10 Knowledge Representation

Let us again consider the MopBot example. Remember, our MopBot has: Our mopbot has:

- Sensors, which enables it to detect whether there is dirt in the room.
- Actuators, which enables it to clean the room or move to another room.

Suppose, we want to tell that MopBot should clean the room B and come back to room A. It might develop the following simple plan p_1 to achieve the goal:

1. Move to B.
2. Clean B.
3. Leave B when no dirt is detected.

Alternatively, the MopBot may also adopt a more complex plan p_2 to achieve the goal:

1. Move to B.
2. If dirt is detected, clean B. Otherwise, there is no need to clean.
3. Leave B when no dirt is detected.

Here, MopBot had known two things:

- No dirt detected when room is clean.
- Leave room B when No dirt detected.

From above two statements, we want MopBot to have knowledge of leave room B when room is clean, in other-words, we want the MopBot to somehow store this knowledge.

10.1 Propositional logic

Knowledge is a concept that is very hard to define, but one definition of knowledge that has been generally agreed upon is: An agent A knows that statement S is true if and only if:

1. S is true.
2. A believes S is true.
3. A is justified in believing that S is true.

We will now introduce the idea of *propositions*, which are variables that take on values *True* or *False*. We can decide whether or not we can make such deductions by replacing these statements with propositions. For instance, assume

- g represent Greeks,
- h represent Humans,
- m represent Mortals.

All greeks are human.	$g \to h$
All humans are mortal.	$h \to m$
All greeks are mortal.	$g \to m$

The problem is in general we can not differentiate between whether statement S is true or Agent A believes that S is true. With the statements "all Greeks are human" and "all humans are mortal", we can deduce that "all Greeks are mortal". However, with "not all Greeks are human" and "all humans are mortal", we are not able to deduce that "not all Greeks are mortal".

Not all greeks are human.	
All humans are mortal.	
Not all greeks are mortal.	can't deduce

After introducing propositions, we need a way of combining them to construct meaningful sentences. This led to the introduction of operators, including:

- unary operators: $\neg$ (Negation)
- binary operations: $\wedge$ and $\vee$, $a \to b \equiv \neg a \vee b$

Finally, we would want to be able to express our knowledge base(KB). The language of Knowledge base:

- PROP: the set of all propositions, and
- FORM: the set of all formulas/sentences that can be expressed by propositional logic.

Using these expressions, we will be able to classify which statements are allowed in propositional logic, and which are not allowed. For instance, $(p \vee q)$ is allowed under PROP and FORM, whereas $()p \wedge q$ is not allowed. FORM can be recursively defined as such:

$$\begin{aligned}\text{FORM}_0 &= \text{PROP}\\ \text{FORM}_{i+1} &= \text{FORM}_i \cup \{(\alpha \circ \beta) | \alpha, \beta \in \text{FORM}_i\} \cup \{(\neg\alpha) | \alpha \in \text{FORM}_i\}\\ \text{FORM} &= \bigcup_{i=0}^{\infty} \text{FORM}_i\end{aligned}$$

where $\alpha \circ \beta \equiv \{(\alpha \vee \beta)\} \vee \{(\alpha \wedge \beta)\}$. Every $\alpha \in KB$ is also in FORM.

Every formula φ may be true or false depending on the situation. For example: $p \vee q$ is true when $p = 0$ and $q = 1$, but false when $p = 0, q = 0$. Here, we define the notion of truth assignment τ.

$$\tau : PROP \to \{0, 1\}$$

If a formula φ is true in τ, we say $\tau \models \varphi$. To check whether $\tau \models \varphi$ or not, we can substituted every propositional variables with its truth assignment, and if φ evaluates to true then $\tau \models \varphi$ else not.

$$\tau \models \varphi \text{ iff } \varphi(\tau) = 1$$

where $\varphi(\tau)$ is φ after substituting propositions of φ with their values in τ.

Let us consider an example to understand the aforementioned statement.

$$\begin{array}{lll} \varphi = ((P \lor q) \land (\neg r)) & & \\ \tau_1 = \{p \to 1, q \to 1, r \to 1\} & \varphi(\tau_1) = 0 & \tau_1 \not\models \varphi \\ \tau_2 = \{p \to 1, q \to 0, r \to 0\} & \varphi(\tau_2) = 1 & \tau_2 \models \varphi \end{array}$$

There are $2^{|PROP|}$ truth assignment possible.

- φ is SAT(Satisfiable): $\exists \tau$ such that $\tau \models \varphi$.
- φ is Valid: $\forall \tau$, $\tau \models \varphi$.
- φ is UNSAT(Unsatisfiable): $\forall \tau$,$\tau \not\models \varphi$. or there does not exists τ such that $\tau \models \varphi$.
- One important observation: **φ is VALID iff $\neg\varphi$ is UNSAT**

A few examples to understand the above definitions.

- $(p \lor \neg p)$ is VALID, SAT.
- $(p \lor q)$ is SAT, but not VALID.
- $(p \lor q) \land (\neg p \lor q) \land (p \lor \neg q) \land (\neg p \lor \neg q)$ is UNSAT.

This allows us to find meanings in the sentences; for example $(p \lor q)$ and $(q \lor p)$ are different strings but seem to have meanings. In general two formulas φ and Φ are called semantically equivalent $(\varphi \leftrightarrow \Phi)$ if following holds:

$$\varphi \leftrightarrow \psi \text{ iff } \forall \tau, \varphi(\tau) = \psi(\tau)$$

Some definitions:

- $\psi \models \varphi$ iff $\psi \to \varphi$ is VALID.
- $\psi \models \models\!\!| \varphi$ iff $\psi \leftrightarrow \varphi$ is VALID.
- If $\psi \models \models\!\!| \varphi$, then φ and ψ are called semantically equivalent.

For example: let $\varphi = (p \lor (q \land r))$ and $\Phi = ((p \lor q) \land r)$. $\varphi \not\leftrightarrow \Phi$ as for $\tau = \{p \to 1, q \to 1, r \to 0\}, \varphi(\tau) \neq \psi(\tau)$. Similarly:

$$(C_1 \land (C_2 \land (C_3 \land C_4))) \leftrightarrow ((C_1 \land C_2) \land (C_3 \land C_4))$$

Where $C_1, C_2, C_3, C_4 \in$ FORM. Essentially, the position of parenthesis does not matter when all C_i's are connected with $\land$; in that case, we are going to simply pretend that parenthesis do not exists.

10.2 Conjunctive Normal Form(CNF)

$$C_1 \land C_2 \land C_3 \land \ldots \land C_m$$

Where every *Clause* C_i is expressed as disjunction of *Literals.*
$C_i = (l_{1_i} \lor l_{2_i} \lor \ldots \lor l_{k_i})$ where $l_{j_i} \in \{p | p \in PROP\} \cup \{\neg p | p \in PROP\}$. Example:

$$\begin{aligned} C_1 &: (p \lor q \lor \neg r) \\ C_2 &: (q \lor \neg s \lor r) \\ C_3 &: (p \lor \neg q \lor s) \\ \varphi &: C_1 \land C_2 \land C_3 \end{aligned}$$

Clauses: C_1, C_2, C_3, literals: $p, \neg p, q, \neg q, r, \neg r, s, \neg s$.

By applying simple distribution of $\vee$, and $\wedge$, we can convert every formula into CNF. For example:

$$\begin{aligned}\varphi &: ((p \wedge q) \vee (r \wedge s))\\ &: (p \vee (r \wedge s)) \wedge (q \vee (r \wedge s))\\ &: ((p \vee r) \wedge (p \vee s)) \wedge ((q \vee r) \wedge (q \vee s))\\ &: (p \vee r) \wedge (p \vee s) \wedge (q \vee r) \wedge (q \vee s)\end{aligned}$$

Theorem 1 *Every Formula φ can be converted into CNF, say φ_{CNF}.*

Now, let us go back to our classification of formula being SAT or UNSAT. The important question here is, how to find if a formula φ is SAT or UNSAT. The naive approach can be to simply go over every possible truth assignment τ, and check if $\varphi(\tau)$ = TRUE or FALSE. But, this approach will not work, if the formula is too large; let $\varphi = p_1 \wedge \neg p_1 \wedge (p_2 \vee p_3 \vee p_4) \wedge \dots$ with $PROP = \{p_1, \dots, p_{1000}\}$, is φ SAT or UNSAT ? As we know $p_1 \wedge \neg p_1 \to \Box$, the formula φ is UNSAT. $\Box$ represents the placeholder for UNSAT.

10.3 Resolution Refutation

Let's consider a statement $\varphi = (\alpha \vee p) \wedge (\neg p \vee \beta)$. Next, using a truth table, it can be shown that $\varphi \to (\alpha \vee \beta)$

α	p	β	φ	$\alpha \vee \beta$	$\varphi \to (\alpha \vee \beta)$
1	1	1	1	1	1
1	1	0	0	1	1
1	0	1	1	1	1
1	0	0	1	1	1
0	1	1	1	1	1
0	1	0	0	0	1
0	0	1	0	1	1
0	0	0	0	0	1

Table 1: Truth table for statement $\varphi \equiv (\alpha \vee p) \wedge (\neg p \vee \beta)$

Therefore, as seen from Table 1,

$$(\alpha \vee p) \wedge (\neg p \vee \beta) \Rightarrow (\alpha \vee \beta)$$

$$\frac{(\alpha \vee p) \wedge (\neg p \vee \beta)}{\alpha \vee \beta} : \text{Resolution}$$

Observe that if $C_1 \wedge C_2 \to C_3$ is VALID then $C_1 \wedge C_2 \leftrightarrow C_1 \wedge C_2 \wedge C_3$ Therefore,

$$(\alpha \vee p) \wedge (\neg p \vee \beta) \leftrightarrow (\alpha \vee p) \wedge (\neg p \vee \beta) \wedge (\alpha \vee \beta)$$

Thus, to prove that a formula in CNF is UNSAT, we can apply repeatedly resolution until a contradiction (e.g. $p \wedge \neg p$) is reached.

Example of Resolution to Prove UNSAT Consider the following statement in CNF.

$$(p \vee q \vee r) \wedge (\neg p \vee q \vee s) \wedge (\neg s \vee r) \wedge (\neg q \vee r) \wedge (\neg q \vee \neg r) \wedge (q \vee \neg r)$$

Denote each clause by C_i where i is the order the clause appears in the statement. In other words, the statement is equivalent to $C_1 \wedge C_2 \wedge \cdots \wedge C_6$.

Resolve C_1 and C_2:

$$\frac{(p \vee q \vee r) \wedge (\neg p \vee q \vee s)}{C_7 : q \vee r \vee s}$$

Resolve C_7 and C_3:

$$\frac{(q \vee r \vee s) \wedge (\neg s \vee r)}{C_8 : q \vee r \vee r}$$

$C_8 \equiv (q \vee r)$

Resolve C_8 and C_4:

$$\frac{(q \vee r) \wedge (\neg q \vee r)}{C_9 : r}$$

Resolve C_5 and C_6:

$$\frac{(\neg q \vee \neg r) \wedge (q \vee \neg r)}{C_{10} : \neg r}$$

Resolve C_9 and C_{10}:

$$\frac{(r) \wedge (\neg r)}{\text{UNSAT} \, \square}$$

Definition 2 *A* ***resolution refutation*** *of a formula φ given in CNF, is a list of clauses $C_1, C_2, \cdots, C_t$ such that either $C_i \in \varphi$ or C_i is derived from C_a, C_b using resolution where $a, b < i$, and the last clause $C_t = \square$.*

In the previous example, the *resolution refutation* is the list of clauses $C_1, C_2, C_3, \cdots, C_{10}, C_{11}$, where $C_{11} = \square$ which refers to a contradiction.

Theorem 3 *A formula φ is UNSAT if and only if there exists a resolution refutation of φ*

With the definition of resolution refutation, and Theorem 3, we can also prove that a formula is valid. A formula φ is valid then $\neg\varphi$ must be UNSAT.

$$\begin{aligned} \text{KB} \models \alpha &\Leftrightarrow (\text{KB} \to \alpha) \text{ is VALID} \\ &\Leftrightarrow (\neg\text{KB} \vee \alpha) \text{ is VALID} \\ &\Leftrightarrow (\text{KB} \wedge \neg\alpha) \text{ is UNSAT} \end{aligned}$$

Thus, to show $KB \models \alpha$, we only need to show $(\text{KB} \wedge \neg\alpha)$ is UNSAT using resolution.

Example of Resolution to Prove VALID Given two statements "all Greeks are mortal" and "all mortals are human"; how resolution can be used to deduce that "all Greeks are human".

$C_1 : g \rightarrow h \equiv \neg g \vee h$
$C_2 : h \rightarrow m \equiv \neg h \vee m$

Thus the agent's knowledge base in CNF is,

$$\text{KB} \equiv (\neg g \vee h) \wedge (\neg h \vee m)$$

An agent want to prove that KB$\models (\neg g \vee m)$ is VALID, or alternatively KB$\wedge\neg(\neg g \vee m)$ is UNSAT.

$\text{KB} \wedge \neg(\neg g \vee m) \equiv (\neg g \vee h) \wedge (\neg h \vee m) \wedge (g) \wedge (\neg m)$

$C_1 = (\neg g \vee h)$
$C_2 = (\neg h \vee m)$
$C_3 = (g)$
$C_4 = (\neg m)$

Resolve C_1 and C_3:

$$\frac{(\neg g \vee h) \wedge (g)}{C_5 : (h)}$$

Resolve C_2 and C_4:

$$\frac{(\neg h \vee m) \wedge (\neg m)}{C_6 : (\neg h)}$$

Resolve C_5 and C_6:

$$\frac{(h) \wedge (\neg h)}{C_7 : \square\text{UNSAT}}$$

KB$\models (g \rightarrow m)$ is VALID.

Concluding Remark However, the order to apply resolution to derive a contradiction is still a challenging task that can take exponential time.