1 Propositional Logic

How could we convert a propositional sentence into CNF?

1. Replace →
2. Use De Morgan’s law to move the ¬ symbol onto literals
3. Continuously use distributive law AND-over-OR.

Example:

\[(A \rightarrow B) \lor \neg((A \land C) \land B)\]

SAT problem for CNF is VERY hard, and it is a million-dollar reward problem. In fact, most of the hard problem can be converted to SAT problem, which means if we could find a solution for SAT, we could find a solution for the original problem.

Questions:

How could we apply the laws listed in the previous notes to transform one sentence to an equivalent another?

How could we prove that two sentences are logical equivalent.

How could we represent a CNF in Java?

We would need literal set, representation for disjunctions of literals, and conjunction of disjunctions.

First Order Logic

Two quantifiers:

\[\exists, \forall\]

The meaning of these two quantifiers will be applied to propositional functions.

The meaning of \(\exists\):

The meaning of \(\forall\):
The sequence of $\exists, \forall$ will make difference.

We could use generalized De Morgan’s law to get

$$\neg(\forall x P(x)) \equiv \exists x \neg(P(x))$$

and

$$\neg(\exists x P(x)) \equiv \forall x \neg(P(x))$$