AN EARLY (1971) "CONVERSATION"

USER

Cats kill mice.
Tom is a cat who does not like mice who eat cheese.
Jerry is a mouse who eats cheese.
Max is not a mouse.
What does Tom do?

COMPUTER

Tom does not like mice who eat cheese.
Tom kills mice.

USER
Who is a cat?
COMPUTER
Tom.
USER
What does Jerry eat?
COMPUTER
Cheese.
USER
Who does not like mice who eat cheese?
COMPUTER
Tom

USER

What does Tom eat?

COMPUTER

What cats who do not like mice who eat cheese eat.

ANOTHER CONVERSATION

USER Every psychiatrist is a person.
Every person he analyzes is sick.
Jacques is a psychiatrist in Marseille.
Is Jacques a person?
Where is Jacques?
Is Jacques sick?

COMPUTER

Yes.
In Marseille.
I don't know.

LOGIC PROGRAMMING

- A program is a collection of axioms, from which theorems can be proven.
- A goal states the theorem to be proved. A logic programming language implementation attempts to satisfy the goal given the axioms and built-in inference mechanism.

HORN CLAUSES

- A standard form for writing axioms, e.g.:

  father(X, Y) ← parent(X, Y), male(X).

- The Horn clause consists of:

  - A head or consequent term H, &
  - A body consisting of terms B_i:

  $$H \leftarrow B_0, B_1, \ldots, B_n$$

- The semantics is:

  "If $B_0, B_1, \ldots, B_n$, then H".

TERMS

Constants

rpi

troy

Variables

University

City

Predicates

located_at(rpi, troy)

pair(a, pair(b,c)) can be nested.

RESOLUTION

- To derive new statements, Robinson's resolution principle says that if two Horn clauses:

$$H_1 \leftarrow B_{11}, B_{12}, \ldots, B_{1m}$$

$$H_2 \leftarrow B_{21}, B_{22}, \ldots, B_{2n}$$

are such that H_1 matches B_{2i}, then:

$$H_2 \leftarrow B_{21}, \ldots, B_{2(i-1)}, \underbrace{B_{11}, B_{12}, \ldots, B_{1m}}, B_{2(i+1)}, \ldots, B_{2n}$$

we can replace B_{2i} with $B_{11}, \ldots, B_{1m}$.

e.g.:

$$\frac{\begin{array}{c} C \leftarrow A, B \\ D \leftarrow C \end{array}}{D \leftarrow A, B}$$

RESOLUTION EXAMPLE

father (X, Y) :- parent (X, Y), male (X).

ancestor (X, Y) :- father (X, Y).

---

ancestor (X, Y) :- parent (X, Y), male (X).

":-" is Prolog's notation for $\leftarrow$.

UNIFICATION

- During resolution, free variables acquire values through unification with expressions in matching terms.

e.g.:

male(carlos)
parent(carlos, tatiana)
father(X, Y) :- parent(X, Y), male(X)

---

father(carlos, tatiana)

UNIFICATION PROCESS

In Prolog,

- A constant unifies only with itself.
- Two predicates unify iff they have the same functor, the same number of arguments, and the corresponding arguments unify.
- A variable unifies with anything. If the other thing has a value, then the variable is instantiated. If it is an uninstantiated variable, the two variables are associated.

PROLOG LISTS

[a, b, c] is syntactic sugar

for .(a, .(b, .(c, [])))
where [] is the empty list,
and . is a built-in cons-like functor

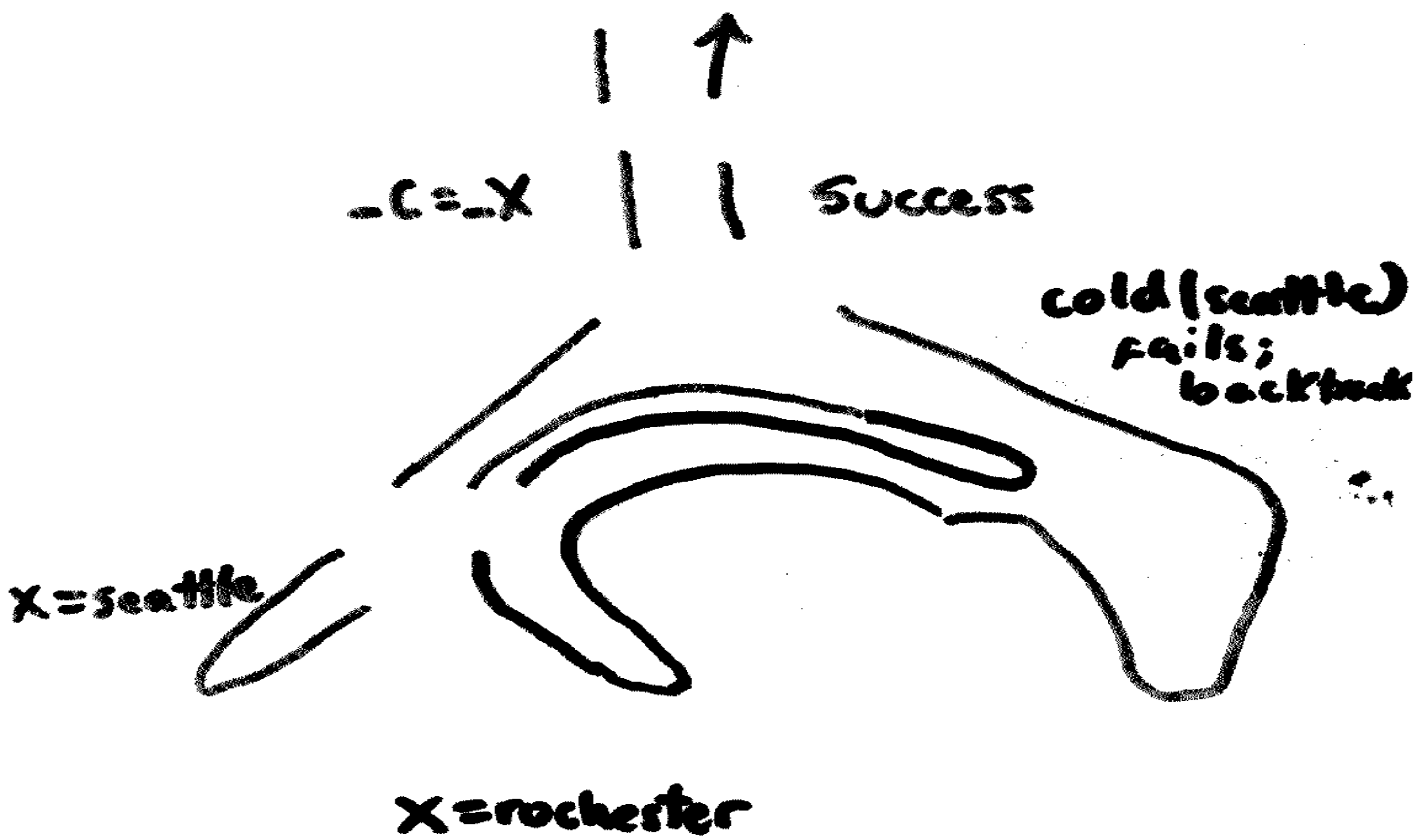
[a, b, c] can also be expressed as:

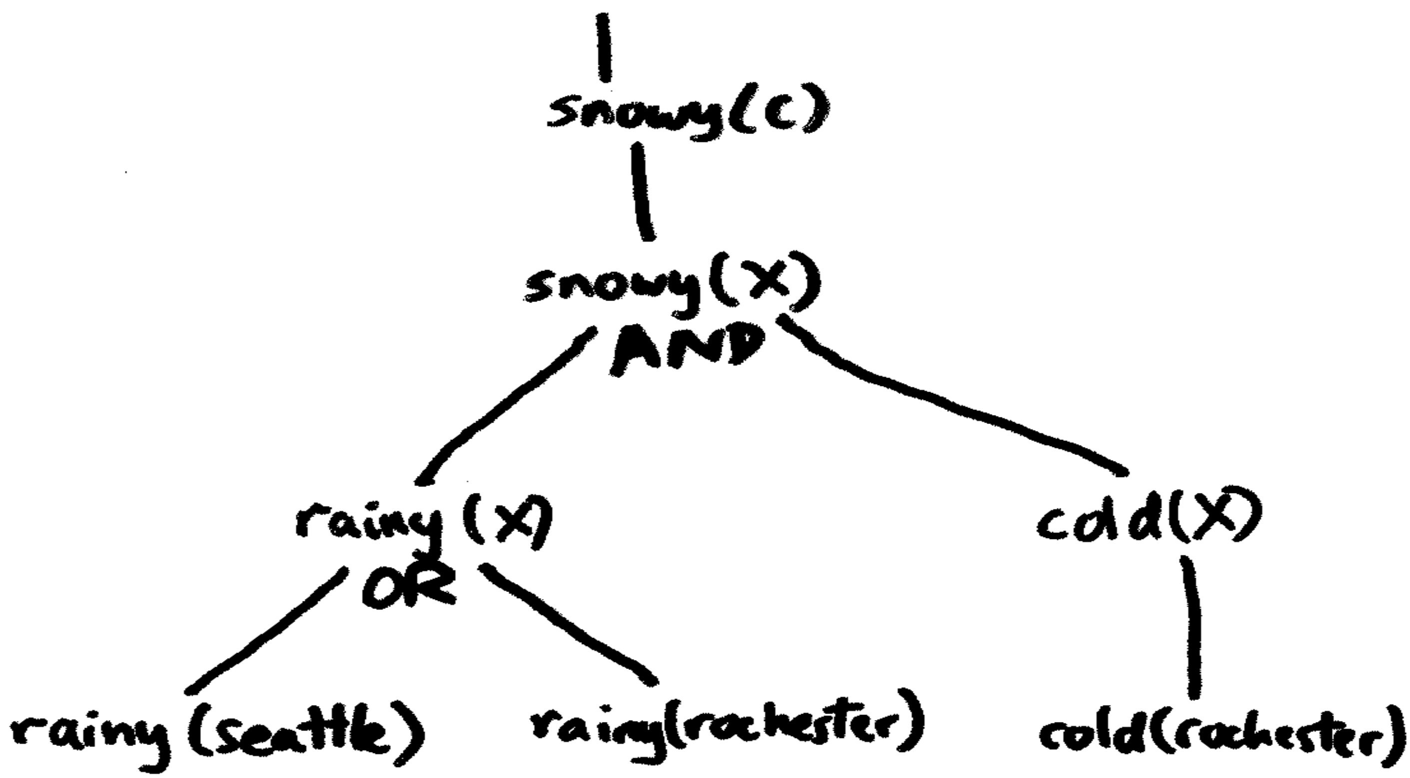
[a | [b, c]] , or
[a, b | [c]] , or
[a, b, c | []]

e.g. "append([], A, A).
append([H|T], A, [H, L]) :-
append(T, A, L) "

BACKTRACKING

- Forward chaining goes from axioms forward into goals.
- Backward chaining goes from goals and works backward to prove them with existing axioms.


_C=_X
Success
cold(seattle)
fails;
backtrack
X=seattle
X=rochester


snowy(c)
snowy(X)
AND
rainy(X)
OR
cold(X)
rainy(seattle)
rainy(rochester)
cold(rochester)

BACKTRACKING

```
rainy(seattle).
rainy(rochester).
cold(rochester).
snowy(X) :- rainy(X), cold(X).
```