Inference in first order logic

- Most popular method:
  resolution refutation
- Large steps based on and-elimination and Modus Ponens, as in propositional logic
- Need to handle **logical variables**:
  substitution and unification

Deductive System: example

"The US law says that it is a crime for an American to sell weapons to hostile nations.

Nono, a country which is US's enemy, has some missiles, and all of them were sold by colonel West, who is American".

How to **formally** prove that colonel West is a criminal?

Step 1: representation...

...it is a crime for an american to sell weapons to hostile nations...

(1) ∀ x,y,z Amer(x) AND Weapon(y) AND Nation(z) AND Hostil(z) AND Vende(x,z,y) → Crim(x)

...Nono...has some missiles...

(2) ∃ x Owns(Nono,x) AND Missil(x)

...all missiles were sold by Colonel West...

(3) ∀ x Owns(Nono,x) AND Missil(x) → Sells(West,Nono,x)

(4) ∀ x Missil(x) → Weapon(x)

(5) ∀ x Enemy(x,US) → Hostile(x)

Facts:

(6) American(West)

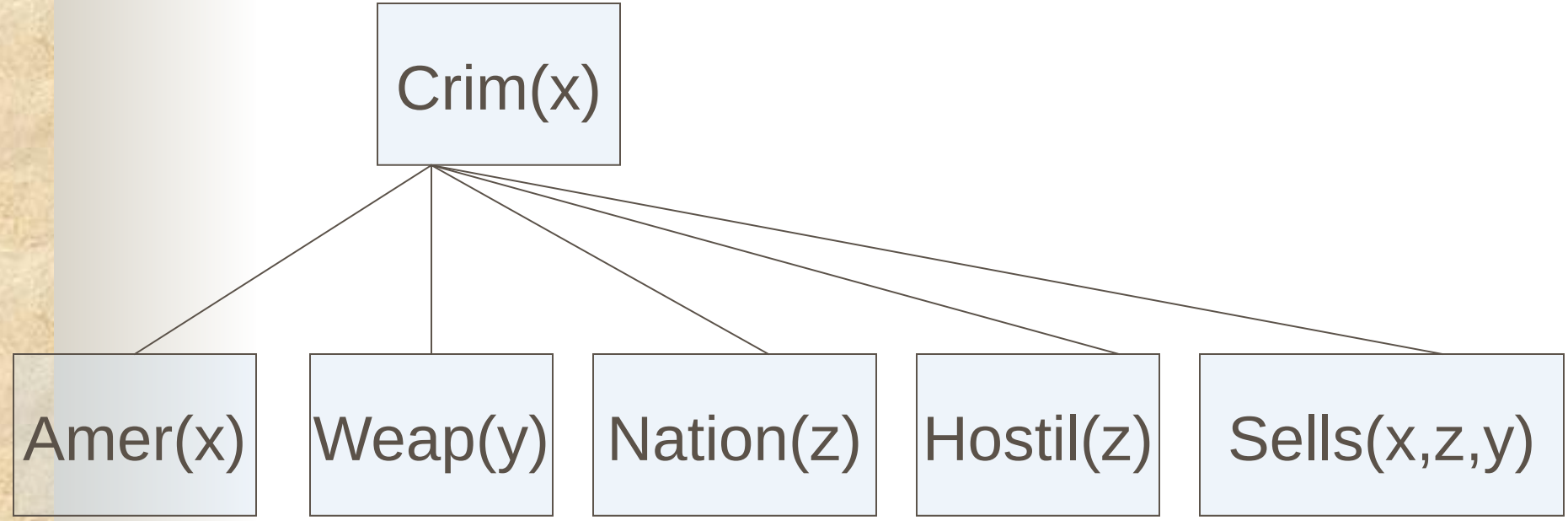
(7) Nation(Nono)

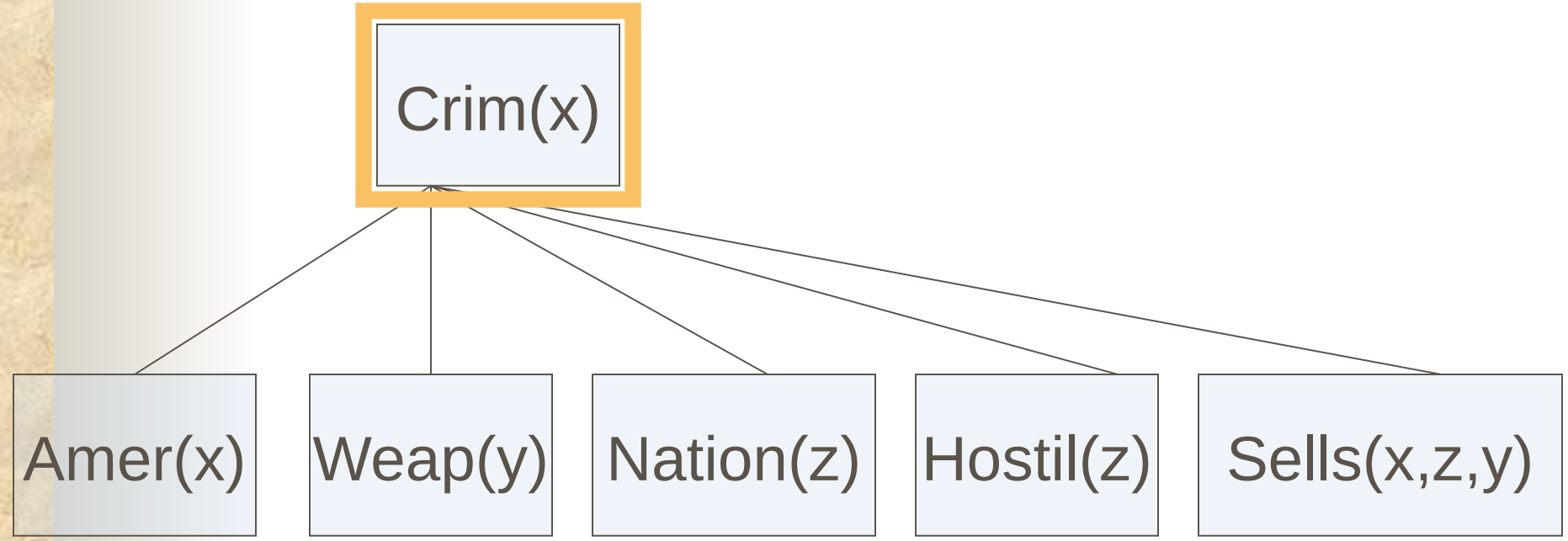
(8) Enemy(Nono,US)

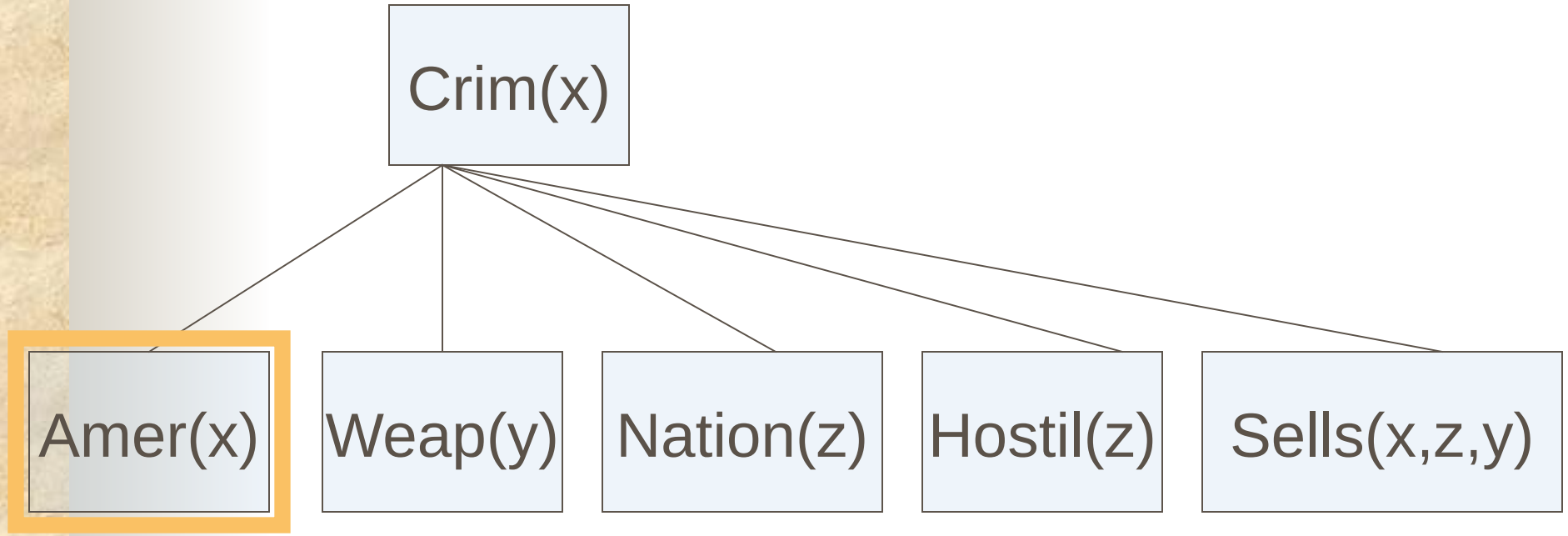
(9) Nation(US)

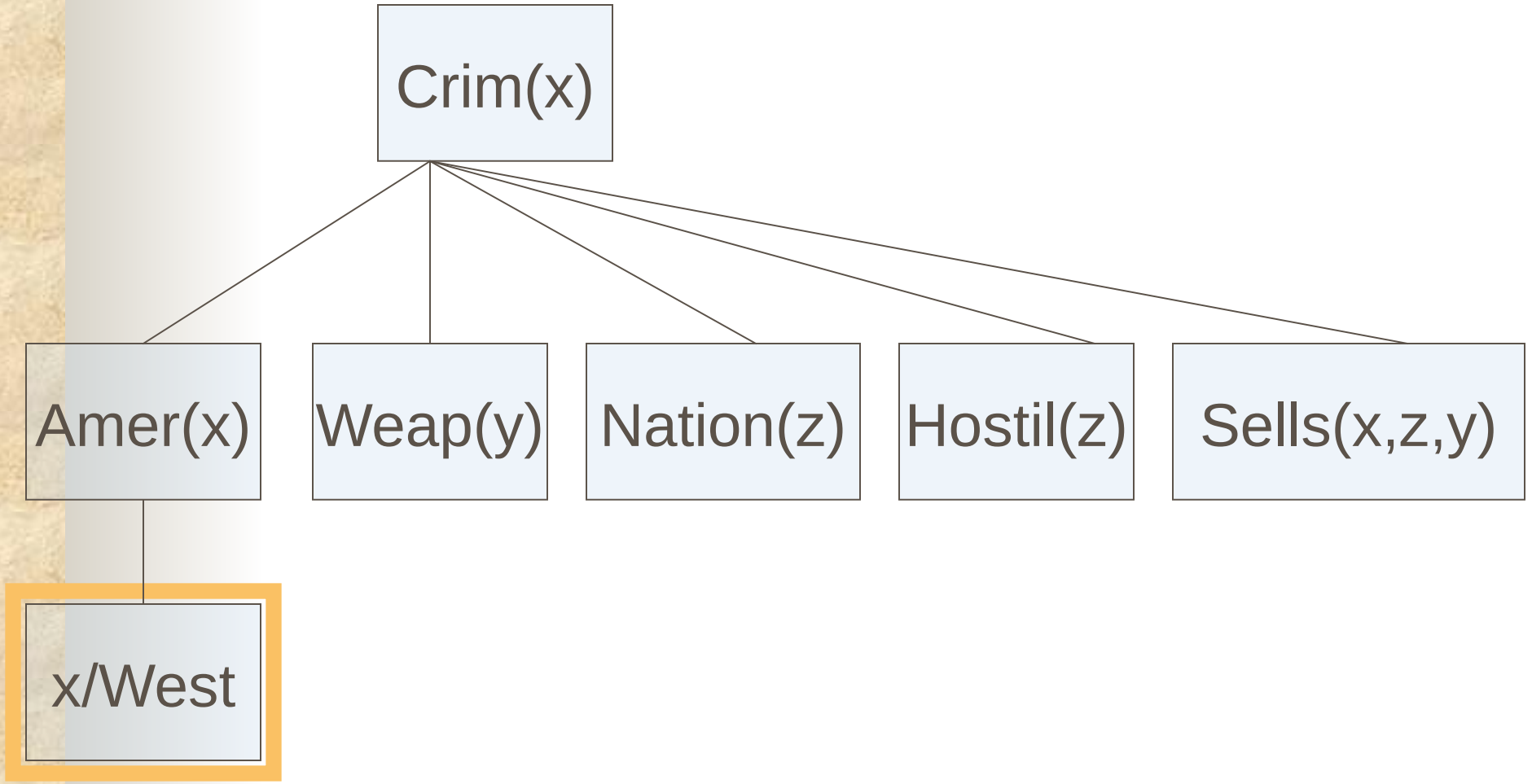
(10) Weapon(M1)

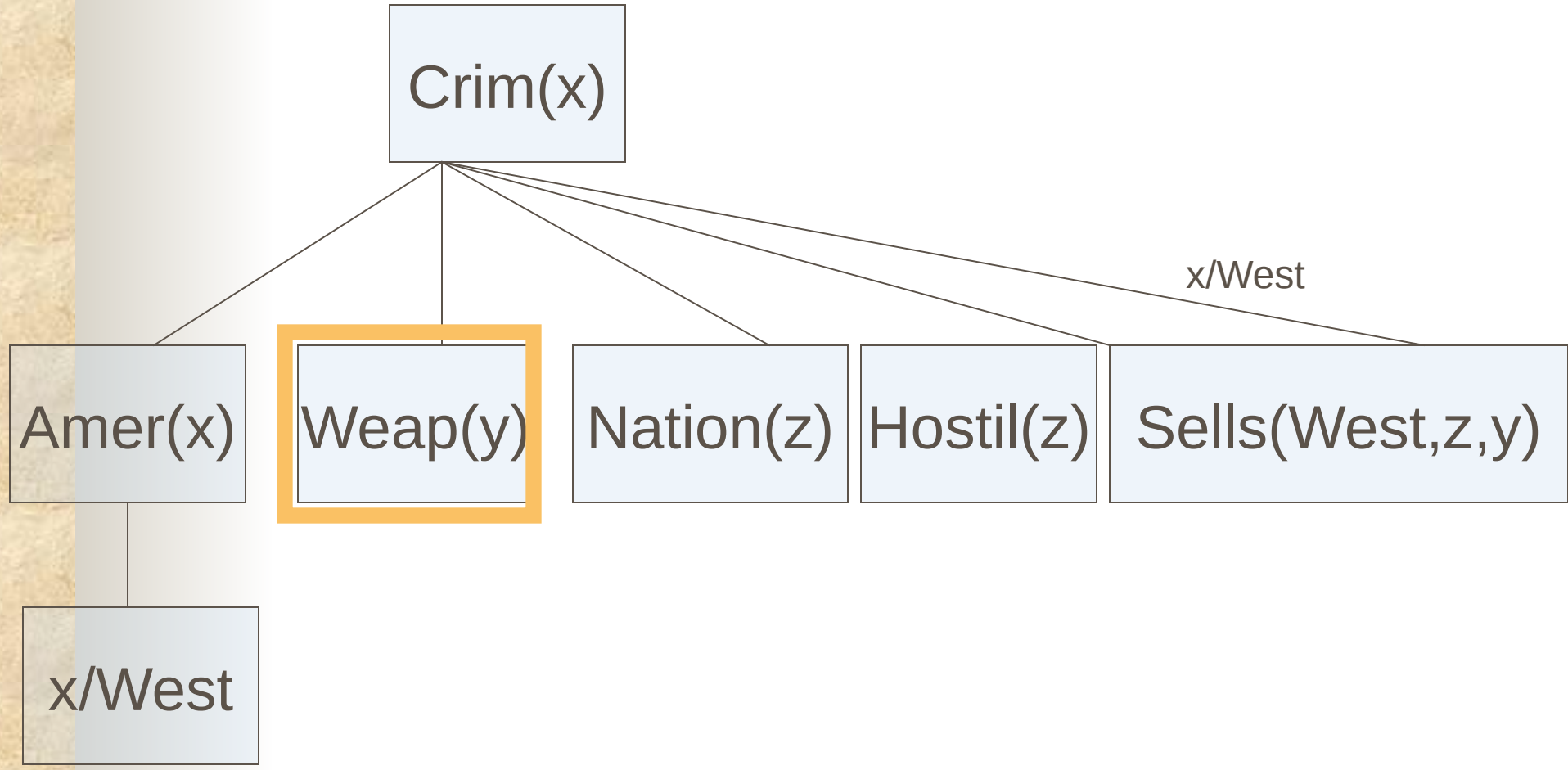
Step 2: inference...

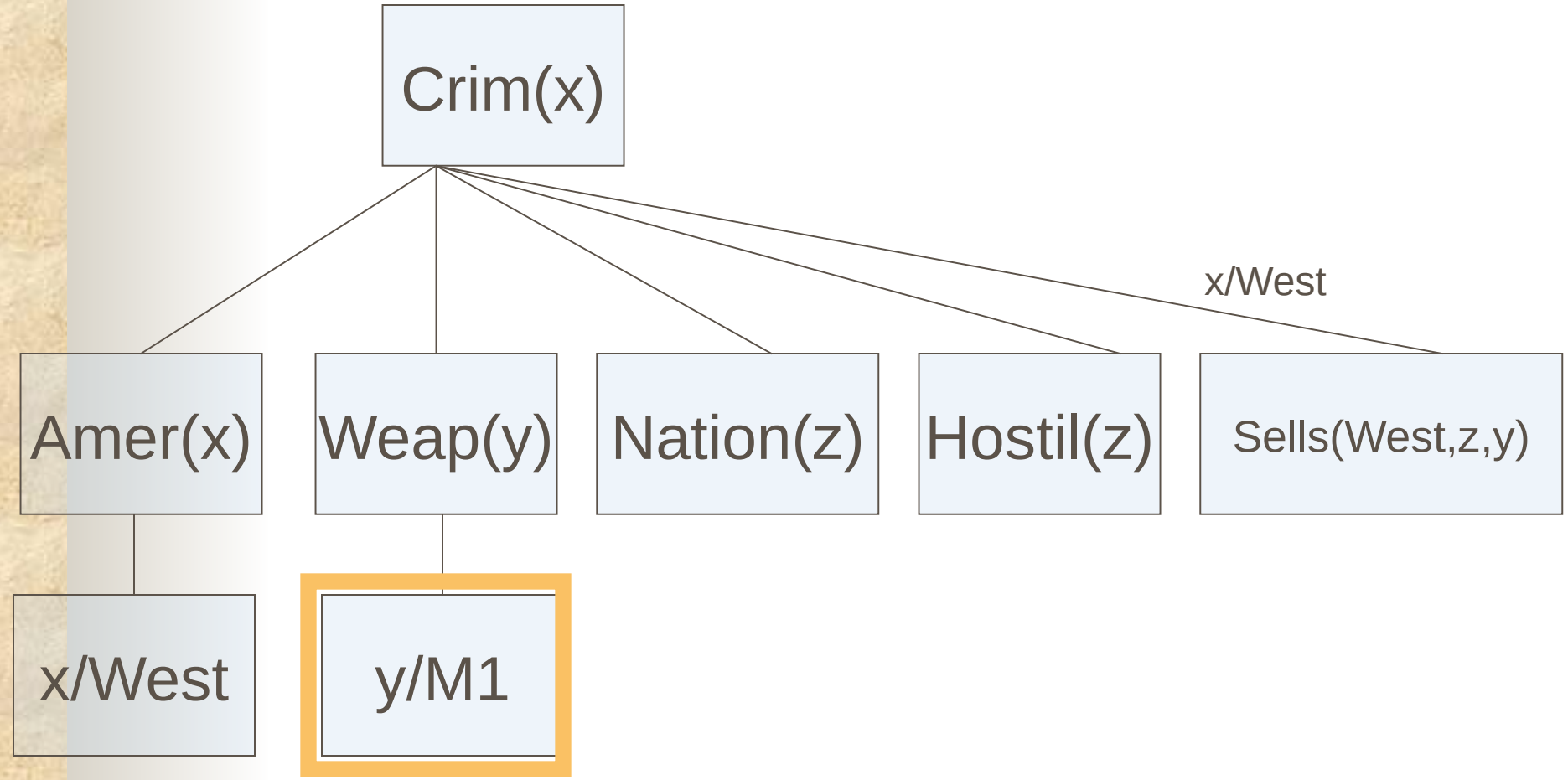


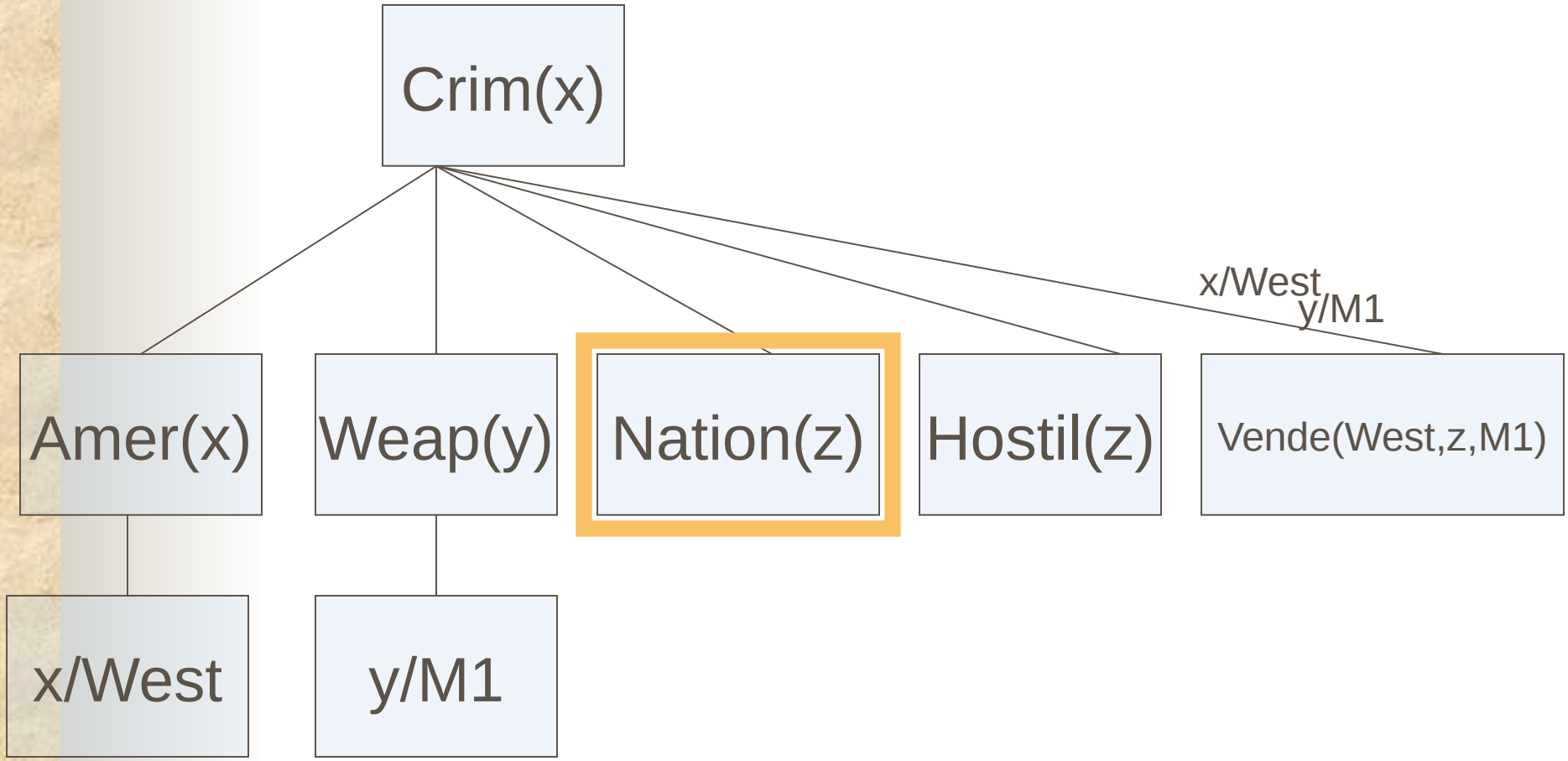

Crim(x)
Amer(x)
Weap(y)
Nation(z)
Hostil(z)
Sells(x,z,y)

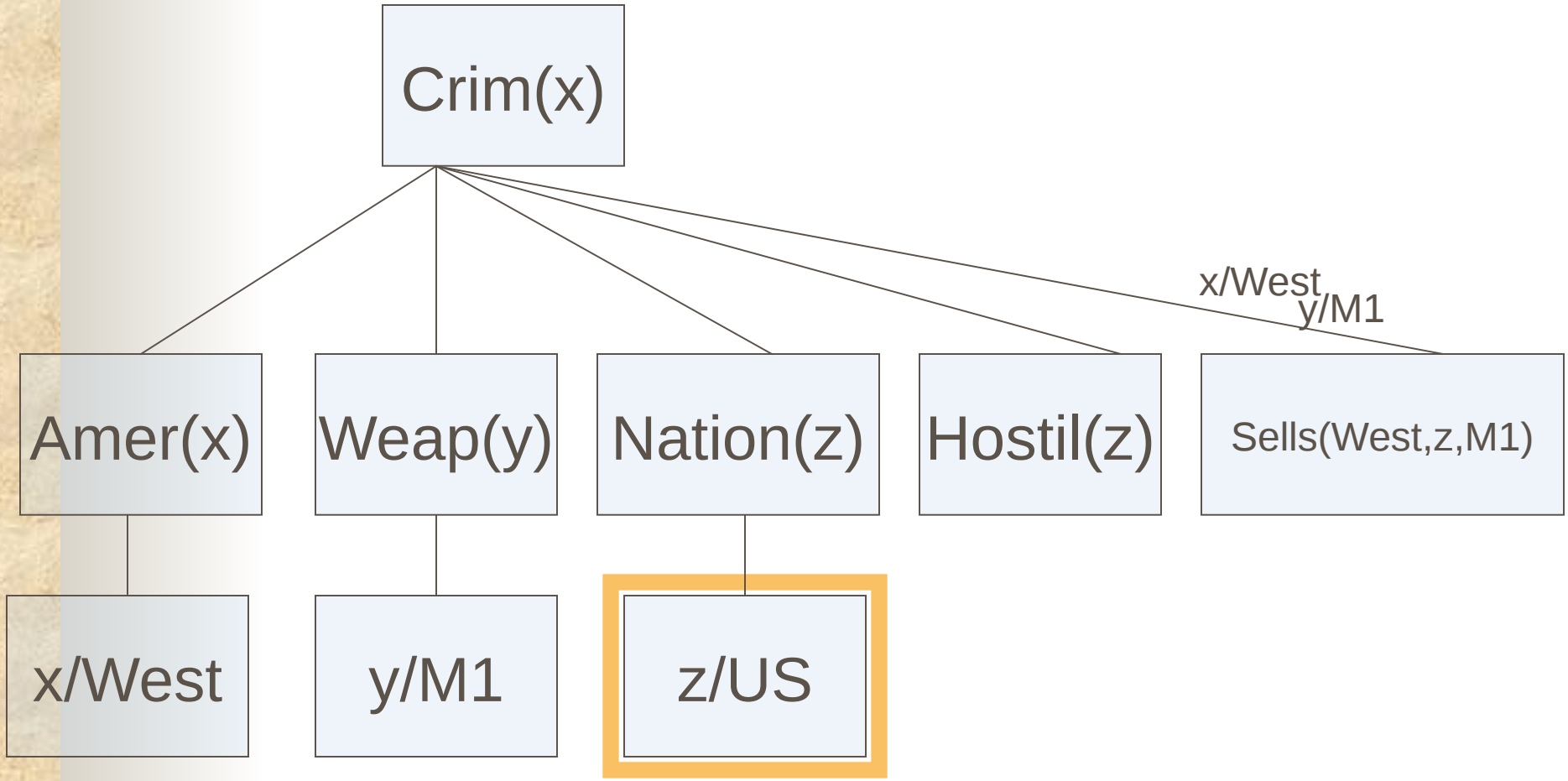

Crim(x)
Amer(x)
Weap(y)
Nation(z)
Hostil(z)
Sells(x,z,y)

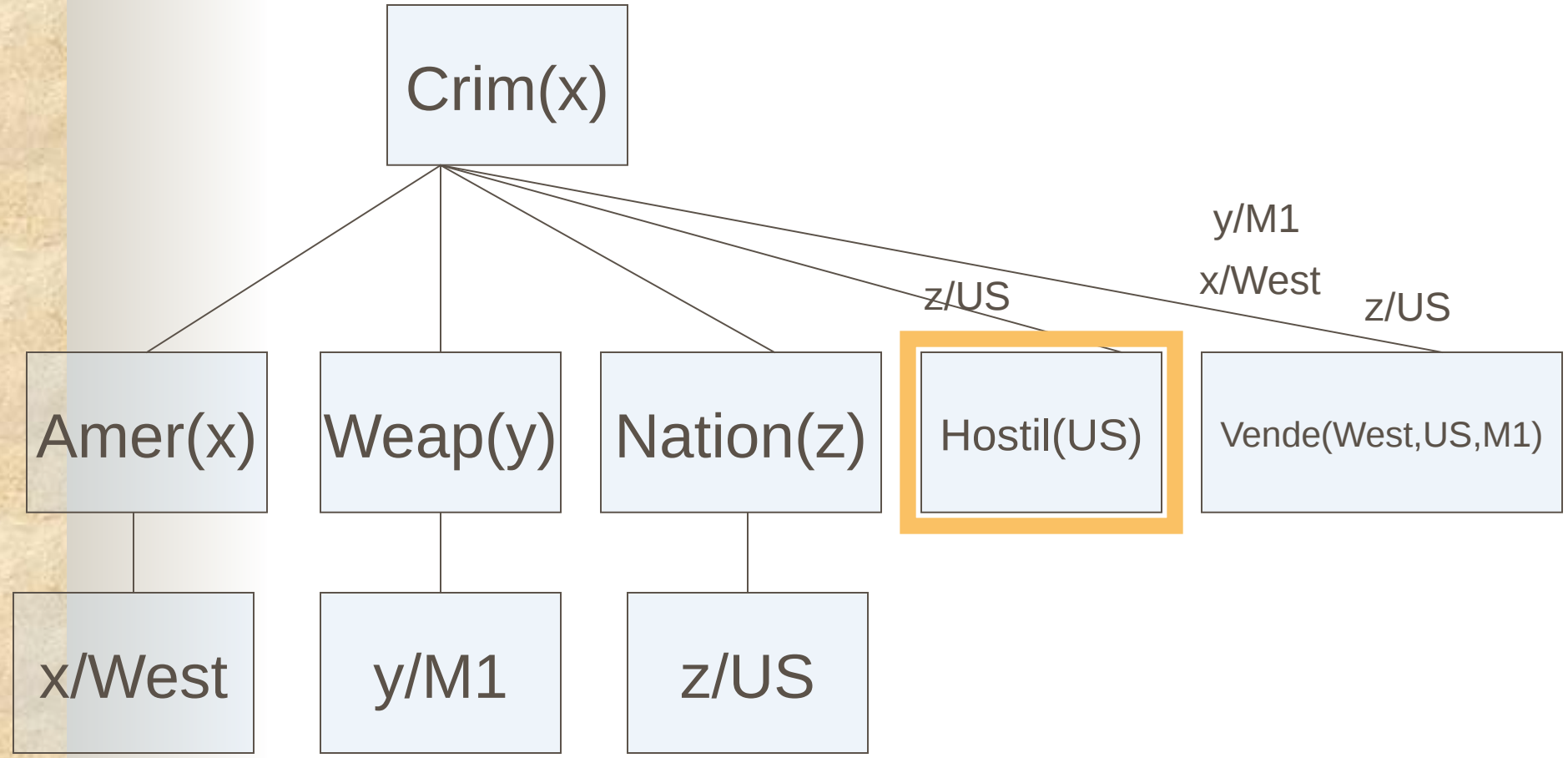

Crim(x)
Amer(x)
Weap(y)
Nation(z)
Hostil(z)
Sells(x,z,y)
x/West

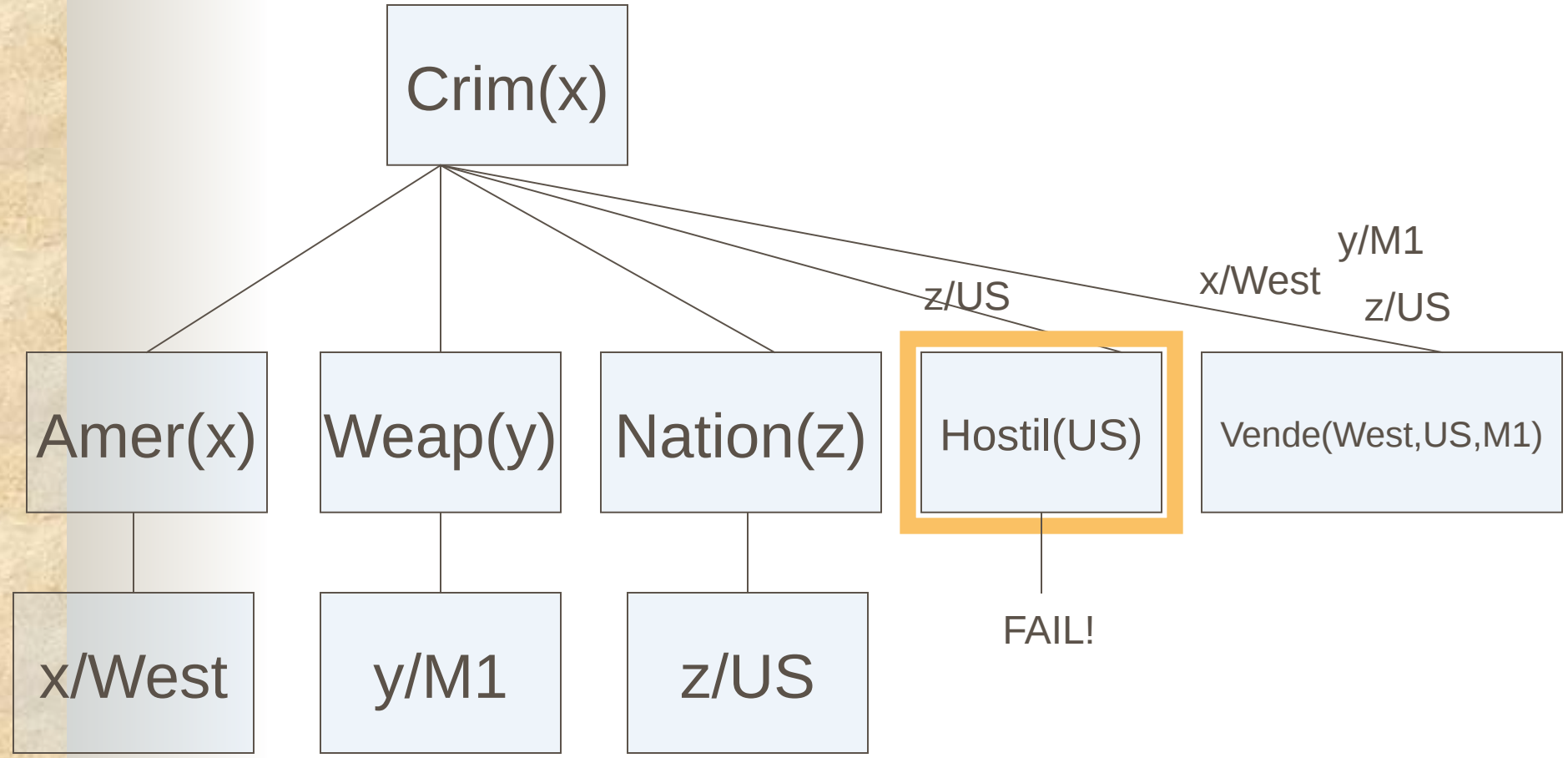

Crim(x)
x/West
Amer(x)
Weap(y)
Nation(z)
Hostil(z)
Sells(West,z,y)
x/West

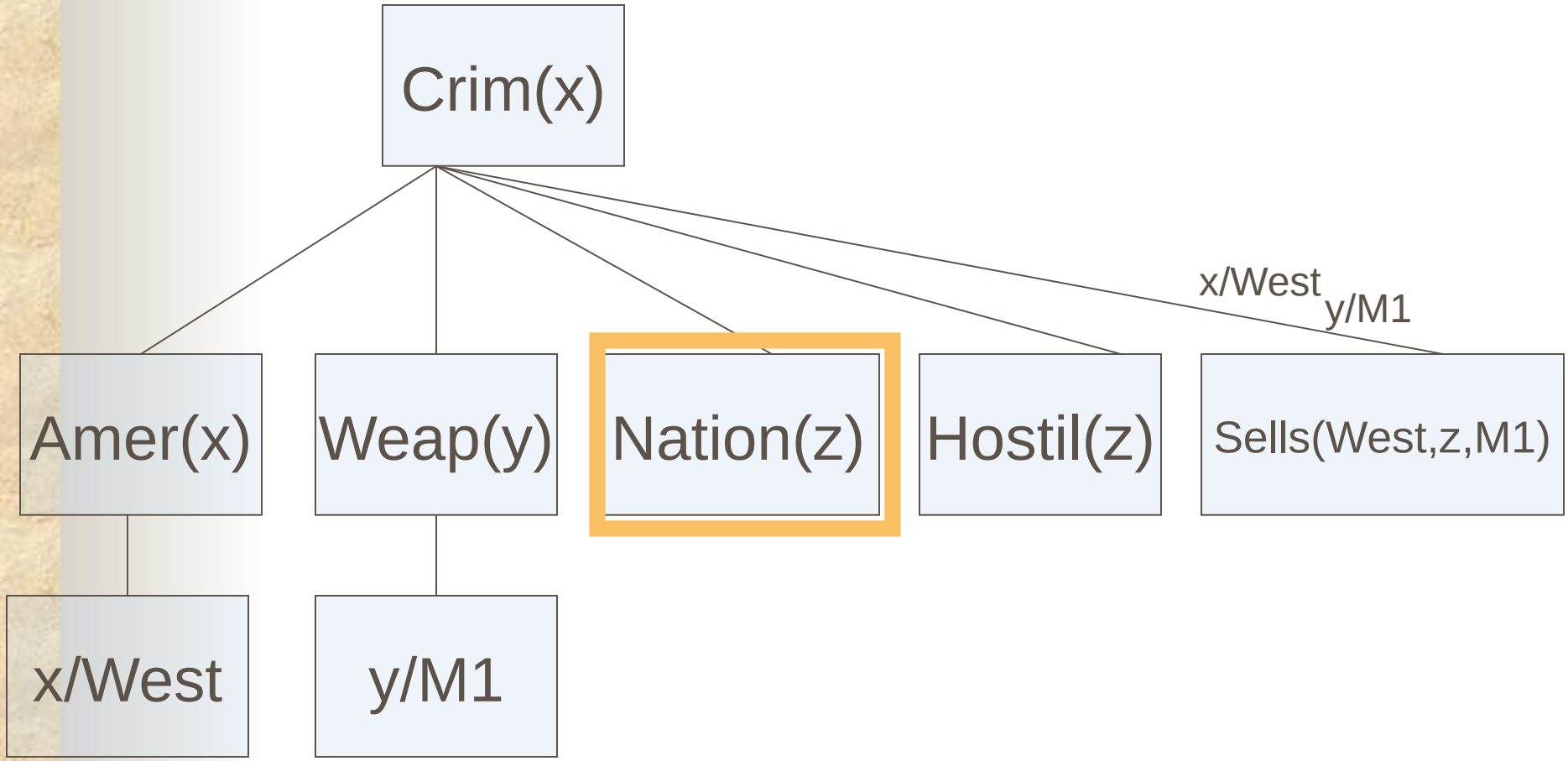

Crim(x)
x/West
Amer(x)
Weap(y)
Nation(z)
Hostil(z)
Sells(West,z,y)
x/West
y/M1

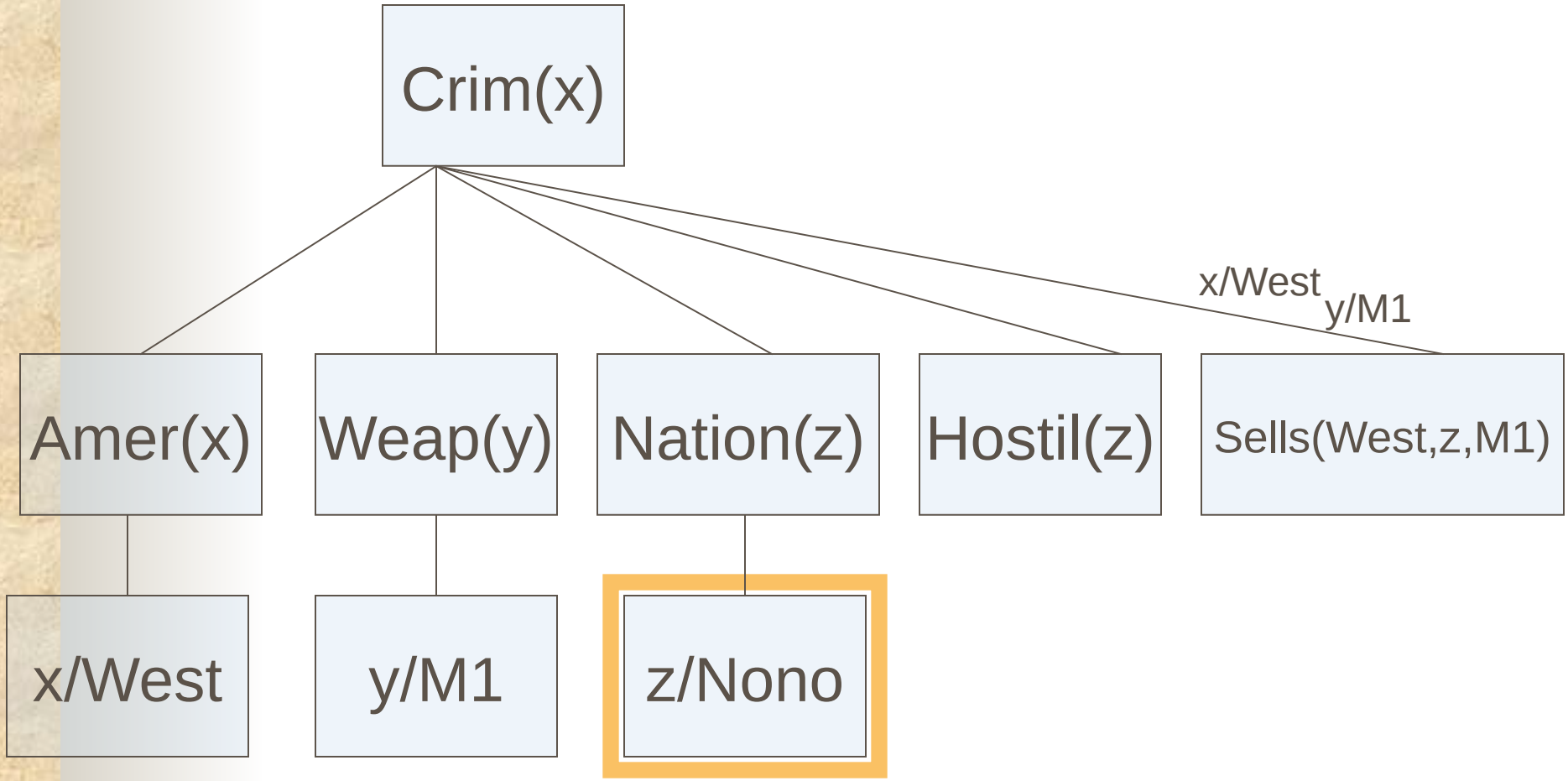

Crim(x)
x/West
y/M1
Amer(x)
Weap(y)
Nation(z)
Hostil(z)
Vende(West,z,M1)
x/West
y/M1

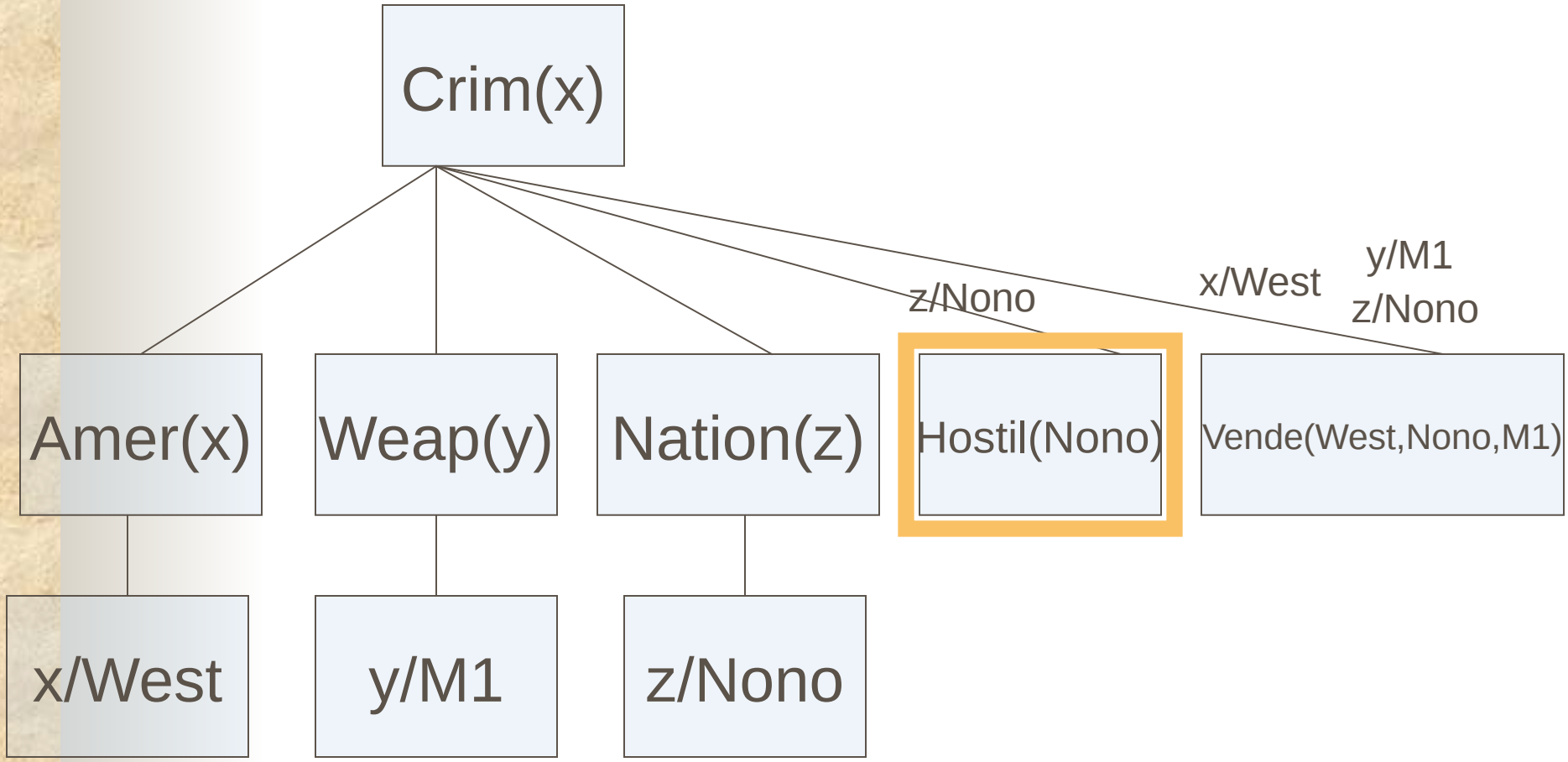

Crim(x)
x/West
y/M1
Amer(x)
Weap(y)
Nation(z)
Hostil(z)
Sells(West,z,M1)
x/West
y/M1
z/US

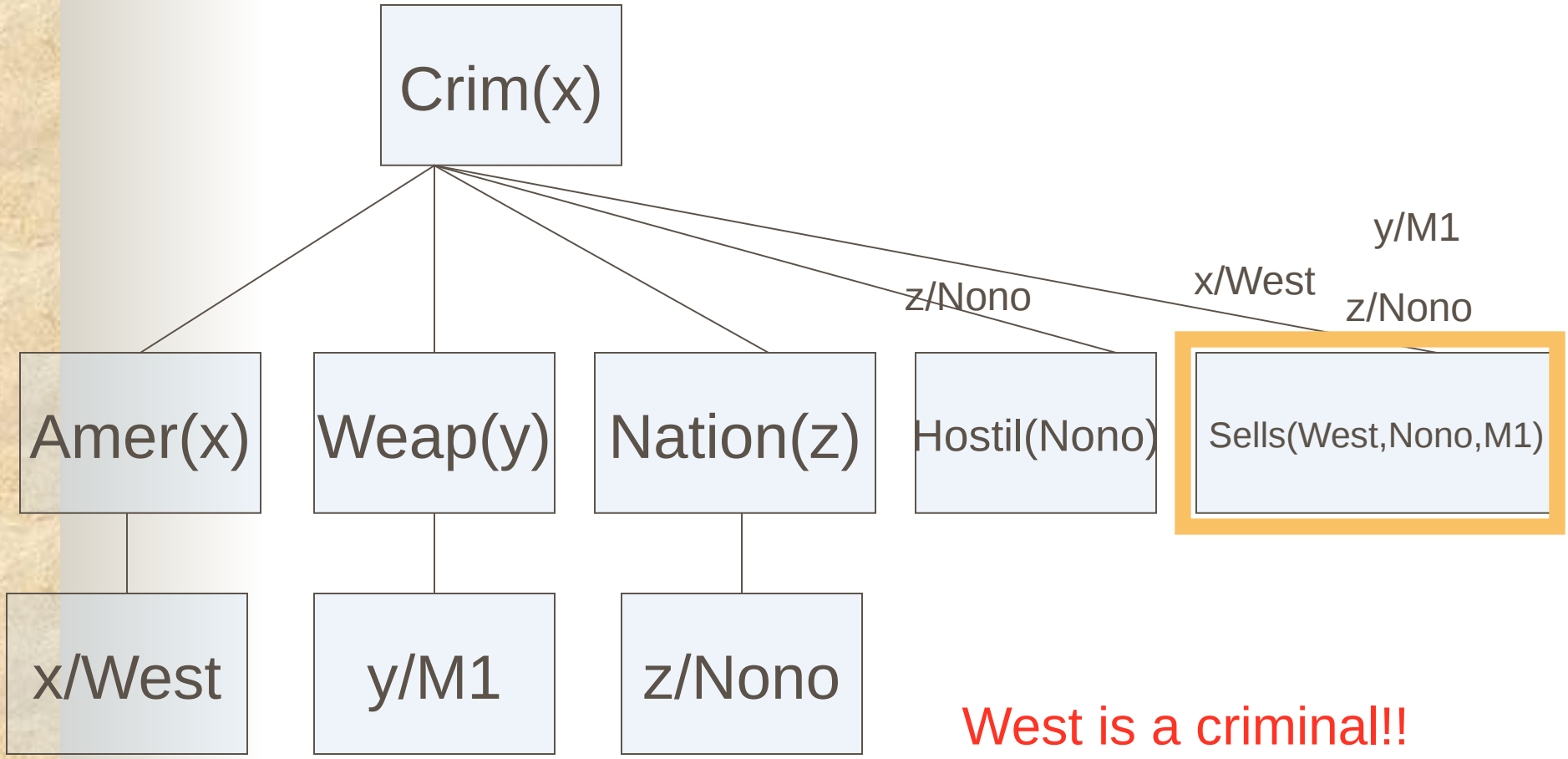

Crim(x)
y/M1
x/West
z/US
z/US
Amer(x)
Weap(y)
Nation(z)
Hostil(US)
Vende(West,US,M1)
x/West
y/M1
z/US


Crim(x)
Amer(x)
Weap(y)
Nation(z)
Hostil(US)
Vende(West,US,M1)
x/West
y/M1
z/US
z/US
x/West
y/M1
z/US
FAIL!


Crim(x)
x/West
y/M1
Amer(x)
Weap(y)
Nation(z)
Hostil(z)
Sells(West,z,M1)
x/West
y/M1


Crim(x)
x/West
y/M1
Amer(x)
Weap(y)
Nation(z)
Hostil(z)
Sells(West,z,M1)
x/West
y/M1
z/Nono


Crim(x)
z/Nono
x/West
y/M1
z/Nono
Amer(x)
Weap(y)
Nation(z)
Hostil(Nono)
Vende(West,Nono,M1)
x/West
y/M1
z/Nono



West is a criminal!!

Programming in Prolog

- Prolog programs are represented by *Horn clauses*, a subset of *first order logic (fol)*, where each clause has at most one positive literal in the clause head (consequent of the implication is positive).
- Prolog: declarative language.
- Program: set of *facts* and/or *rules* that define relations among objects.

Programming in Prolog

- Facts:

```
valuable(gold).
female(jane).
father(john,mary).
human(socrates).
greek(socrates).
```

- **Attention to the syntax!**

- Rules:

```
likes(john,X) :-
    likes(X,vinho).
bird(X) :-
    animal(X),
    has_feathers(X).
sister(X,Y) :-
    female(X),
    parents(M,F,X),
    parents(M,F,Y).
```

Programming in Prolog

- The execution of a Prolog program is deduction!
- *Facts*: relations always true (axioms).
- *Rules*: Relations that are true or false depending on other relations

Programming in Prolog: Syntax

- Terms:
  - Variables: X, Y, C1, _ABC, Input
  - Constants: prolog, a, 123, 'rio de janeiro', porto
  - Structures (compund terms): dono(john,livro(ulysses,autor(james,joyce)))
- Chars: uppercase and lowercase letters, digits, other keyboard symbols.
- Special symbols: **:- ; , .**
- Comments:
  - line: % This is a comment.
  - block: /* This is also a comment that can spread through several lines*/

Programming in Prolog: Syntax

- Operators: +, -, *, / etc.
- Equality and ``matching'':

  a(b,c,d(e,F,g(h,i,j))) = a(B,C,d(E,f,g(H,i,j)))
- Relational operators: =, \=, <, >, >=, =<
- Comparing strings/terms: ==, \== @<, @>

Programming in Prolog: Syntax

- Note: Prolog **does not evaluate** arithmetic expressions that do not appear explicitly in the body of a clause.
- In order to have the expression evaluated one needs to put the expression in the body and use the special operator **is**.
  - p(2+3,4*5).
  - The two expressions above are not evaluated!!!
  - To force evaluation:

    p(X,Y) :- X is 2+3, Y is 4*5.

Programming in Prolog: Example

```
parent(C,M,F) :-
    mother(C,M),
    father(C,F).
```

```
mother(john,ann).
mother(mary,ann).
father(mary,fred).
father(john,fred).
female(mary).
```

Query:

```
?-female(mary),parent(mary,M,F),parent(john,M,F).
```

Programming in Prolog: Lists

- Lists: special data structure in Prolog.
- E.g.:
  - []: empty list.
  - [the,men, [like,to,fish]]
  - [a,V1,b, [X,Y]]

Programming in Prolog: Lists

- Non-empty list: **[Head|Tail]**
- Head: first element of the list
  (it can be of any type).
- Tail: **list** with the remaining elements
  (type is always a list).

Programming in Prolog: Lists

- Examples:

List	Head	Tail
[a,b,c]	a	[b,c]
[a]	a	[]
[[the,cat],sat]	[the,cat]	[sat]
[the,[cat,sat]]	the	[[cat,sat]]
[X+Y,x+y]	X+Y	[x+y]
[]	no head	no tail