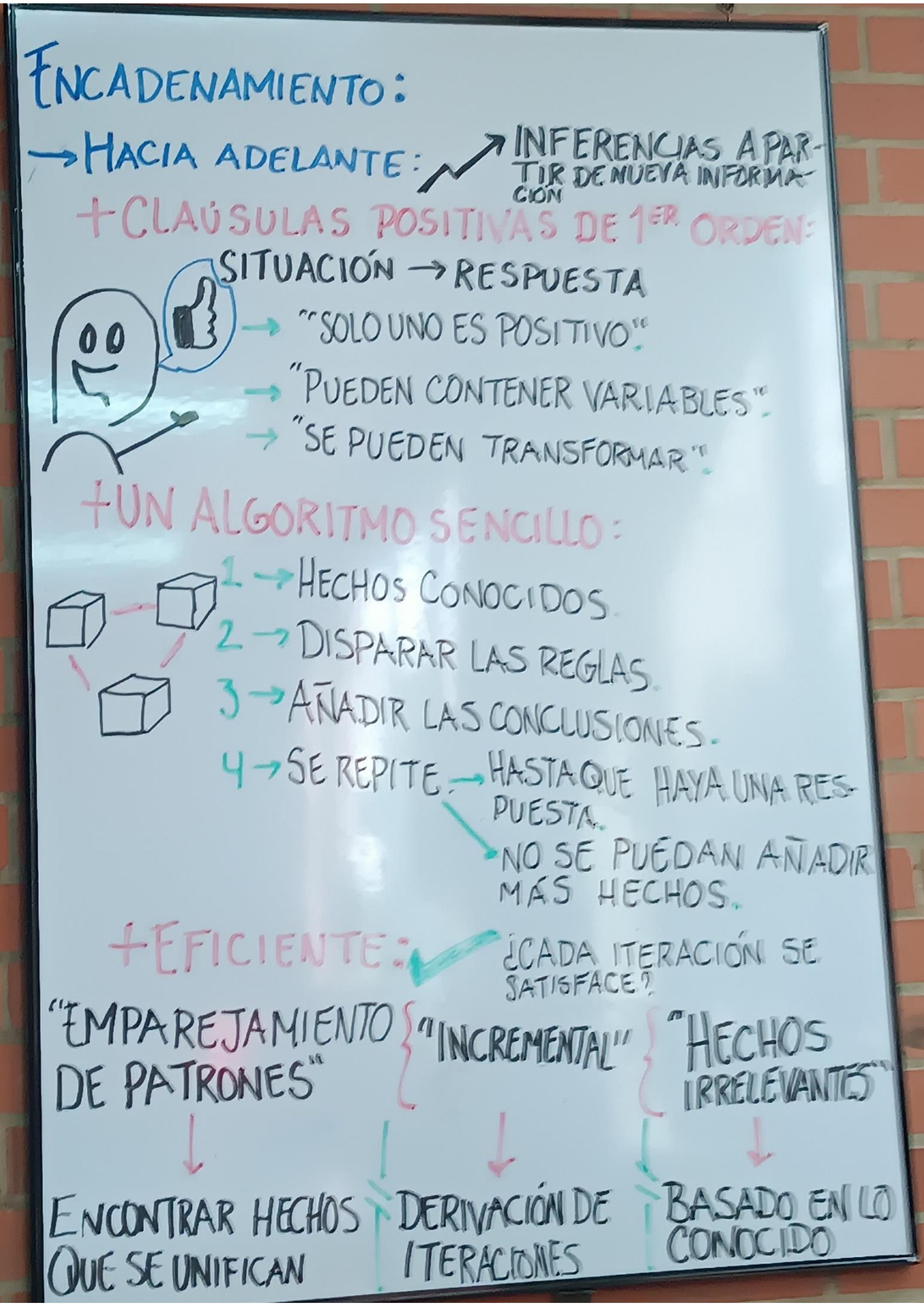


ENCADENAMIENTO:

→ HACIA ADELANTE: ↗ INFERENCIAS A PARTIR DE NUEVA INFORMACIÓN

+ CLAÚSULAS POSITIVAS DE 1ER ORDEN:

SITUACIÓN → RESPUESTA

→ "SOLO UNO ES POSITIVO".

→ "PUEDEN CONTENER VARIABLES".

→ "SE PUEDEN TRANSFORMAR".

+ UN ALGORITMO SENCILLO:

1 → HECHOS CONOCIDOS.

2 → DISPARAR LAS REGLAS.

3 → AÑADIR LAS CONCLUSIONES.

4 → SE REPITE. → HASTA QUE HAYA UNA RESPUESTA.

→ NO SE PUEDAN AÑADIR MÁS HECHOS.

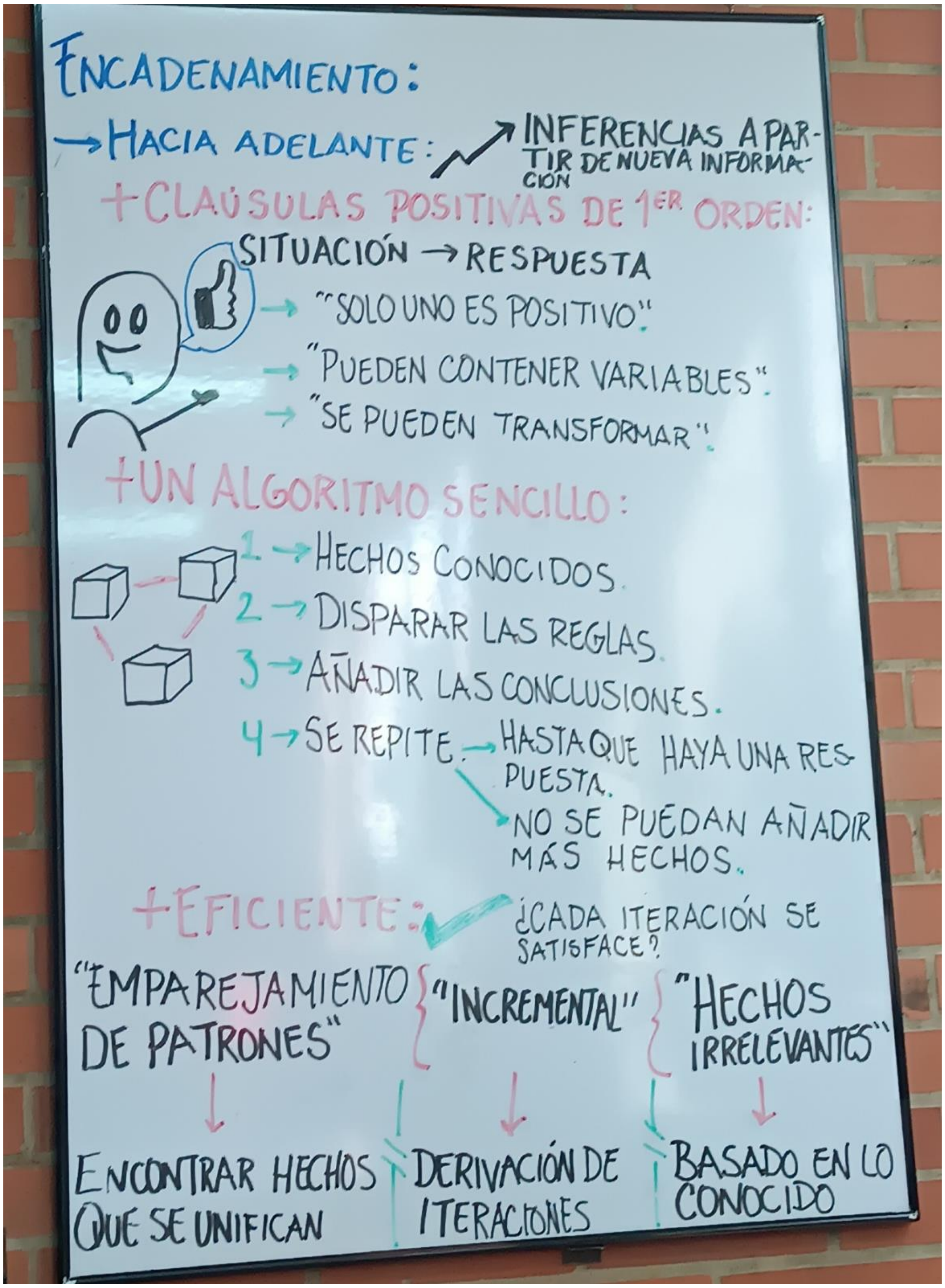
+ EFICIENTE: ✓ ¿CADA ITERACIÓN SE SATISFACE?

"EMPAREJAMIENTO DE PATRONES" { "INCREMENTAL" { "HECHOS IRRELEVANTES"

↓ ENCONTRAR HECHOS QUE SE UNIFICAN

↓ DERIVACIÓN DE ITERACIONES

↓ BASADO EN LO CONOCIDO



ENCADENAMIENTO:

→ HACIA ADELANTE: ↗ INFERENCIAS A PARTIR DE NUEVA INFORMACIÓN

+ CLAÚSULAS POSITIVAS DE 1ER ORDEN:

SITUACIÓN → RESPUESTA

→ "SOLO UNO ES POSITIVO".

→ "PUEDEN CONTENER VARIABLES".

→ "SE PUEDEN TRANSFORMAR".

+ UN ALGORITMO SENCILLO:

1 → HECHOS CONOCIDOS.

2 → DISPARAR LAS REGLAS.

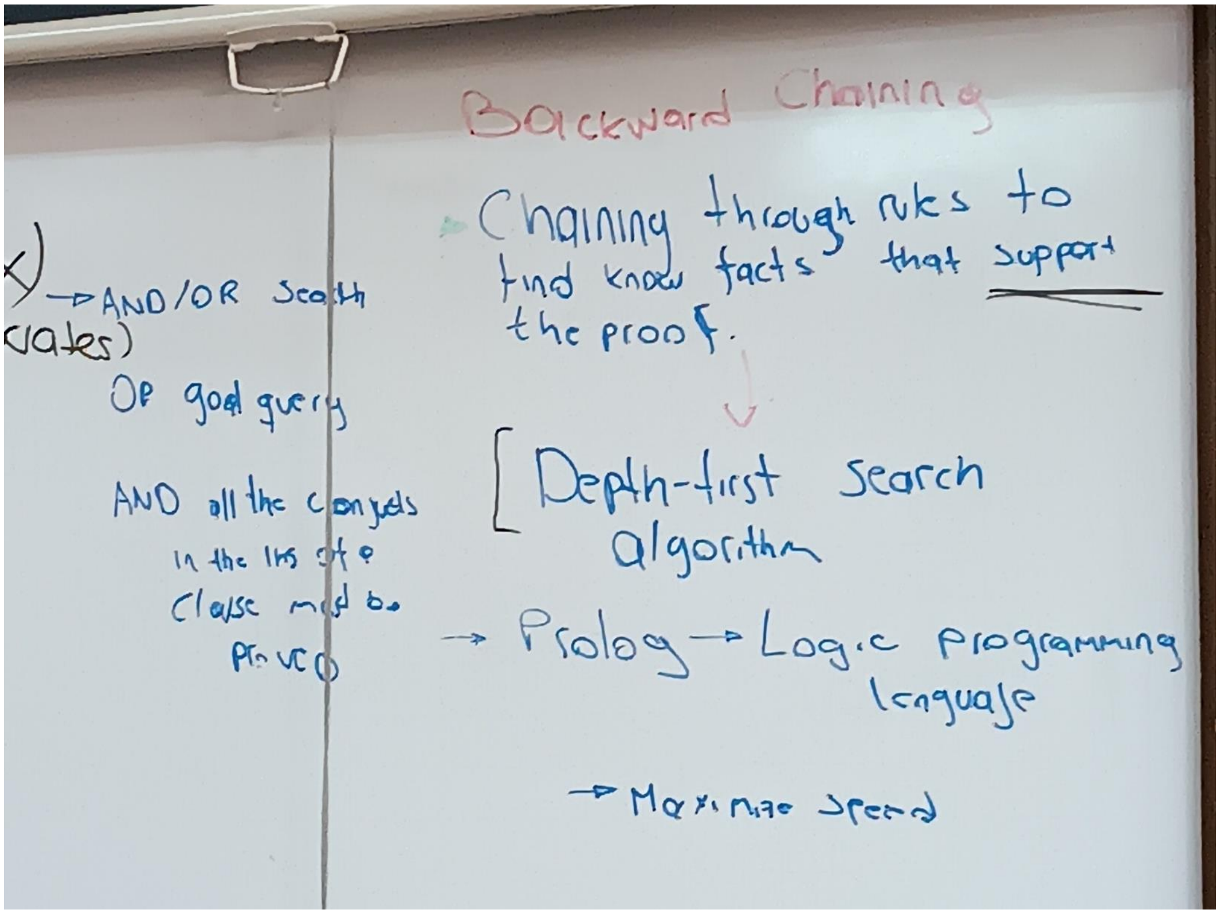
3 → AÑADIR LAS CONCLUSIONES.

4 → SE REPITE. → HASTA QUE HAYA UNA RESPUESTA.

→ NO SE PUEDAN AÑADIR MÁS HECHOS.

+ EFICIENTE: ✓ ¿CADA ITERACIÓN SE SATISFACE?

"EMPAREJAMIENTO DE PATRONES"	"INCREMENTAL"	"HECHOS IRRELEVANTES"
↓	↓	↓
ENCONTRAR HECHOS QUE SE UNIFICAN	DERIVACIÓN DE ITERACIONES	BASADO EN LO CONOCIDO



x) → AND/OR Search
vates)

OR goal query

AND all the conjuncts in the lhs of a clause must be proved

Backward Chaining

- Chaining through rules to find know facts that support the proof.

[Depth-first search algorithm

→ Prolog → Logic programming language

→ Maximize speed