Chapter Four

Network representation

This type of knowledge representation captures knowledge as a graph in which the nodes represent objects or concepts in the problem domain and the arcs represent relations or associations between them.

By : Dr Muhanad Tahrir Younis.

4-1 Semantic Networks

In semantic networks we represent knowledge as a graph, with the nodes corresponding to concepts/facts in the problem domain, and the arcs to relations or associations between concepts.

Let us look at how the following English sentences can be represented using a semantic network.

"Frosty is a snowman. A snowman is made of snow. Snow is frozen water. It is slippery and soft. Snow is cold. Ice is also cold. It is also frozen water, but unlike snow, which is soft, ice is hard. Ice is clear in color."

Figure (4-1) shows the information above represented as a semantic network. Notice that edges represent relationships between concepts, and concepts are represented as nodes.

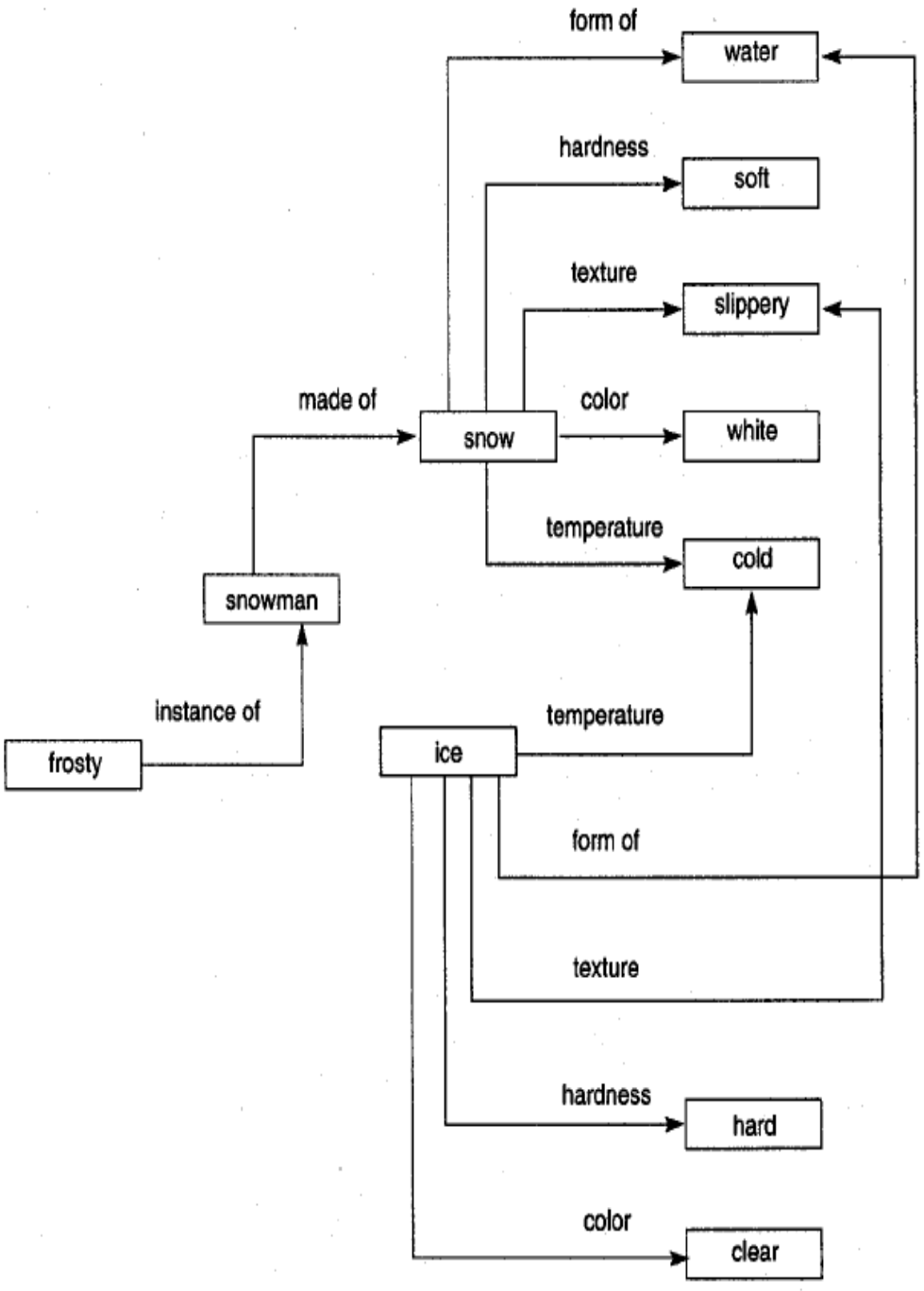

Figure (4-1): Semantic network for Frosty, snowman, ice, and snow.

Another semantic network is shown in Figure (4-2). In this semantic network we see that a canary's feature that it can sing is directly attached to it, while the canary's capability to fly is inherited from a canary's parent concept, bird. In this way it will take longer to extract the information than when the feature is attached to the concept explicitly.

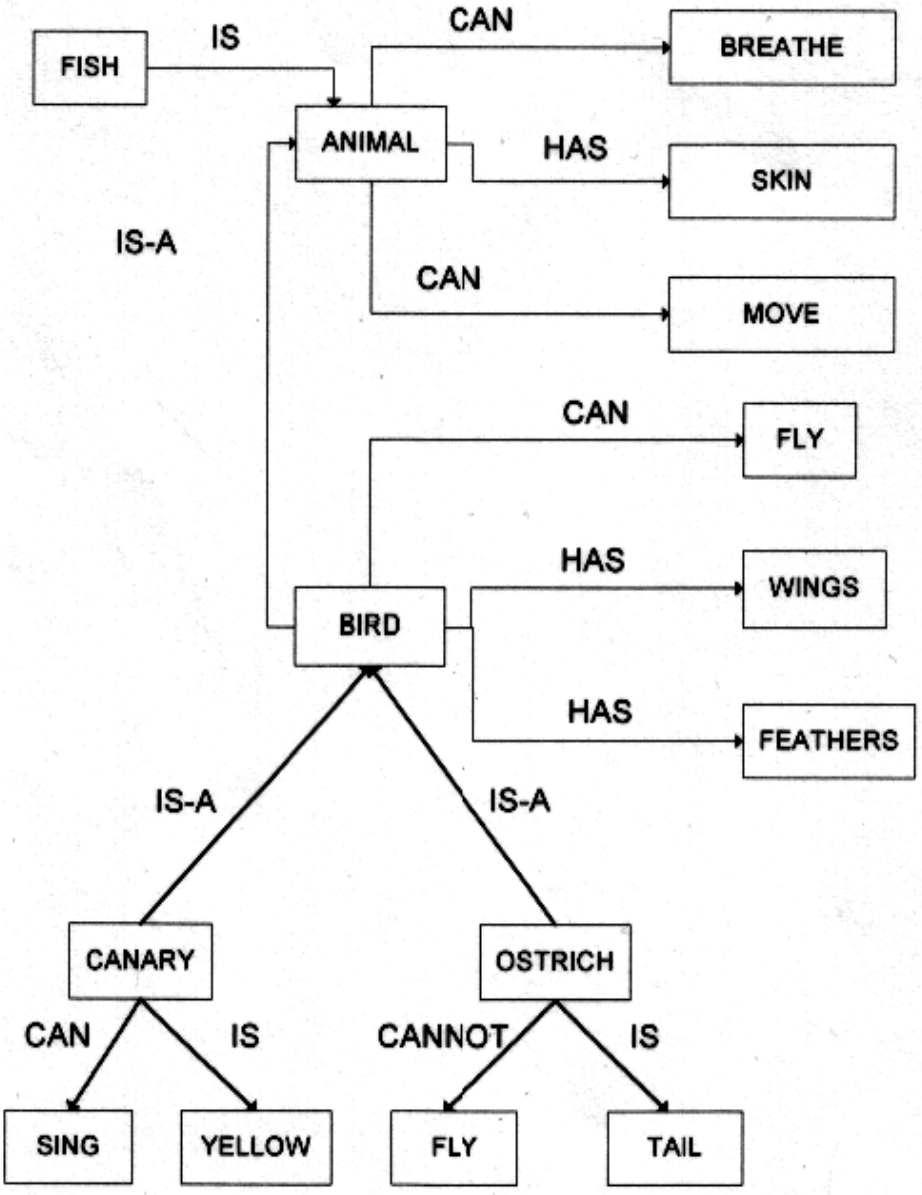

Figure (4-2): Semantic network for Animal, Bird, and Canary.

Another approach, which uses standard relationships, focuses on the *case structure* of English verbs. In this verb-oriented approach, links define the roles played by nouns and noun phrases in the action of the sentence. Case relationships include *agent*, *object*, *instrument*, *location*, and *time*. A sentence is represented as a verb node, with various case links to nodes representing other participants in the action. This structure is called a *case frame*. In parsing a sentence, the program finds the verb and retrieves the case frame for that verb from its knowledge base. It then binds the values of the agent, object, etc., to the appropriate nodes in the case frame. Using this approach, the sentence "Sarah fixed the chair with glue" might be represented by the network in Figure (4-3).

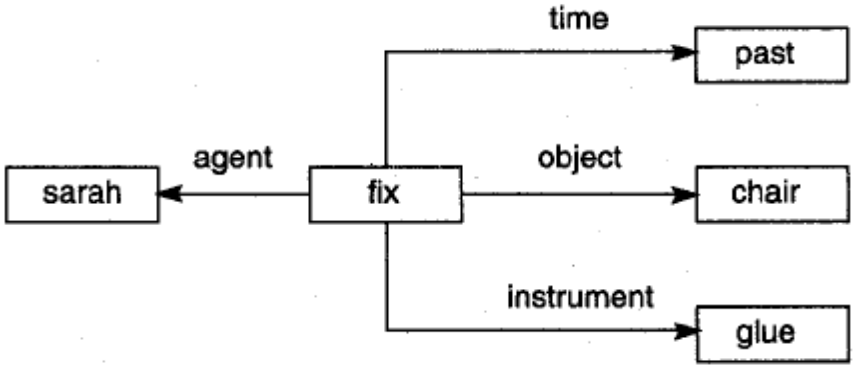

Figure (4-3): Case frame representation of the sentence "***Sarah fixed the chair with glue***".

4-2 Conceptual Graphs

A conceptual graph is a finite, connected, bipartite graph. The nodes of the graph are either concepts or conceptual relations. Concepts are represented as boxes and relations as ellipses.

In the first conceptual graph depicted in figure (4-4), we represent the concept that a bird flies. In the same figure, we see that a dog is colored

brown. Notice that the relation flies and color are stored in an ellipse node and the concepts bird, dog and brown are stored in boxes.

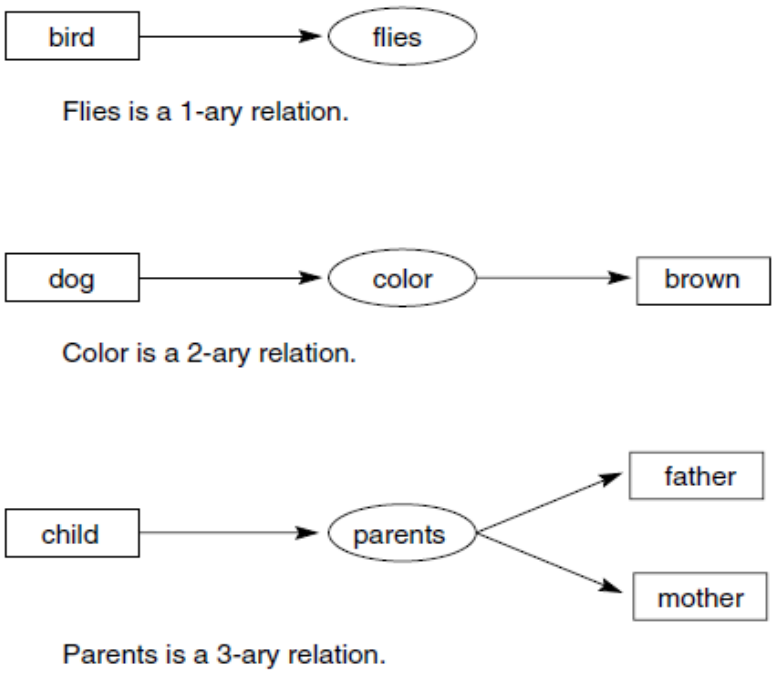

Figure (4-4): Conceptual relations of different arities.

Each conceptual graph represents a single proposition. A typical knowledge base will contain a number of such graphs. Graphs may be arbitrarily complex but must be finite. For example, one graph in Figure (4-5) represents the proposition "**A dog has a color of brown**". Figure (4-6) is a graph of somewhat greater complexity that represents the sentence "**Mary gave John the book**". This graph uses conceptual relations to represent the cases of the verb "**to give**" and indicates the way in which conceptual graphs are used to model the semantics of natural language.

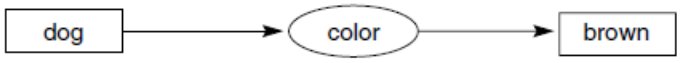

Figure (4-5): Conceptual graph of the sentence "**A dog has a color of brown** ".

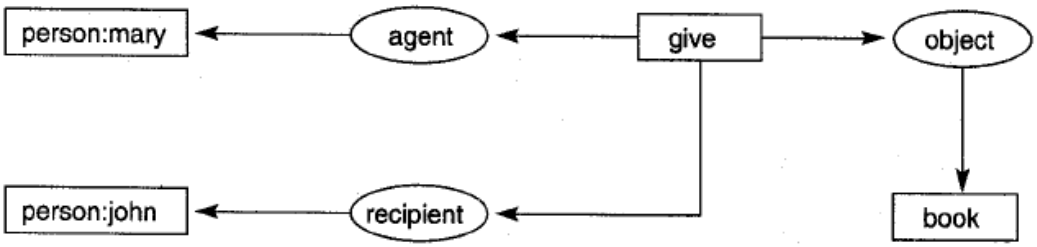

Figure (4-6): Conceptual graph of the sentence "**Mary gave John the book**".

4-3 Types, Individuals, and Names in Conceptual Graphs

Many early designers of semantic networks were careless in defining class/member and class/subclass relationships, with resulting semantic confusion. For example, the relation between an individual and its class is different from the relation between a class (such as dog) and its superclass (carnivore). Similarly, certain properties belong to individuals, and others belong to the class itself; the representation should provide a vehicle for making this distinction. The properties of having fur and liking bones belong to individual dogs; the class "dog" does not have fur or eat anything. Properties that are appropriate to the class include its name and membership in a zoological taxonomy.

In conceptual graphs, every concept is a unique individual of a particular type. Each concept box is labeled with a ***type*** label, which indicates the class or type of individual represented by that node. Thus, a node labeled dog represents some individual of that type. Types are organized into a hierarchy. The type dog is a subtype of carnivore, which is a subtype of mammal, etc. Boxes with the same type label represent concepts of the same type; however, these boxes may or may not represent the same individual concept.

Each concept box is labeled with the names of the type and the individual. The type and individual labels are separated by a colon ":". The graph of Figure (4-7) indicates that ***the dog "Emma" is brown***. The graph of Figure (4-8) asserts that ***some unspecified entity of type dog has a color of brown***. If the individual ***is not indicated***, the concept represents an ***unspecified individual of that type***.

Conceptual graphs also let us indicate specific but unnamed individuals. A unique token called a ***marker*** indicates each individual in the world of discourse. This marker is written as a **number preceded by a #**. Markers are different from names in that they are unique: individuals may have one name, many names, or no name at all, but they have exactly one marker. Similarly, different individuals may have the same name but may not have the same marker. This distinction gives us a basis for dealing with the semantic ambiguities that arise when we give objects names. The graph of Figure (4-8) asserts that **a particular dog, #1352, is brown**.

Markers allow us to separate an individual from its name. If dog #1352 is named "Emma," we can use a conceptual relation called name to add this to the graph. The result is the graph of Figure (4-9). The name is enclosed in double quotes to indicate that it is a string. Where there is no danger of ambiguity, we may simplify the graph and refer to the individual directly by name. Under this convention, the graph of Figure (4-9) is equivalent to the graph of Figure (4-7).

Figure (4-7): Conceptual graph indicating that the dog named emma is brown.

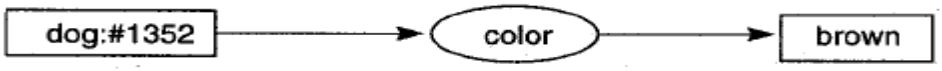

Figure (4-8): Conceptual graph indicating that a particular (but unnamed) dog is brown.

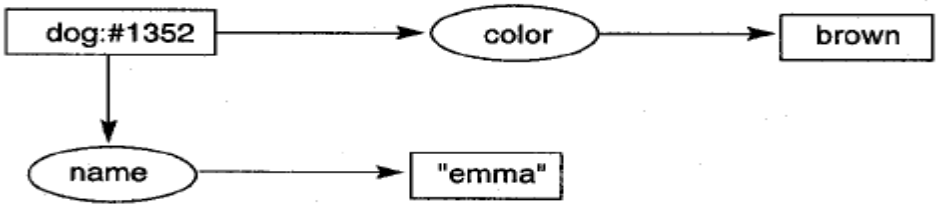

Figure (4-9): Conceptual graph indicating that a dog named emma is brown.

Although we frequently ignore it both in casual conversation and in formal representations, this distinction between an individual and its name is an important one that should be supported by a representation language. For example, if we say that "John" is a common name among males, we are asserting a property of the name itself rather than of any individual named "John". This allows us to represent such English sentences as "'Chimpanzee' is the name of a species of primates". Similarly, we may want to represent the fact that an individual has several different names. The graph of Figure (4-10) represents the situation described in the song lyric: "**Her name was McGill, and she called herself Lil, but everyone knew her as Nancy**".

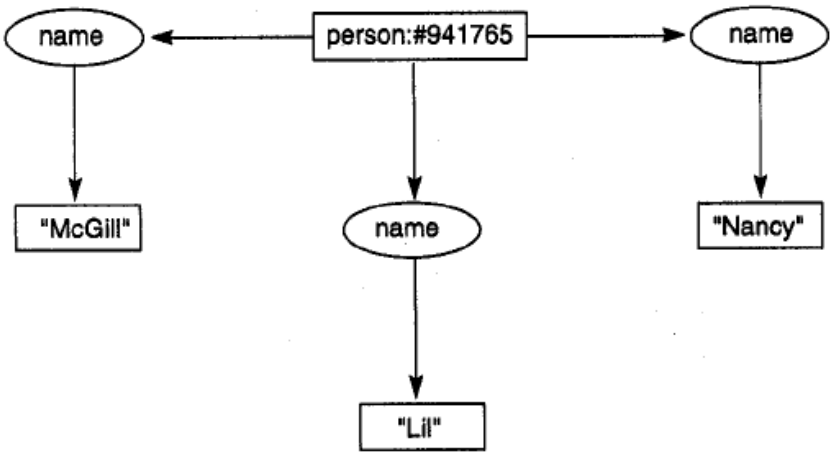

Figure (4-10): Conceptual graph of a person with three names.

As an alternative to indicating an individual by its marker or name, we can also use the **generic marker *** to indicate an unspecified individual. By convention, this is often omitted from concept labels; a node given just a type label, dog, is equivalent to a node labeled dog:*. In addition to the generic marker, conceptual graphs allow the use of named variables. These are represented by an asterisk followed by the variable name (e.g., *X or *foo). This is useful if two separate nodes are to indicate the same, but unspecified, individual. The graph of Figure (4-11) represents the assertion "**The dog scratches its ear with its paw**". Although we do not know which dog is scratching its ear, the variable *X indicates that the paw and the ear belong to the same dog that is doing the scratching.

To summarize, each concept node can indicate an individual of a specified type. This individual is the *referent* of the concept. This reference is indicated either individually or generically. If the referent uses an individual marker, the concept is an *individual* concept; if the referent uses the generic marker, then the concept is *generic*.

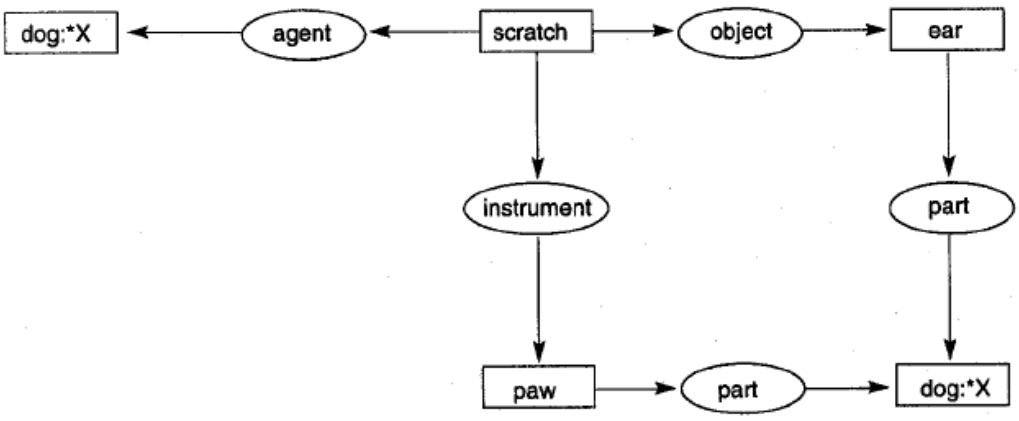

Figure (4-11): Conceptual graph of the sentence "**The dog scratches its ear with its paw**".

4-4 Propositional Nodes in Conceptual Graphs

In addition to using graphs to define relations between objects in the world, we may also want to define relations between propositions. Consider, for example, the statement "Tom believes that Jane likes pizza". "Believes" is a relation that takes a proposition as its argument.

Conceptual graphs include a concept type, *proposition*, that takes a set of conceptual graphs as its referent and allows us to define relations involving propositions. Propositional concepts are indicated as a box that contains another conceptual graph. These proposition concepts may be used with appropriate relations to represent knowledge about propositions. Figure (4-12) shows the conceptual graph for the above assertion about Jane, Tom, and pizza. The experiencer relation is loosely analogous to the agent relation in that it links a subject and a verb. The experiencer link is used with belief states based on the notion that they are something one experiences rather than does.

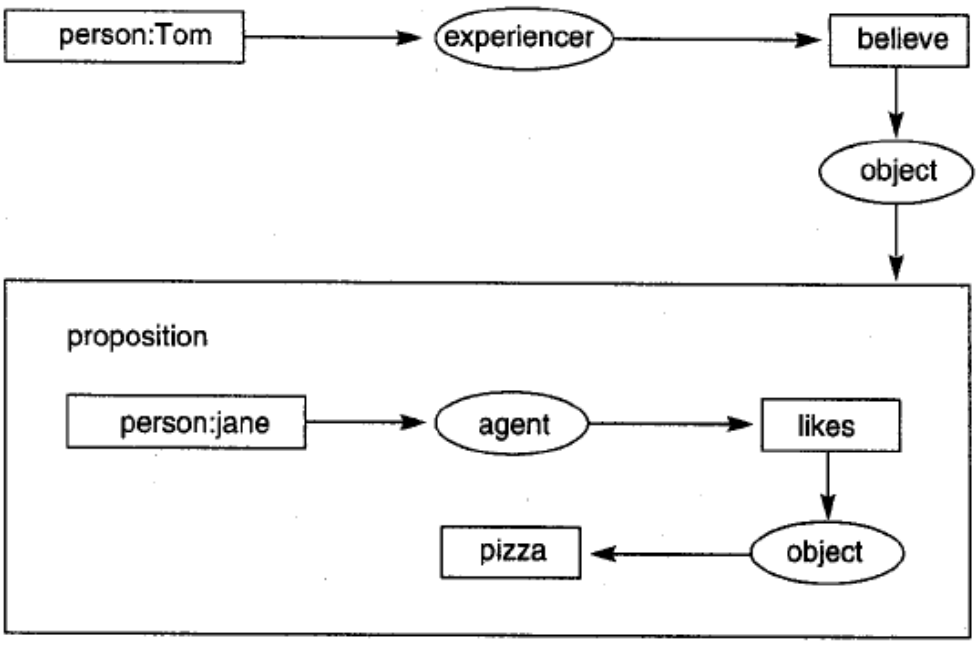

Figure (4-12): Conceptual graph of the statement "**Tom believes that Jane likes pizza**", showing the use of a propositional concept.

4-5 Conceptual Graphs and Logic

We may implement negation using propositional concepts and a unary operation called **neg**. neg takes as argument a proposition concept and asserts that concept as false. The conceptual graph of Figure (4-13) uses neg to represent the statement "**There are no pink dogs**".

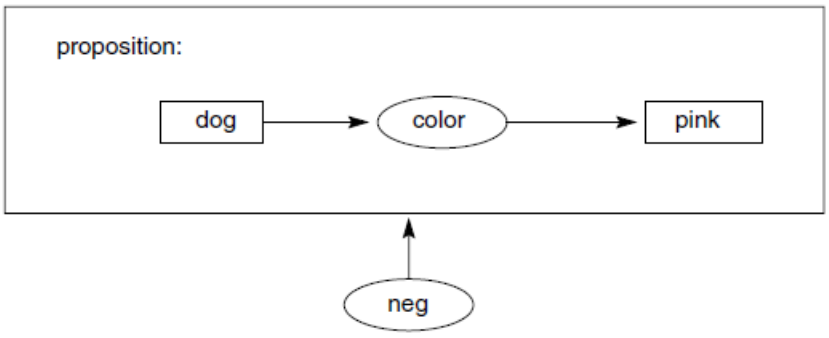

Figure (4-13): Conceptual graph of the proposition "**There are no pink dogs**".