with relational content attached to them. This allows making metalevel statements about other propositions.
- **Conceptual Graphs (CGs)** by John F. Sowa (1976): Combine linguistic features of dependency graphs and logical features of existential graphs. In CGs, concept boxes (e.g., [Farmer]) represent restricted quantifiers (e.g., ∃x:Farmer), unlike EG's general existential quantifiers. CGs also use nesting to represent propositions and their relations.

5. Other Types of Networks and Conclusion

5.1. Implicational Networks

These networks use **implication** as their primary relation. They can be interpreted as **belief networks, causal networks, Bayesian networks, or truth-maintenance systems (TMS)**.

- **Example**: An implicational network for "slippery grass" shows propositions like "Rainy season" implying "Recently rained", which implies "Grass is wet", which implies "Grass is slippery".
- **Reasoning**: Systems like TMS propagate truth values (logic-based) or probabilities (Bayesian networks) through the network, allowing for forward and backward reasoning to deduce new information or verify consistency.
- **Gentzen's clause form** for implications ($p_1, ..., p_n \Rightarrow q_1, ..., q_m$) shows how conjunctions of antecedents imply disjunctions of consequents, providing a basis for propositional logic.

5.2. Executable Networks

These networks contain **mechanisms that cause changes within the network itself**, distinguishing them from static data structures.

- **Mechanisms**:
    1. **Message passing**: Nodes pass data (markers, tokens, numeric weights, or messages).
    2. **Attached procedures**: Programs associated with a node perform actions or computations.
    3. **Graph transformations**: Combine, modify, or break graphs.
- **Historical influence**: Otto Selz's concept of **schematic anticipation** (filling empty slots in a pattern) influenced AI pioneers like Herbert Simon and Allen Newell. **Ross Quillian's marker passing algorithm** (spreading activations) was a significant innovation, adopted by later systems.
- **Examples**:
    - **Dataflow graphs**: Passive nodes hold data, active nodes perform functions. They can form complete programming languages when extended with conditions and recursion.

- **Petri nets**: Combine marker passing with procedures, using places (passive nodes with tokens) and transitions (active nodes that fire based on tokens). They model preconditions and postconditions and are useful for distributed parallel processes.

5.3. Learning Networks

Learning systems modify internal representations to respond more effectively to their environment. Networks can be modified in three ways:

1. **Rote memory**: Convert new information to a network and add it.
2. **Changing weights**: Modify numerical values associated with nodes and arcs, often representing probabilities. This is central to **neural nets**, where the structure is fixed, and learning involves adjusting weights through methods like backpropagation.
3. **Restructuring**: Make fundamental changes to the network structure itself. Examples include:
    - **Winston's system**: Generalized relational graphs from positive and negative examples.
    - **NanoKlaus**: Learned definitional networks through dialogue.
    - **Case-based reasoning**: Stores cases and associated information, using restructuring to find common patterns for indexing and similarity measures.
    - **Game playing (Levinson)**: Combined rote learning, restructuring for generalizations, similarity measures, and backpropagation for evaluating positions.

5.4. Hybrid Networks

Most complex AI systems are hybrids. For knowledge representation, the **Krypton system** combined KL-ONE for defining terms (T-Box) with a theorem prover for FOL assertions (A-Box). **UML (Unified Modeling Language)** is another widely used hybrid, integrating definitional networks (for object types and inheritance), executable networks (state charts, activity diagrams similar to Petri nets), and relational graphs (for metadata). UML's **Object Constraint Language (OCL)** is a form of first-order logic for specifying assertions.

5.5. Graphic vs. Linear Notations

While graphic and linear notations can express equivalent information, they have different advantages.

- **Graphic notations** (like semantic networks) excel at showing **direct connections**. They offer **human readability** and **heuristic value**, helping users discover patterns and insights that are difficult to see in linear forms. Peirce's existential graphs, for instance, provided "direct insight" into the structure of proofs and simplified Gentzen's rules of natural deduction.
- **Linear notations** rely on **repeated occurrences of variables** or names to show connections. While individual rules in linear logic can be readable, the relationships between rules may not be as immediately clear as direct connections in a graphic representation.

Semantic networks, in their various forms, remain a powerful and versatile tool for knowledge representation and reasoning in AI, with ongoing developments and broad applications in areas like the Semantic Web.