

## Prolog Implementation

- Prolog uses the following search
  - Goals are solved left to right
  - For each goal, the database is searched top to bottom

1

---

---

---

---

---

---

---

---

## Example

1. a :- b, c, d.
  2. a :- e, f.
  3. b :- f.
  4. e.
  5. f.
  6. a :- f.
- ?- a, e.

2

---

---

---

---

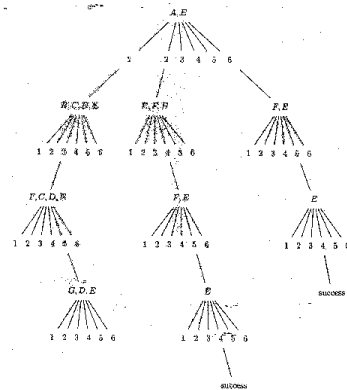
---

---

---

---

## Tree



---

---

---

---

---

---

---

---

### Example

1. a :- b, c, d.
  2. a :- e, f.
  3. b :- f.
  4. e.
  5. f.
  6. a :- f, a.
- ?- a, e.

4

---

---

---

---

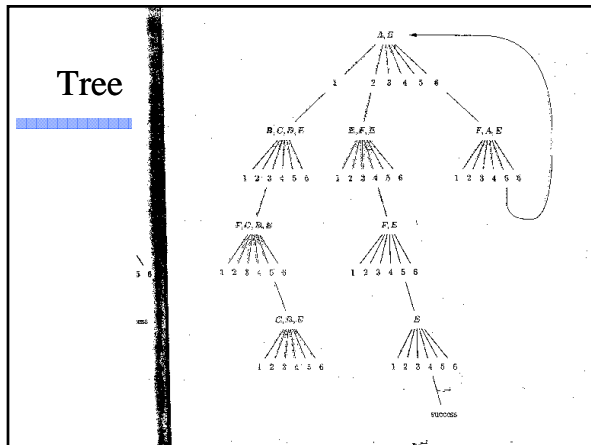
---

---

---

---

### Tree



---

---

---

---

---

---

---

---

### Example

1. a :- f, a.
  2. a :- b, c, d.
  3. a :- e, f.
  4. b :- f.
  5. e.
  6. f.
- ?- a, e.

6

---

---

---

---

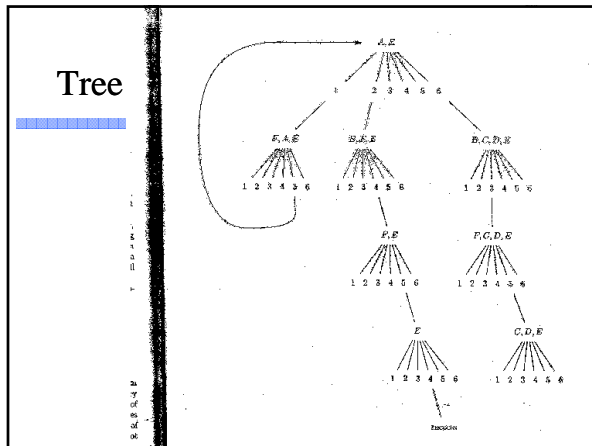
---

---

---

---

## Tree




---

---

---

---

---

---

---

---

## Cut

1. a :- c, l, c.
2. a.
3. b :- a.
4. b :- c.
5. c.
6. c :- d.
- ?- b.

8

---

---

---

---

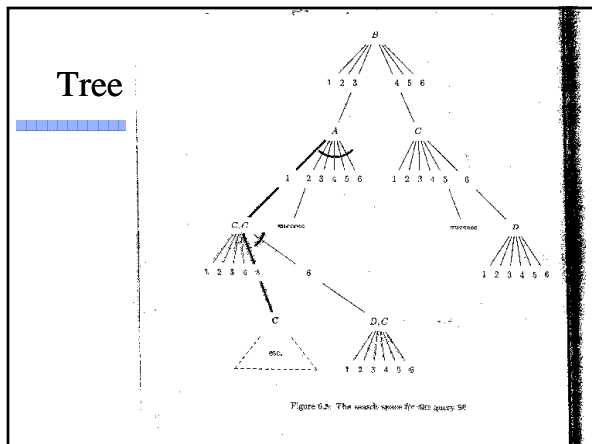
---

---

---

---

## Tree




---

---

---

---

---

---

---

---

## fail

**a :- b, fail, c.**

**d(a) :- !, fail.  
d(X).**

10

---

---

---

---

---

---

---

---

## Lists

### ML

- [], nil
- A :: nil
- x :: y
- A :: x :: y

### Prolog

- []
- A . []
- [X | Y]
- [a, X | Y]

11

---

---

---

---

---

---

---

---

## Arithmetic

**factorial(0, 1).**

**factorial(N, Value) :-**

**N > 0, K is N - 1, factorial(K, Kfact),  
    Value is Kfact \* N, !.**

12

---

---

---

---

---

---

---

---

## Functors

Function symbol for building data structures

```
pred1(f(X)).  
pred2(f(X)) :- pred1(X).
```

13

---

---

---

---

---

---

---

---

## Functor Example

```
height(leaf(X), 0).  
height(node(L,R), H) :-  
    height(L, H1), height(R, H2),  
    H is max(H1, H2) + 1.
```

```
?- height(node(node(leaf(1),leaf(2)),  
            leaf(3)), X).
```

14

---

---

---

---

---

---

---

---

## Unification

```
Unify( c(X, c(Y, c(Z, n))), c(he, c(she, c(it, n))) )
```

```
[X = he, Y = she, Z = it]
```

```
Unify( c(we, n), c(X, Y) )
```

```
[X = we, Y = n]
```

```
Unify( c(they, c(you, n)), c(You, c(X, n)) )
```

```
fails
```

```
Unify( c(X, c(she, c(it, n))), c(he, c(she, c(Y, n))) )
```

```
[X = he, Y = it]
```

```
Unify( a(X, b, Y), a(Z, Z, Z) )
```

```
[X = b, Y = b, Z = b]
```

15

---

---

---

---

---

---

---

---

## Unification

Given: terms  $s, t$

Find: most general unifying substitution  $\sigma$ , such that

$$s \sigma = t \sigma$$

16

---

---

---

---

---

---

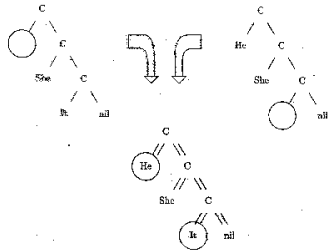
---

---

## Unification

16.6. UNIFICATION

206



---

---

---

---

---

---

---

---

## The Occurs Check

`append([], Y, Y).`

`append([H | T], Y, [H | Z]) :-  
append(T, Y, Z).`

`?- append([], E, [a, b | E]).`

`E = Y implies Y = [a, b | Y].`

18

---

---

---

---

---

---

---

---