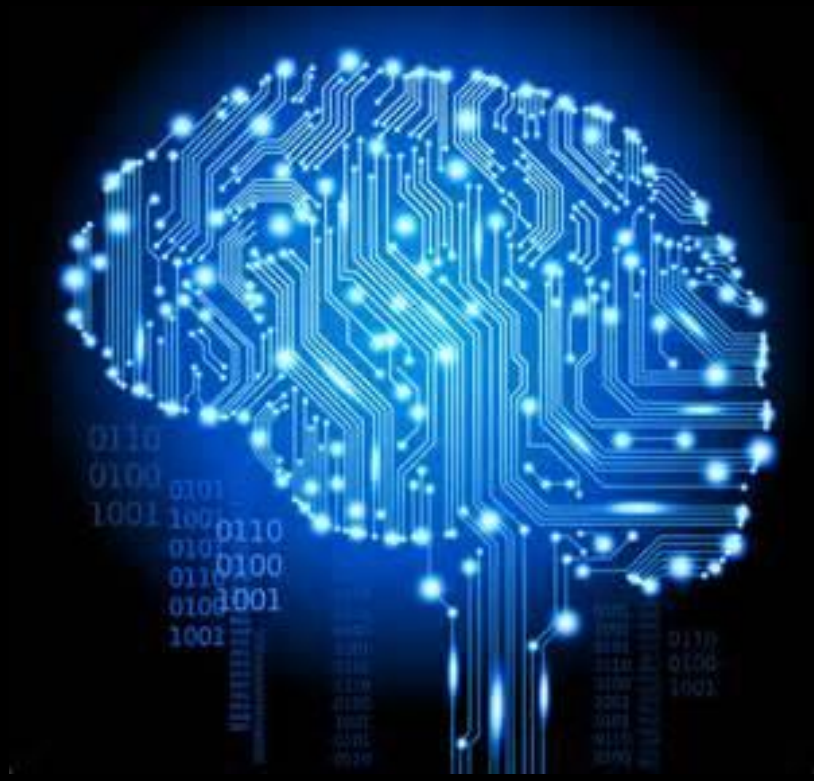
PROGRAMACIÓN LÓGICA



UNIVERSIDAD NACIONAL DE COLOMBIA

Luis Eduardo Martín De La Peña
Anni Alejandra Piragauta Urrea

CONTENIDO

UNIVERSIDAD NACIONAL DE COLOMBIA

1. Introducción
    1. Paradigma de programación
    2. Historia
    3. Filosofía del paradigma.
2. Programación Lógica
    1. ¿Qué es?
    2. Conceptos claves
3. Ventajas y Desventajas
4. Lenguajes de Programación
5. Aplicaciones

UNIVERSIDAD
NACIONAL
DE COLOMBIA

INTRODUCCIÓN

PARADIGMA DE PROGRAMACIÓN

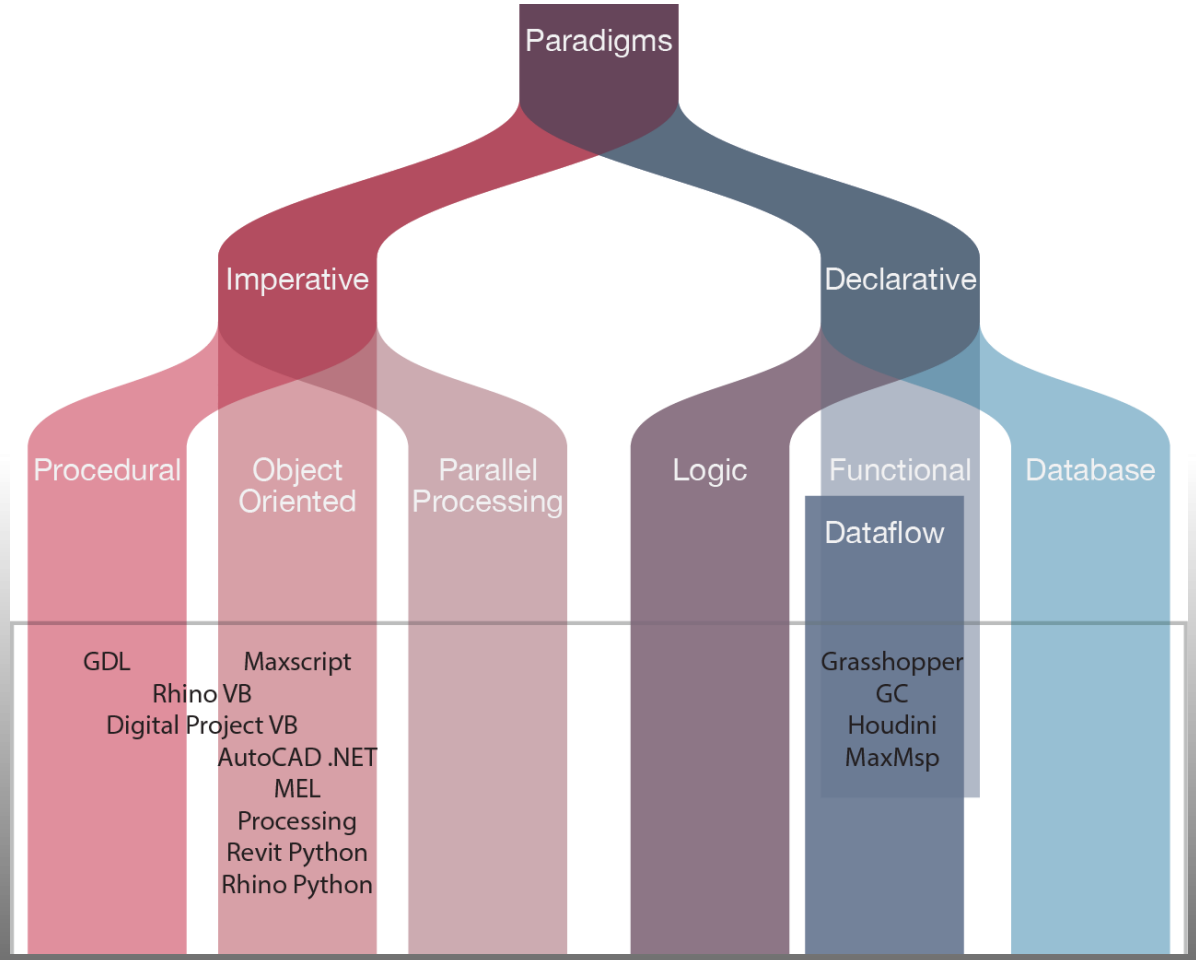
UNIVERSIDAD NACIONAL DE COLOMBIA

Indica un método mediante el cual resolver uno a varios problemas claramente delimitados.

Representa un enfoque particular o filosofía para diseñar soluciones.

"es el conjunto de principios subyacentes que dan forma al estilo de un lenguaje de programación. "

Concepts, Techniques, and Models of Computer Programming.



Imperativo vs. Declarativo

UNIVERSIDAD NACIONAL DE COLOMBIA

Imperativo

¿**Cómo** resolver el problema?

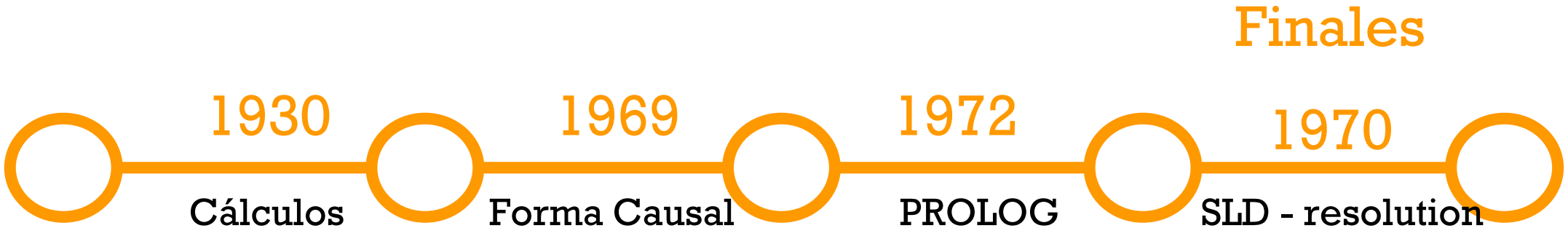
- Programación modular
- Programación estructurada
- Orientada a eventos

Declarativo

¿**Qué** hacer para resolver el problema?

- Funcional
- Lógica
- Programación reactiva
- Lenguajes descriptivos

HISTORIA


1930
Cálculos
Lambda
Alonzo Church
1969
Forma Causal
de la Lógica
Cordell Green
1972
PROLOG
Alain Colmerauer
Philippe Roussel
Finales
1970
SLD - resolution
Robert Kowalski

FILOSOFÍA DEL PARADIGMA

UNIVERSIDAD NACIONAL DE COLOMBIA

Aplicación de reglas de la lógica para inferir conclusiones a partir de datos.

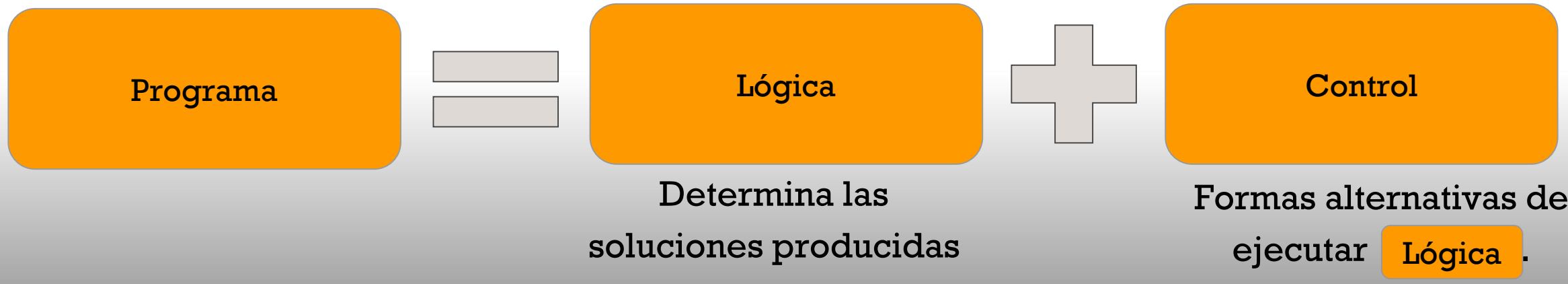
UNIVERSIDAD
NACIONAL
DE COLOMBIA

PROGRAMACIÓN LÓGICA

¿Qué es?

Paradigma de programación basado en la lógica de primer orden.

Se puede ver como una deducción controlada.



Lógica Proposicional

UNIVERSIDAD NACIONAL DE COLOMBIA

- También llamada ***lógica de enunciados***: toma como elemento básico las frases declarativas simples o **proposiciones**.

- **Proposiciones:** Elementos de una frase que constituyen por sí solos una unidad de comunicación de conocimientos y pueden ser considerados verdaderos o falsos.

  - Simple: "Pepito es humano".
  - Compuesta: "Pepito es hombre y pepita es mujer"

Lógica Proposicional

UNIVERSIDAD NACIONAL DE COLOMBIA

Sentencia ⟶ *Sentencia Atómica* | *Sentencia Compleja*
Sentencia Atómica ⟶ **Verdadero** | **Falso** | *Símbolo Proposicional*
Símbolo Proposicional ⟶ **P** | **Q** | **R** | **...**
Sentencia Compleja ⟶ ¬*Sentencia*
| (*Sentencia* ^ *Sentencia*)
| (*Sentencia* v *Sentencia*)
| (*Sentencia* ⇒ *Sentencia*)
| (*Sentencia* ⇔ *Sentencia*)

P	*Q*	¬*P*	*P* ^ *Q*	*P* v *Q*	*P* ⇒ *Q*	*P* ⇔ *Q*
falso	*falso*	*verdadero*	*falso*	*falso*	*verdadero*	*verdadero*
falso	*verdadero*	*verdadero*	*falso*	*verdadero*	*verdadero*	*falso*
verdadero	*falso*	*falso*	*falso*	*verdadero*	*falso*	*falso*
verdadero	*verdadero*	*falso*	*verdadero*	*verdadero*	*verdadero*	*verdadero*

Lógica de Primer orden

UNIVERSIDAD NACIONAL DE COLOMBIA

- También llamada ***lógica de predicados***: es un sistema deductivo basado en un Lenguaje Lógico Matemático formal.

- Incluye proposiciones lógicas, predicados y cuantificadores.

- Más expresiva de la **Lógica proposicional.**
  - ¿**Qué** se afirma? (*predicado o relación*)
  - ¿De **quién** se afirma? (*objeto*)

Lógica de Primer orden

UNIVERSIDAD NACIONAL DE COLOMBIA

Sentencia	⟶	*Sentencia Atómica* | *(Sentencia Conectiva Sentencia)* | *Cuantificador Variable … Sentencia* | ¬*Sentencia*
Sentencia Atómica	⟶	*Predicado (Término…)* | *Término = Término*
Término	⟶	*Función(Término)* | *Constante* | *Variable*
Conectiva	⟶	^ |v|⇒| ⇔
Cuantificador	⟶	¬*Sentencia*
Variable	⟶	*a* | *x* | *s* | …
Predicado	⟶	*TieneColor* | *EstáLloviendo* | …
Función	⟶	*Hombre* | *Humano* | *Mujer* | …

Cláusulas de Horn

- Alfred Horn (1951)

UNIVERSIDAD NACIONAL DE COLOMBIA

Secuencia de literales que contiene a lo sumo uno de sus literales positivos (disyunción de literales).

$$\neg p \vee \neg q \vee \cdots \vee \neg t \vee u$$

$$(p \wedge q \wedge \cdots \wedge t) \rightarrow u$$

Cláusula 'definite': Cláusula de Horn con exactamente un literal positivo.

Hecho: Cláusula 'definite' sin literales negativos.

Cláusula objetivo: Sin ningún literal positivo. (consulta)

antecedente -> consecuente

Se escribe primero el consecuente luego el antecedente.

Cláusulas de Horn - Ejemplo

UNIVERSIDAD NACIONAL DE COLOMBIA

$$\neg mujer(A) \vee \neg padre(B, A) \vee hija(A, B)$$

$$(mujer(A) \wedge padre(B, A)) \rightarrow hija(A, B)$$

"A es hija de B si A es mujer y B es padre de A"

```
hija(A,B) :- mujer(A), padre(B,A)
```

Resolución SLD - Robert Kowalski (Finales 70's)

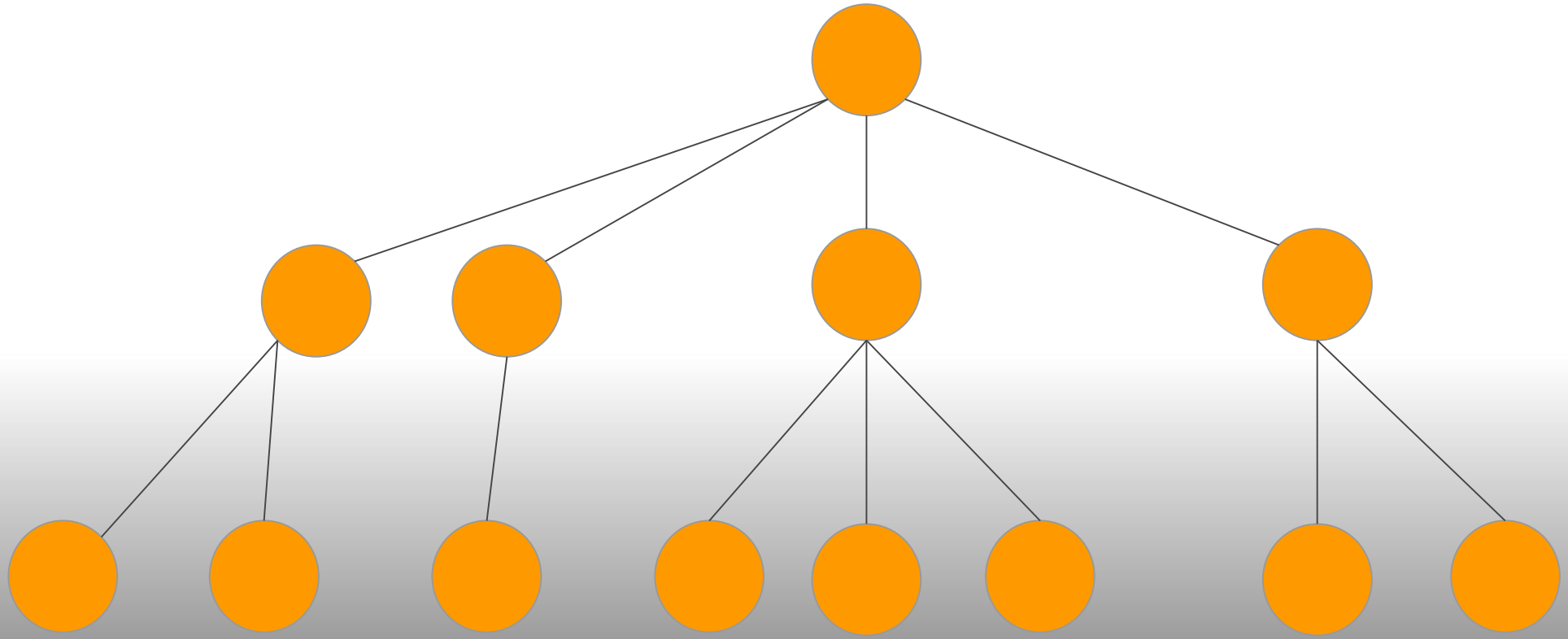
UNIVERSIDAD NACIONAL DE COLOMBIA

SLD (*Selective Linear Definite* clause resolution) Es un método de prueba por refutación que emplea el algoritmo de unificación como mecanismo de base y permite la extracción de respuestas.

El **antecedente** puede ser una conjunción de condiciones que se denomina **secuencia de objetivos**. Todos los objetivos terminan su ejecución en éxito ("verdadero"), o en fracaso ("falso").

Unificación: Cada objetivo determina un subconjunto de cláusulas susceptibles de ser ejecutadas. Cada una de ellas se denomina **punto de elección.**

Backtracking



CONCEPTOS CLAVES

UNIVERSIDAD NACIONAL DE COLOMBIA

Hecho: Declaración, cláusula o **proposición cierta o falsa**, el hecho establece una relación entre objetos y es la forma más sencilla de sentencia

"Pepito es Humano"

```
humano(pepito).
```

CONCEPTOS CLAVES

UNIVERSIDAD NACIONAL DE COLOMBIA

Regla: Implicación o inferencia lógica que **deduce nuevo conocimiento**, la regla permite definir nuevas relaciones a partir de otras ya existentes

"x es mortal si x es humano"

```
mortal(X) :- humano(X).
```

CONCEPTOS CLAVES

UNIVERSIDAD NACIONAL DE COLOMBIA

Consulta: Se especifica el problema, la proposición a demostrar o el objetivo.

```
humano(pepito).
```
Hecho

```
mortal(X) :- humano(X).
```
Regla

```
55 ?- mortal(pepito).
true.
```

UNIVERSIDAD NACIONAL DE COLOMBIA

VENTAJAS Y DESVENTAJAS

VENTAJAS

UNIVERSIDAD NACIONAL DE COLOMBIA

- Puede mejorarse la eficiencia modificando el componente de control sin tener que modificar la lógica del algoritmo.
- Relaciones multipropósito.
- Simplicidad.
- Generación rápida de prototipos e ideas complejas.
- Sencillez en la implementación de estructuras complejas.
- Potencia.

DESVENTAJAS

UNIVERSIDAD NACIONAL DE COLOMBIA

- Altamente ineficiente.
- Pocas áreas de aplicación
- No existen herramientas de depuración efectivas.
- En problemas reales, es poco utilizado.
- Si el programa no contiene suficiente información para contestar una consulta responde *false.*

UNIVERSIDAD NACIONAL DE COLOMBIA

LENGUAJES DE PROGRAMACIÓN

Programación Lógica

PROLOG

UNIVERSIDAD NACIONAL DE COLOMBIA

- Alain Colmerauer y Philippe Roussel (Finales de los 70's)
- Proviene del francés **PRO**grammation en **LOG**ique.
- Producción interpretada
- Se basa en Lógica de primer orden
- Es declarativo
- Backtracking

Mercury

UNIVERSIDAD NACIONAL DE COLOMBIA

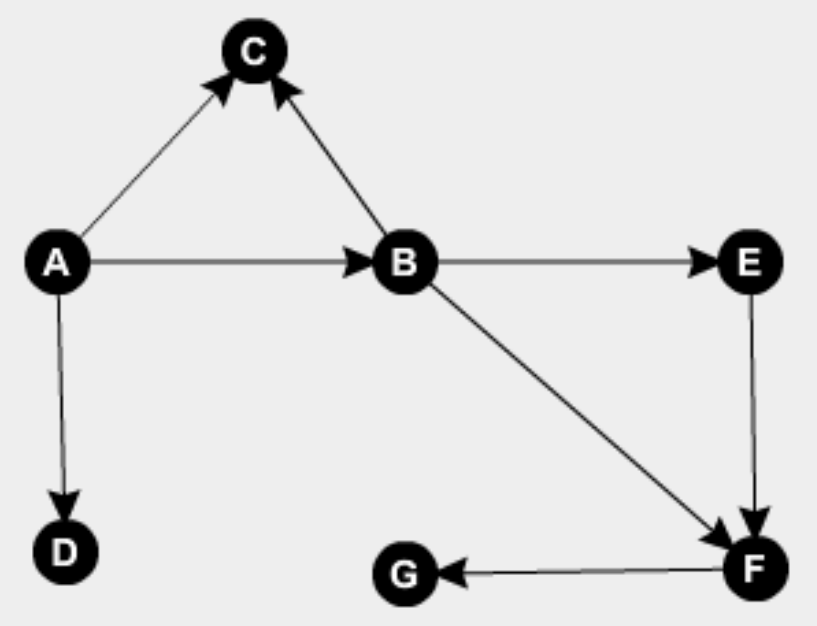
- Fergus Henderson, Thomas Conway y Zoltan Somogyi (1995)
- Sintaxis parecida a PROLOG
- Diseñado para resolver "aplicaciones del mundo real" de forma robusta.
- Soporta polimorfismo
- Un programa escrito en Mercury es más rápido que uno equivalente realizado en Prolog.

OTROS LENGUAJES

UNIVERSIDAD NACIONAL DE COLOMBIA

- CLAC(Logical Composition with the Assistance of Computers)
- Gödel
- Curry
- Ace
- PALs
- Actor Prolog
- CLP (FD)
- CSP (Constraint Satisfaction Problem)
- Lambda Prolog
- Logtalk
- Alma-0

EJEMPLOS - PROLOG




```
arc(a,b).
arc(a,c).
arc(b,e).
arc(b,f).
arc(b,c).
arc(a,d).
arc(e,f).
arc(f,g).

path(X,Y):- arc(X,Y).
path(X,Y):- arc(X,Z),path(Z,Y).

pathall(X,X,[]).
pathall(X,Y,[X,Z|L]):- arc(X,Z),pathall(Z,Y,L).
```

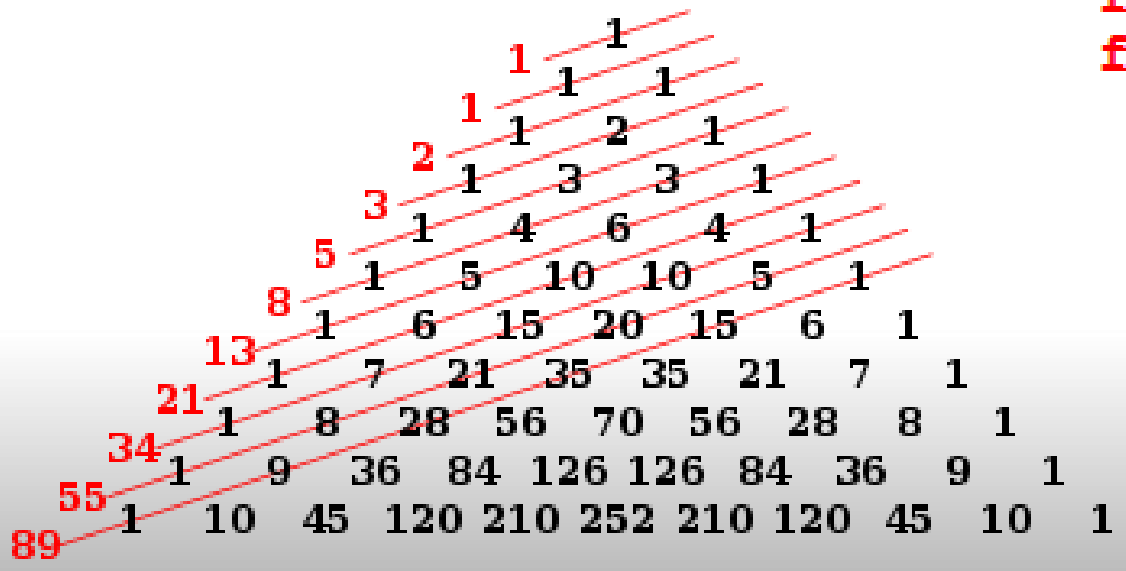
```
4 ?- path(a,b).
true .

5 ?- pathall(a,g,R).
R = [a, b, b, e, e, f, f, g] ;
R = [a, b, b, f, f, g] ;
false.
```

```
6 ?- path(a,e).
true .

7 ?- path(c,g).
false.
```

EJEMPLOS - PROLOG

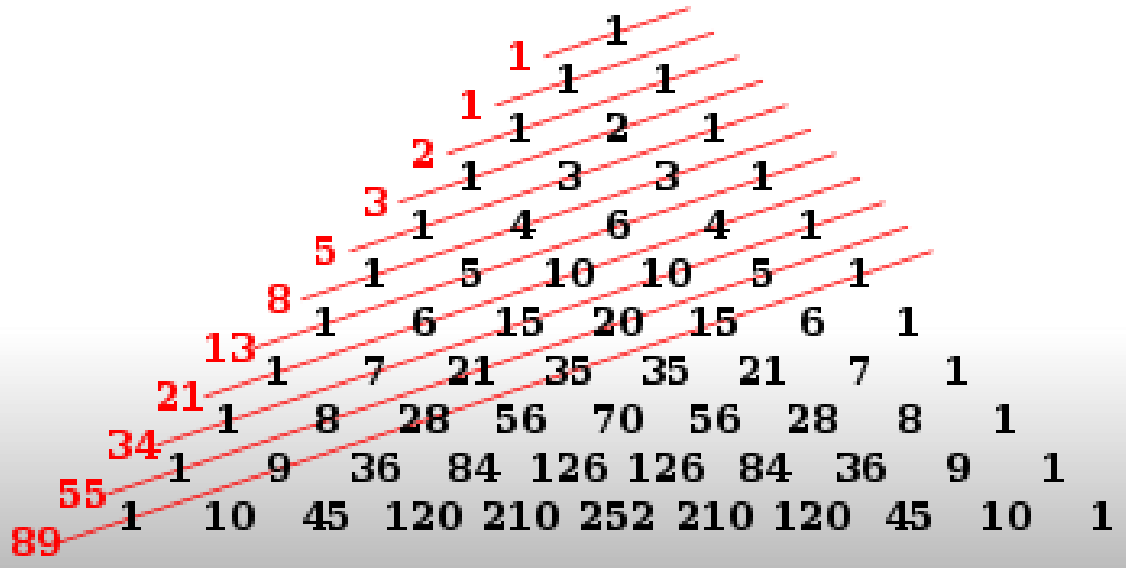



```
fib(0,0).
fib(1,1).
fib(X,Y):- X1 is X-1, X2 is X1-1,
        fib(X1,Y1),
        fib(X2,Y2),
        Y is Y1+Y2.
```

```
4 ?- fib(6,X).
X = 13 .

5 ?- fib(21,X).
X = 10946 .
```

EJEMPLOS - Mercury




```
:- module fib.
:- interface.
:- import_module io.

:- pred main(io::di, io::uo) is det.

:- implementation.
:- import_module int.

:- pred fib(int::in, int::out) is det.

fib(X,Y):- (if X =< 2
              then Y = 1
              else fib(X - 1, Y1), fib(X - 2, Y2), Y = Y1
+ Y2
             ).

main(!IO) :-
          fib(17,X),
          io.write_string("fib(17, ", !IO),
          io.write_int(X, IO),
          io.write_string(")\n, !IO).
```

EJEMPLOS - PROLOG

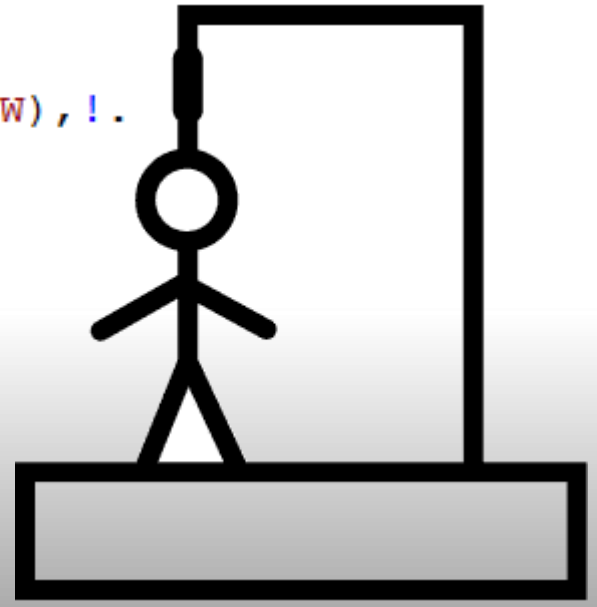
UNIVERSIDAD NACIONAL DE COLOMBIA

```
in_mind([l,o,v,e]).

start:- write('Guess first letter'),read(X),
        in_mind([X|T]),write('OK. '),guess(T).

guess([]):- write('Congratulations! The word is '),in_mind(W),write(W),!.
guess(L):-  repeat,write('Next letter'),read(X),
                ((L=[X|T1],write('OK. '),guess(T1));
                (write('Fail. Try again!'),guess(L))).
```

```
5 ?- start.
Guess first letter
|: l.
OK. Next letter|: a.
Fail. Try again!Next letter|: o.
OK. Next letter|: v.
OK. Next letter|: e.
OK. Congratulations! The word is [l,o,v,e]
true .
```



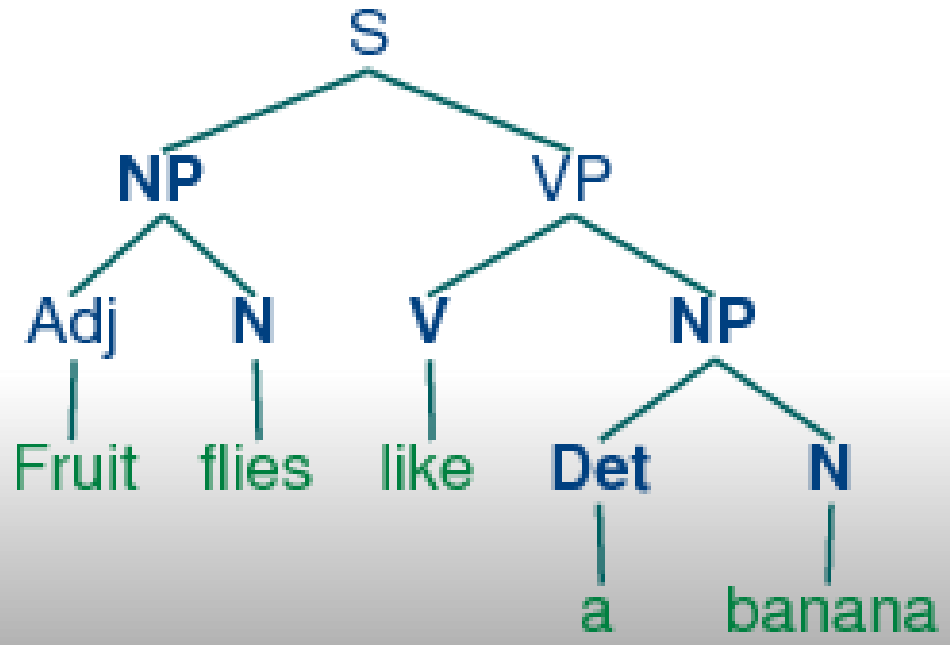
UNIVERSIDAD NACIONAL DE COLOMBIA

APLICACIONES DEL PARADIGMA

APLICACIONES

- Desarrollo de aplicaciones de inteligencia artificial.
- Prueba de teoremas
- Construcción de Sistemas expertos
- Procesamiento del lenguaje natural
- Consultas lógicas basadas en reglas
  - Búsquedas en bases de datos
  - Sistemas de control de voz

Análisis de lenguaje natural




```
oracion(O) :- sintagma_nominal(SN),
              sintagma_verbal(SV),
              append(SN,SV,O).
sintagma_nominal(SN) :- nombre(SN).
sintagma_nominal(SN) :- articulo(A),
                        nombre(N),
                        append(A,N,SN).
sintagma_verbal(SV) :- verbo(V),
                       sintagma_nominal(SN),
                       append(V,SN,SV).

articulo([el]).
nombre([gato]).
nombre([perro]).
nombre([pescado]).
nombre([carne]).
verbo([come]).
```

```
7 ?- oracion([perro,come,pescado]).
true .
```

REFERENCIAS

1. Programación lógica
   - http://programacion-programacionlogica.blogspot.com.co/
   - https://en.wikipedia.org/wiki/Logic_programming
   - https://www.youtube.com/watch?v=SykxWpFwMGs
2. Lógica de primer orden
   - https://drive.google.com/file/d/0B7IRdmOoUVf5RFRBN2RJdDNlNm8/view
   - https://drive.google.com/drive/u/1/folders/0B7IRdmOoUVf5V2pxal81X3NQc2M
3. PROLOG
   - http://www.anselm.edu/homepage/mmalita/culpro/index.html
   - https://es.wikipedia.org/wiki/Prolog
4. Mercury
   - https://mercurylang.org/

UNIVERSIDAD
NACIONAL
DE COLOMBIA

GRACIAS