1 Box model

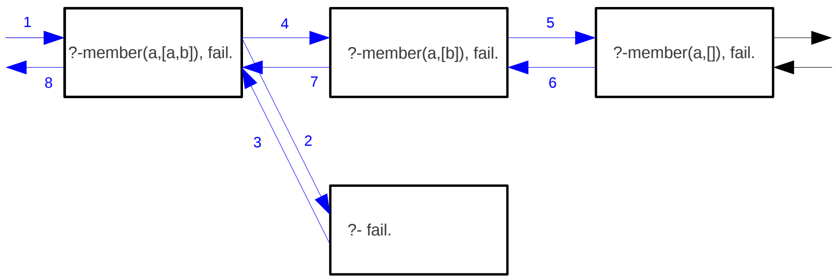
Exercise 1.1: Let's have the following program and queries. Demonstrate the processing of the queries using the box model representation.

```
member(X,[X|_]).
member(X,[_|T]) :- member(X,T).

?- member(a,[b]).
?- member(a,[b,a]).
?- member(a,[a,b]), fail.
```

Solution 1.1: Simplified (not nested) demonstration of the processing of the query

```
?- member(a,[a,b]), fail.
```



In addition to this drawings it is possible to look at the ports of boxes directly using tracing tools in Prolog. For example:

```
?- trace.
?- member(a,[b,a]).
        1      1 Call: member(a,[b,a]) ?
        2      2 Call: member(a,[a]) ?
        2      2 Exit: member(a,[a]) ?
        1      1 Exit: member(a,[b,a]) ?
yes
```

2 Metainterpreters, backward and forward chaining

Exercise 2.1: Write a metainterpreter for backward chaining

- for Prolog clauses with conjunction and disjunction
- for rules in a form rule(LefthandSide, RighthandSide) where LefthandSide is a list of Prolog predicates (a comma means conjunction) and RighthandSide is a predicate.

Solution 2.1:

- ```
  prove(true).
  prove((A;B)) :- prove(A) ; prove(B).
  prove((A,B)) :- prove(A), prove(B).
  prove(H) :- clause(H,B), prove(B).
  ```
- ```
  prove([]).
  prove([H|T]) :- prove(H), prove(T).
  prove(RHS) :- rule(LHS,RHS), prove(LHS).
  ```

Exercise 2.2: Write a forward chaining interpreter for rules in a form rule(LHS, RHS).

Solution 2.2:

```
fc(K1,K3) :- step(K1,K2), fc(K2,K3).
fc(K1,K1).

step(K1,K2) :-
 rule(L,R),
 true_in(L,K1),
 not true_in(R,K1),
 append(K1,[R],K2).

true_in([],_).
true_in([H|T],K) :-
 true_in(H,K),
 true_in(T,K).

true_in(X,K) :- member(X,K).
```

Exercise 2.3: Rewrite the Prolog rules below into the form for the forward chaining interpreter and simulate its behaviour for these two facts: `d.` `e.`

```
a :- d, e, b.
b :- c, e.
b :- a, c.
c :- d.
```

Solution 2.3:

```
rule([d,e,b],a).
rule([c,e],b).
rule([a,c],b).
rule([d],c).

?- fc([d,e],X).
```

3 SAT: Davis Putnam (DP), DPLL

Exercise 3.1: Is the following set of clauses satisfiable? (Use DP algorithm to find the solution).

$$\begin{aligned} S &= \{\{P,Q,R\},\{P,\neg Q,\neg R\},\{P,\neg W\},\{\neg Q,\neg R,\neg W\},\{\neg P,\neg Q,R\},\\ &\quad \{U,X\},\{U,\neg X\},\{Q,\neg U\},\{\neg R,\neg U\}\} \end{aligned}$$

Solution 3.1: DP Algorithm (one of the possibilities):

- input: a formula in CNF without tautologies (set representation)
- repeat while there are any variables and $\square$ is not in the set of clauses:
  - choose a variable
  - create all the resolvents using clauses that contain the chosen variable, add the non-tautologic resolvents into the set of clauses
  - discard all the clauses containing the chosen variable
- output:
  - SAT when the set of clauses is empty (NO CLAUSES),
  - UNSAT when there is $\square$ in the set (EMPTY CLAUSE)

a) chosen variable: W
clauses containing: $\{P,\neg W\},\{\neg Q,\neg R,\neg W\}$
resolvents: no
new set: $\{\{P,Q,R\},\{P,\neg Q,\neg R\},\{\neg P,\neg Q,R\},\{U,X\},\{U,\neg X\},\{Q,\neg U\},\{\neg R,\neg U\}\}$

b) chosen variable: X
clauses containing: $\{U,X\},\{U,\neg X\}$
resolvents: $\{U\}$
new set: $\{\{P,Q,R\},\{P,\neg Q,\neg R\},\{\neg P,\neg Q,R\},\{U\},\{Q,\neg U\},\{\neg R,\neg U\}\}$

c) chosen variable: U
clauses containing: $\{U\},\{Q,\neg U\},\{\neg R,\neg U\}$
resolvents: $\{Q\},\{\neg R\}$
new set: $\{\{P,Q,R\},\{P,\neg Q,\neg R\},\{\neg P,\neg Q,R\},\{Q\},\{\neg R\}\}$

d) chosen variable: Q
clauses containing: $\{P, Q, R\}, \{P, \neg Q, \neg R\}, \{\neg P, \neg Q, R\}, \{Q\}$
resolvents: $\{\neg P, R\}, \{P, \neg R\}, \{P, R, \neg R\}, \{P, \neg P, R\}$
tautologies: $\{P, R, \neg R\}, \{P, \neg P, R\}$
new set: $\{\{\neg R\}, \{\neg P, R\}, \{P, \neg R\}\}$

e) chosen variable: P
clauses containing: $\{\neg P, R\}, \{P, \neg R\}$
resolvents: $\{R, \neg R\}$
tautologies: $\{R, \neg R\}$
new set: $\{\{\neg R\}\}$

f) chosen variable: R
clauses containing: $\{\neg R\}$
resolvents: no
new set: empty (NO CLAUSES)

Conclusion: SAT (the set S is satisfiable).

Exercise 3.2: Is the following set of clauses satisfiable? (Use DPLL algorithm to find the solution).

$$S = \{\{P, \neg Q, \neg R\}, \{R, \neg Q\}, \{\neg P, \neg Q\}, \{P, \neg R\}, \{P, R\}, \{R\}, \{Q, \neg P, Q\}\}$$

Solution 3.2: DPLL Algorithm: until SAT or UNSAT can be used, apply the rest of the following rules to S:

- UNSAT: $\square$ is an element of S
- SAT: S is empty
- MULT: eliminate duplicate literals within one clause
- SUBS: discard a clause that is a superset of another clause
- UNIT: discard the literal $\neg L$ in all clauses when S contains the clause $\{L\}$
- TAUT: discard a clause that is a tautology (contains both L and $\neg L$)
- PURE: discard all clauses containing L when there is no occurrence of $\neg L$ in S
- SPLIT: discard all clauses containing either L or $\neg L$ and add all their possible resolvents

a) $\{\{P, \neg Q, \neg R\}, \{R, \neg Q\}, \{\neg P, \neg Q\}, \{P, \neg R\}, \{P, R\}, \{R\}, \{Q, \neg P, Q\}\}$ MULT

b) $\{\{P, \neg Q, \neg R\}, \{R, \neg Q\}, \{\neg P, \neg Q\}, \{P, \neg R\}, \{P, R\}, \{R\}, \{\neg P, Q\}\}$ SUBS

c) $\{\{P, \neg Q, \neg R\}, \{R, \neg Q\}, \{\neg P, \neg Q\}, \{P, \neg R\}, \{R\}, \{\neg P, Q\}\}$ UNIT R

d) $\{\{P, \neg Q\}, \{R, \neg Q\}, \{\neg P, \neg Q\}, \{P\}, \{R\}, \{\neg P, Q\}\}$ PURE R

e) $\{\{P, \neg Q\}, \{\neg P, \neg Q\}, \{P\}, \{\neg P, Q\}\}$ UNIT P

f) $\{\{P, \neg Q\}, \{\neg Q\}, \{P\}, \{Q\}\}$ PURE P

g) $\{\{\neg Q\}, \{Q\}\}$ SPLIT Q

h) $\{\square\}$ UNSAT

Conclusion: UNSAT (the set S is unsatisfiable).