Subsumption lattice over clauses [Why not implication?]

↳ ① Many nice results for subsumption

② Subsumption is decidable, implication is not. } If $\Sigma \not\models \Box$, Then, Res may not terminate

③ More efficient to implement subsumption

ILP systems are programs that search quasi order sets

[Note: Atom, $\subseteq \;\equiv\; \models$]
Not for clauses

Both $\subseteq$ & $\models$ are quasi orders.

Subsumption over clauses (C).

$$\{\overset{C_1}{mem(A,[A|B]) \leftarrow}, \overset{C_2}{mem(A,[B,A|C]) \leftarrow}\}$$

$$\succeq$$

$$\{\underset{D_1}{mem(1,[1,2]) \leftarrow}, \underset{D_2}{mem(2,[1,2]) \leftarrow}\} \Big\} \text{Set of equivalent clauses}$$

↳ can be proved to be a quasi order.

↳ C_E has partial order

↳ Q: When are two clauses subsume equivalent?

Subsume equivalence

↳. If C_1 is C_2 but with duplicate literals removed, $C_1 \equiv C_2$

$$\{P(x) \vee Q(a)\} \equiv \{P(x) \vee Q(a) \vee P(x)\}$$

↳ Order of literals in C_1 & C_2 does not matter

$$\{P(a) \vee Q(b)\} \equiv \{Q(a) \vee P(b)\}$$

↳. What about?

$$\{P(x,x)\} \stackrel{?}{\equiv} \{P(x,x), P(x,y)\}$$

What abt $\{P(x,x), P(x,x), P(x_1, x_2), P(x_2, x_3) \ldots P(x_{n-1}, x_n)\}$

↳ Of course, variants are subsume equivalent

But for clauses, eqn goes much beyond variants

Reduced clause:

C is reduced if $\nexists$ no $D \subset C$ st

$C \equiv D$

From previous example

$\{P(x,x), P(x,y)\}$ is not reduced

<u>But</u>

$\{P(x,x)\}$ is reduced.

<u>Also</u>:

$\{P(x,y), P(y,x)\}$ is reduced

Goal :- Procedure to come up with ... canonical member of ≡

INPUT: A clause C.
OUTPUT: A reduction D of C.
Set $D = C$, $\theta =$;
repeat
  Set D to $D\theta$;
  Find a literal $\mathbf{l} \in D$ and a substitution θ such that $D\theta \subseteq D \setminus \{\mathbf{l}\}$;
until Such a $(\mathbf{l}, \theta)$ does not exist;
return D.

Plotkin's reduction algorithm