Logic

Stephen G. Ware

CSCI 4525 / 5525

Narrative Intelligence LAB

THE UNIVERSITY of NEW ORLEANS

Logic

- How can we represent knowledge about the world in a general, reusable way?
- How can we use existing knowledge to gain new knowledge?

Problem Solving Approaches

Procedural: For each new problem, write a domain-specific algorithm to solve it.

- Fast
- Not reusable
- Requires domain expertise

Declarative: Describe the problem in a common syntax and solve it using general techniques.

- Slow
- Reusable
- Less expertise required

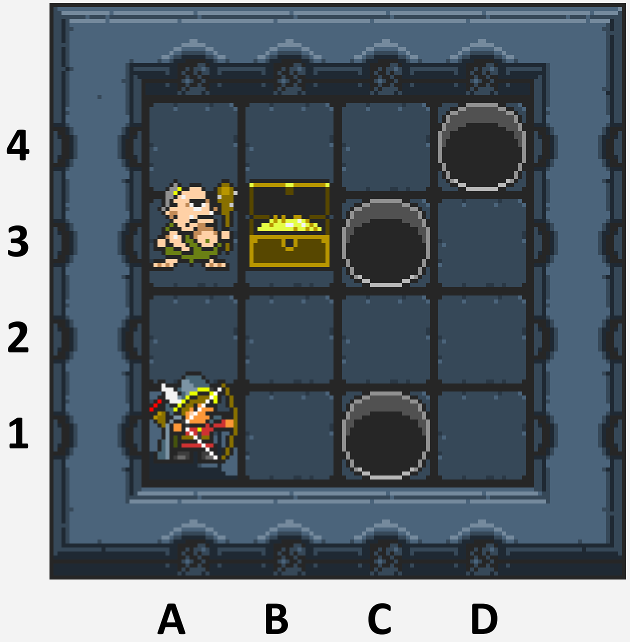
Syntax & Semantics

The rules of **syntax** for a logic define which expressions are legal to write.

The **semantics** of a logic define what an expression means and when it is true or false.

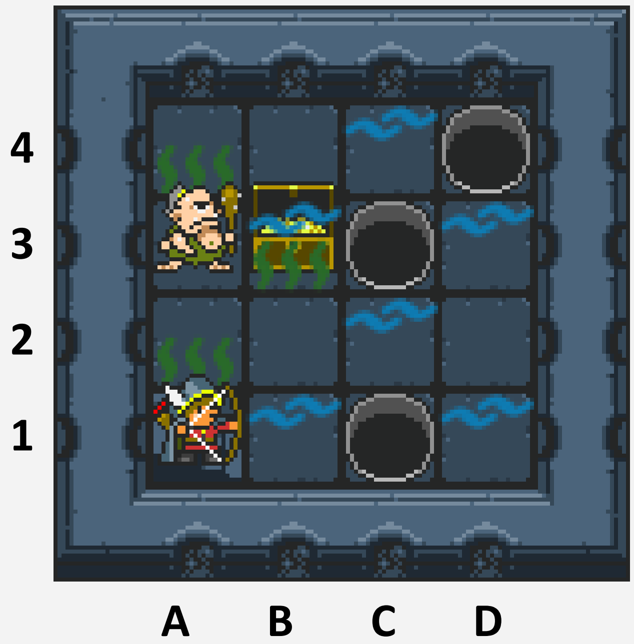
Logical algorithms only manipulate syntax, but it is up to the user or the agent to interpret meaning.

Wumpus World



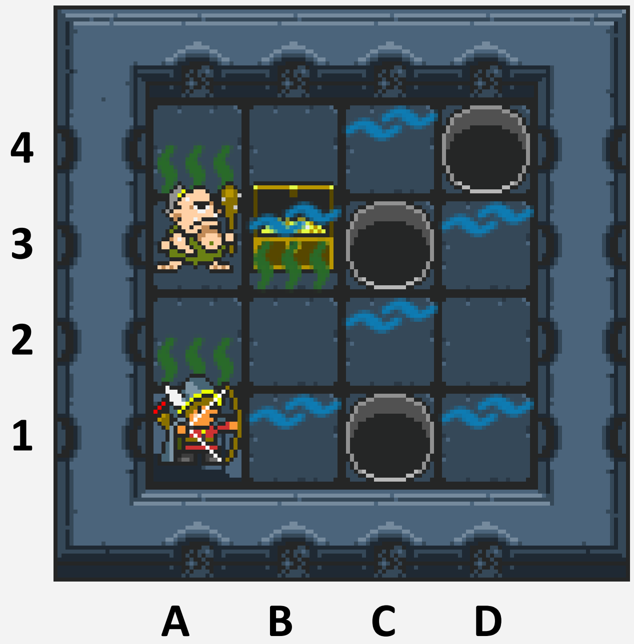
- The player, who can navigate the grid world
- The wumpus, who will eat the player if the player blunders into its square
- Bottomless pits
- A chest of gold

Wumpus World



- When adjacent to the wumpus, the player detects a stench.
- When adjacent to a pit, the player detects a breeze.
- When in the square with the gold, the player detects a glimmer.

Wumpus World



Actions:

- Move forward
- Turn left
- Turn right
- Grab gold
- Fire arrow (kills the wumpus if it lies along the trajectory)

Logics

How statements are made:

- Propositional Logic
- Predicate Logic
- First Order Logic

How statements are combined:

- Boolean Logic

Propositional Logic

Every statement about the world which can be true or false is represented as a unique symbol.

Syntax: R

Semantics: "It is raining outside."

Syntax: PA1

Semantics: "There is a pit at square A1."

Boolean Logic

Boolean operators provide a way to combine individual statements (which can each be true or false) into larger expressions which, as a whole, are true or false.

Negation

A **NOT** expression (or **negation**) has the opposite truth value as its term.

x	$\neg x$
T	F
F	T

Conjunction

An **AND** expression (or **conjunction**) is true just when all of its individual conjuncts are true.

x	y	$x \wedge y$
T	T	T
T	F	F
F	T	F
F	F	F

Disjunction

A **OR** expression (or **disjunction**) is true when at least one of its individual disjuncts is true.

x	y	$x \vee y$
T	T	T
T	F	T
F	T	T
F	F	F

Implication

A **IF** expression (or **implication**) has an **antecedent** followed by a **consequent**. It is true when its consequent is true or when both the antecedent and consequent are false.

x	y	$x \to y$
T	T	T
T	F	F
F	T	T
F	F	T

Biconditional

An **IFF** (if an only if) expression (or **biconditional**) is true just when both sides have the same truth value.

x	y	$x \leftrightarrow y$
T	T	T
T	F	F
F	T	F
F	F	T

The two sides are said to be **logically equivalent**.

Statements vs. Things

The problem with making every statement its own symbol is that we can't say anything about its parts.

We want to distinguish between statements (which have a truth value) and the things those statements are about (which do not have a truth value).

Statements vs. Things

"It is raining outside." is a statement.

It is true or false.

"rain" and "outside" are things.

Statements are made about things.

They are not true or false.

Predicate Logic

The official word for a "thing" is a **term**. There are three types of terms:

1. A **constant** represents a specific thing.
2. A **function** is a parameterized thing.
3. A **variable** stands for any term.

Statements are made by applying a **predicate** (relationship) to 0 or more terms.

Predicate Logic Example

Predicates:

- at = "a thing is at a location"
- breeze = "breeze at"
- stench = "stench at"

Things:

- P = "player"
- W = "wumpus"
- P1 = "pit 1"
- A1 = "square A1"...

Examples:

- $at(P1, B3)$ = "Pit 1 is at square B3."
- $at(P, B2)$ = "The player is at square B2."
- $breeze(B2)$ = "The player detects a breeze at B2."

Predicate Logic Example (alt)

Predicates:

- breeze = “breeze at”
- pit = “pit at”
- stench = “stench at”
- wumpus = “wumpus at”

Things:

- A1 = “square A1”
- A2 = “square A2”…

Examples:

- $breeze(B2)$ = “The player detects a breeze at B2”
- $pit(B3)$ = “There is a pit at B3.”

Functions

A function is a term which can be treated like a constant, but which is parameterized with other terms.

e.g. $above(x)$ means "the square above square x."

e.g. $above(B1)$ means "the square above B1."

Functions

A function is a term which can be treated like a constant, but which is parameterized with other terms.

Do not confuse functions with predicates!

- They look the same, but…
- Functions are things. They have no truth value.
- Predicates are statements about relationships. They do have truth value.

Ground Expressions

We say that a predicate logic expression is ground when it contains no variables.

e.g. $pit(x)$ is not ground

e.g. $pit(C1)$ is ground

First Order Logic

Specifies not only relationships between objects but also over quantities of objects.

The **universal quantifier** $\forall$ means "for all."

e.g. $\forall x\ animal(x) \rightarrow loves(John, x)$ =

"John loves all animals."

The **existential quantifier** $\exists$ means "there exists."

e.g. $\exists x\ loves(x, John)$ = "Someone loves John."

Scope of Quantified Variables

Consider two facts about the Wumpus world:

$$\forall x(breeze(x) \rightarrow \exists y\ (adjacent(x, y) \land pit(y)))$$

$$\forall x\underbrace{(stench(x) \rightarrow \exists y\ (adjacent(x, y) \land wumpus(y)))}_{x \text{ only exists here}}$$

Scope of Quantified Variables

Consider two facts about the Wumpus world:

$$\forall x(breeze(x) \rightarrow \exists y\ (adjacent(x, y) \wedge pit(y)))$$

$$\forall x(stench(x) \rightarrow \exists y\ \underbrace{(adjacent(x, y) \wedge wumpus(y))}_{y \text{ only exists here}})$$

Standardizing Apart

Consider two facts about the Wumpus world:

$\forall x(breeze(x) \rightarrow \exists y\ (adjacent(x, y) \land pit(y)))$

$\forall x(stench(x) \rightarrow \exists y\ (adjacent(x, y) \land wumpus(y)))$

The reuse of x and y in these two facts can get confusing, so we **standardize apart** the variable names to avoid any confusion.

Standardizing Apart

Consider two facts about the Wumpus world:

$\forall w(breeze(w) \rightarrow \exists x\ (adjacent(w, x) \wedge pit(x)))$

$\forall y(stench(y) \rightarrow \exists z\ (adjacent(y, z) \wedge wumpus(z)))$

The reuse of x and y in these two facts can get confusing, so we **standardize apart** the variable names to avoid any confusion.

Conventions for Quantifiers

When using first order logic, we assume that all variables are universally quantified. This means we can leave off universal quantifiers.

$\forall x \; loves(John, x)$

can simply be written as:
$loves(John, x)$

Skolemization

We can replace existentially quantified variables with a single **Skolem constant** representing the object in question.

$\exists x\ loves(x, Mosquitoes)$

can simply be written as:
$loves(P, Mosquitoes)$

P is a person who loves mosquitoes.

Logics (Review)

- **Propositional logic** is simple, so we will often use it in examples, but it is not very expressive.
- **Predicate logic** is more expressive, especially when combined with…
- **First order logic** adds universal and existential quantifiers.
- **Boolean logic** can be used with any logic above.

Important Equivalencies

- Negated Quantifiers
- Implications and Disjunctions
- De Morgan's Laws
- Distribution of OR over AND

Negated Quantifiers

The universal and existential quantifiers are related; each is the negation of the other.

- $\neg\forall x\ loves(John, x)$ means "It is not true that John loves everything."

Its negation would be:

- $\exists x\ \neg loves(John, x)$, which means "There exists something that John does not love."

Negated Quantifiers

The universal and existential quantifiers are related; each is the negation of the other.

- $\neg\exists x\ loves(x, Mosquitoes)$ means "There does not exist anyone who loves mosquitoes."

Its negation would be:

- $\forall x\ \neg loves(x, Mosquitoes)$, which means "Everyone does not loves mosquitoes."

Implications and Disjunction

Notice the similarity:

x	y	$x \to y$
T	T	T
T	F	F
F	T	T
F	F	T

x	y	$\neg x \vee y$
T	T	
T	F	
F	T	
F	F	

Implications and Disjunction

Notice the similarity:

x	y	$x \to y$
T	T	T
T	F	F
F	T	T
F	F	T

x	y	$\neg x \vee y$
T	T	T
T	F	F
F	T	T
F	F	T

The expressions $x \to y$ and $\neg x \vee y$ are equivalent.

De Morgan's Laws

$$\neg(x \land y) \leftrightarrow \neg x \lor \neg y$$

$$\neg(x \lor y) \leftrightarrow \neg x \land \neg y$$

Exercise: Write the truth tables to prove these laws.

Distribution of OR over AND

$$x \lor (y \land z) \leftrightarrow (x \lor y) \land (x \lor z)$$

Exercise: Write truth tables to prove this.

Distribution of OR over AND

This can be generalized to situations like this:

$$(w \land x) \lor (y \land z) \leftrightarrow$$

$$\big((w \land x) \lor y\big) \land \big((w \land x) \lor z\big)$$

$$\big(y \lor (w \land x)\big) \land \big(z \lor (w \land x)\big)$$

$$\big((y \lor w) \land (y \lor x)\big) \land \big((z \lor w) \land (z \lor x)\big)$$

$$(y \lor w) \land (y \lor x) \land (z \lor w) \land (z \lor x)$$

$$(w \lor y) \land (w \lor z) \land (x \lor y) \land (x \lor z)$$

Database Semantics

- **Unique Names Assumption**

  Each constant refers to a different object.

  (Note: This means two constants cannot be equal.)
- **Domain Closure Assumption**

  Only those objects we name exist.
- **Closed World Assumption**

  Facts not explicitly stated to be true are false.

Skolemization (cont.)

Consider this statement:

$$\forall x\ \exists y\ breeze(x) \leftrightarrow adjacent(x, y) \land pit(y)$$

By convention, we drop the universal variable:

$$\exists y\ breeze(x) \leftrightarrow adjacent(x, y) \land pit(y)$$

Can we replace y with a Skolem constant?

Skolemization (cont.)

Original statement:

$$\forall x\ \exists y\ breeze(x) \leftrightarrow adjacent(x, y) \wedge pit(y)$$

Replaced with Skolem constant:

$$breeze(x) \leftrightarrow adjacent(x, S) \wedge pit(S)$$

This does not have the same meaning! It says that all breezy squares are adjacent to *the same* pit square.

Skolemization (cont.)

Original statement:

$$\forall x\ \exists y\ breeze(x) \leftrightarrow adjacent(x, y) \wedge pit(y)$$

Replaced with Skolem function:

$$breeze(x) \leftrightarrow adjacent(x, S(x)) \wedge pit(S(x))$$

This has the same meaning, because the function can stand for more than just one square.

Skolemization (cont.)

When an existential quantifier appears within the scope of one or more universal quantifiers, we replace the existentially quantified variable with a **Skolem function** whose parameters are all the universally quantified variables.

Skolem functions can be seen as a generalization of Skolem constants, because a constant can be thought of as a function with 0 parameters.

Conjunctive Normal Form

- Literals
- Disjunctive Clauses
- Conjunctive Normal Form (CNF)
- Converting to CNF

Literals

A **literal** is an individual statement or its negation.

Propositional logic example:

- x is a literal
- $\neg x$ is a literal

Predicate logic example:

- $pit(A1)$ is a literal
- $\neg pit(A1)$ is a literal

Clauses

A **disjunctive clause** is a disjunction whose disjuncts are all literals. (This includes a disjunction of 1 disjunct, i.e. a literal by itself.)

- $animal(x)$ is a disjunctive clause
- $\neg animal(x) \vee loves(John, x)$ is a disjunctive clause

Conjunctive Normal Form

An expression is in **conjunctive normal form** (CNF) if it is a conjunction of disjunctive clauses. (This includes a conjunction of 1 literal, i.e. a literal by itself.)

- $\underbrace{\big(animal(x)\big)}_{\text{disjunctive clause}} \land \underbrace{\big(\neg animal(x) \lor loves(John, x)\big)}_{\text{disjunctive clause}}$
  is in conjunctive normal form

Disjunctive Normal Form

An expression is in **conjunctive normal form** (CNF) if it is a conjunction of disjunctive clauses. (This includes a conjunction of 1 literal, i.e. a literal by itself.)

- $\underbrace{(animal(x)) \land (\neg animal(x) \lor loves(John, x))}_{\text{CNF expression}}$ is in conjunctive normal form

Conjunctive Normal Form

All logical expressions can be converted to conjunctive normal form.

This allows us to standardize the input to logical algorithms and simplify the code.

Converting to CNF

1. Eliminate implications.
2. Move $\neg$ inwards.
3. Standardize variable names apart.
4. Skolemize.
5. Drop universal quantifiers.
6. Apply De Morgan's laws.

Converting to CNF

Convert the following to CNF:

$$\forall x\ \left(\forall y\ animal(y) \rightarrow loves(x,y)\right) \rightarrow \exists y\ loves(y,x)$$

"Everyone who loves all animals is loved by someone."

Converting to CNF

1. Eliminate implications.

Remember: $(x \rightarrow y) \leftrightarrow (\neg x \vee y)$

$$\forall x \left(\forall y\ animal(y) \rightarrow loves(x, y)\right) \rightarrow \left(\exists y\ loves(y, x)\right)$$

becomes:

Converting to CNF

1. Eliminate implications.

Remember: $(x \rightarrow y) \leftrightarrow (\neg x \vee y)$

$$\forall x \, \big(\forall y \; animal(y) \rightarrow loves(x, y)\big) \rightarrow \big(\exists y \; loves(y, x)\big)$$

becomes:

Converting to CNF

1. Eliminate implications.

Remember: $(x \rightarrow y) \leftrightarrow (\neg x \vee y)$

$$\forall x \left(\forall y\, animal(y) \rightarrow loves(x,y)\right) \rightarrow \left(\exists y\, loves(y,x)\right)$$

becomes:

$$\forall x \left(\neg\left(\forall y\, animal(y) \rightarrow loves(x,y)\right)\right) \vee \left(\exists y\, loves(y,x)\right)$$

Converting to CNF

1. Eliminate implications.

Remember: $(x \rightarrow y) \leftrightarrow (\neg x \vee y)$

$$\forall x \left(\forall y \; animal(y) \rightarrow loves(x, y)\right) \rightarrow \left(\exists y \; loves(y, x)\right)$$

becomes:

$$\forall x \; \left(\neg(\forall y \; animal(y) \rightarrow loves(x, y))\right) \vee \left(\exists y \; loves(y, x)\right)$$

Converting to CNF

1. Eliminate implications.

Remember: $(x \rightarrow y) \leftrightarrow (\neg x \vee y)$

$$\forall x\ \big(\forall y\ animal(y) \rightarrow loves(x,y)\big) \rightarrow \big(\exists y\ loves(y,x)\big)$$

becomes:

$$\forall x\ \Big(\neg\big(\forall y\ \neg animal(y) \vee loves(x,y)\big)\Big) \vee \big(\exists y\ loves(y,x)\big)$$

Converting to CNF

1. Eliminate implications.

Remember: $(x \rightarrow y) \leftrightarrow (\neg x \vee y)$

$$\forall x \left(\forall y\ animal(y) \rightarrow loves(x, y)\right) \rightarrow \left(\exists y\ loves(y, x)\right)$$

becomes:

$$\forall x\ \left(\neg(\forall y\ \neg animal(y) \vee loves(x, y))\right) \vee \left(\exists y\ loves(y, x)\right)$$

Converting to CNF

2. Move $\neg$ inwards.

Remember: Negated quantifiers & De Morgan's laws.

$$\forall x\ (\neg(\forall y\ \neg animal(y) \vee loves(x,y))) \vee (\exists y\ loves(y,x))$$

becomes:

$$\forall x\ \big(\exists y\ animal(y) \wedge \neg loves(x,y)\big) \vee (\exists y\ loves(y,x))$$

"Either there is some animal that *x* does not love, or someone loves *x*."

Converting to CNF

3. Standardize variable names apart.

$$\forall x \left(\exists y\, animal(y) \land \neg loves(x,y)\right) \lor (\exists y\, loves(y,x))$$

becomes:

$$\forall x \left(\exists y\, animal(y) \land \neg loves(x,y)\right) \lor (\exists z\, loves(z,x))$$

Converting to CNF

4. Skolemize.

$$\forall x\ \big(\exists y\ animal(y) \wedge \neg loves(x,y)\big) \vee (\exists z\ loves(z,x))$$

becomes:

$$\forall x\ \big(animal(F(x)) \wedge \neg loves(x, F(x))\big) \vee loves(G(x), x)$$

Converting to CNF

5. Drop universal quantifiers.

$$\forall x \left(animal(F(x)) \land \neg loves(x, F(x))\right) \lor loves(G(x), x)$$

becomes:

$$\left(animal(F(x)) \land \neg loves(x, F(x))\right) \lor loves(G(x), x)$$

Converting to CNF

6. Distribute AND over OR.

$$\big(animal(F(x)) \land \neg loves(x, F(x))\big) \lor loves(G(x), x)$$

becomes:

$$(animal(F(x)) \lor loves(G(x), x))$$
$$\land\ (\neg loves(x, F(x)) \lor loves(G(x), x))$$