Metodología y plan de trabajo (actualización)

Puede sufrir cambios (se notificarán con la antelación suficiente)

Agosto 2023
Lu Ma Mi Ju Vi Sa Do
1 2 3 4 5 6
7 8 9 10 11 12 13
14 15 16 17 18 19 20
21 22 23 24 25 26 27
28 29 30 31

SEPTIEMBRE 2023

Octubre 2023
Lu Ma Mi Ju Vi Sa Do
1
2 3 4 5 6 7 8
9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31

Lunes	Martes	Miércoles	Jueves	Viernes	Sábado	Domingo
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	7calendar.com/es/

Septiembre 2023
Lu Ma Mi Ju Vi Sa Do
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30

OCTUBRE 2023

Noviembre 2023
Lu Ma Mi Ju Vi Sa Do
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30

Lunes	Martes	Miércoles	Jueves	Viernes	Sábado	Domingo
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					7calendar.com/es/

Octubre 2023
Lu Ma Mi Ju Vi Sa Do
1
2 3 4 5 6 7 8
9 10 11 12 13 14 15
16 17 18 19 20 21 22
23 24 25 26 27 28 29
30 31

NOVIEMBRE 2023

Diciembre 2023
Lu Ma Mi Ju Vi Sa Do
1 2 3
4 5 6 7 8 9 10
11 12 13 14 15 16 17
18 19 20 21 22 23 24
25 26 27 28 29 30 31

Lunes	Martes	Miércoles	Jueves	Viernes	Sábado	Domingo
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			7calendar.com/es/

Noviembre 2023
Lu Ma Mi Ju Vi Sa Do
1 2 3 4 5
6 7 8 9 10 11 12
13 14 15 16 17 18 19
20 21 22 23 24 25 26
27 28 29 30

DICIEMBRE 2023

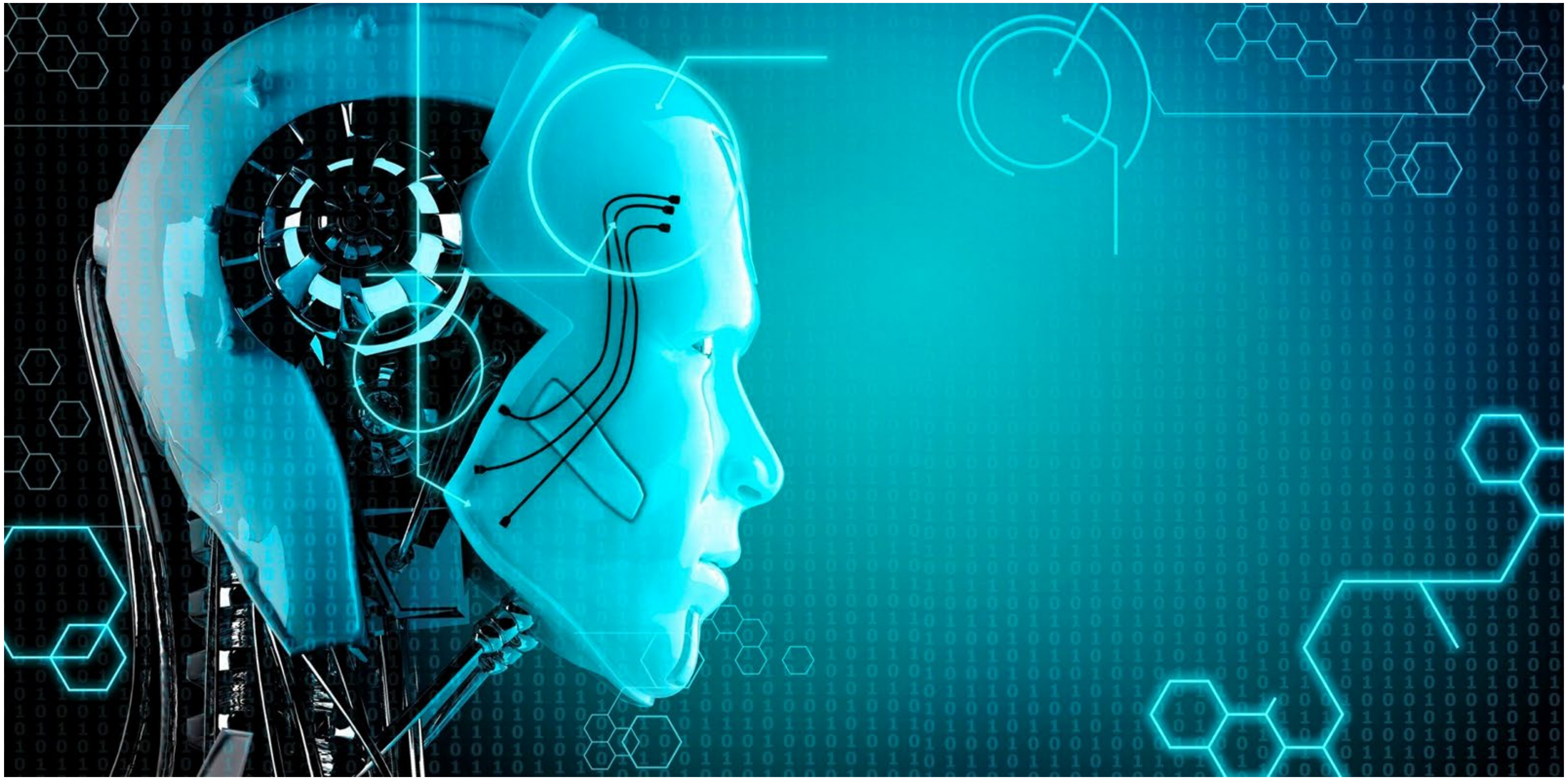
Enero 2024
Lu Ma Mi Ju Vi Sa Do
1 2 3 4 5 6 7
8 9 10 11 12 13 14
15 16 17 18 19 20 21
22 23 24 25 26 27 28
29 30 31

Lunes	Martes	Miércoles	Jueves	Viernes	Sábado	Domingo
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31
						7calendar.com/es/

Examen final: 17 enero

Inteligencia Artificial

Lógica



[Transparencias adaptadas de Dan Klein and Pieter Abbeel: CS188 Intro to AI, UC Berkeley (ai.berkeley.edu)]

Universidad Rey Juan Carlos

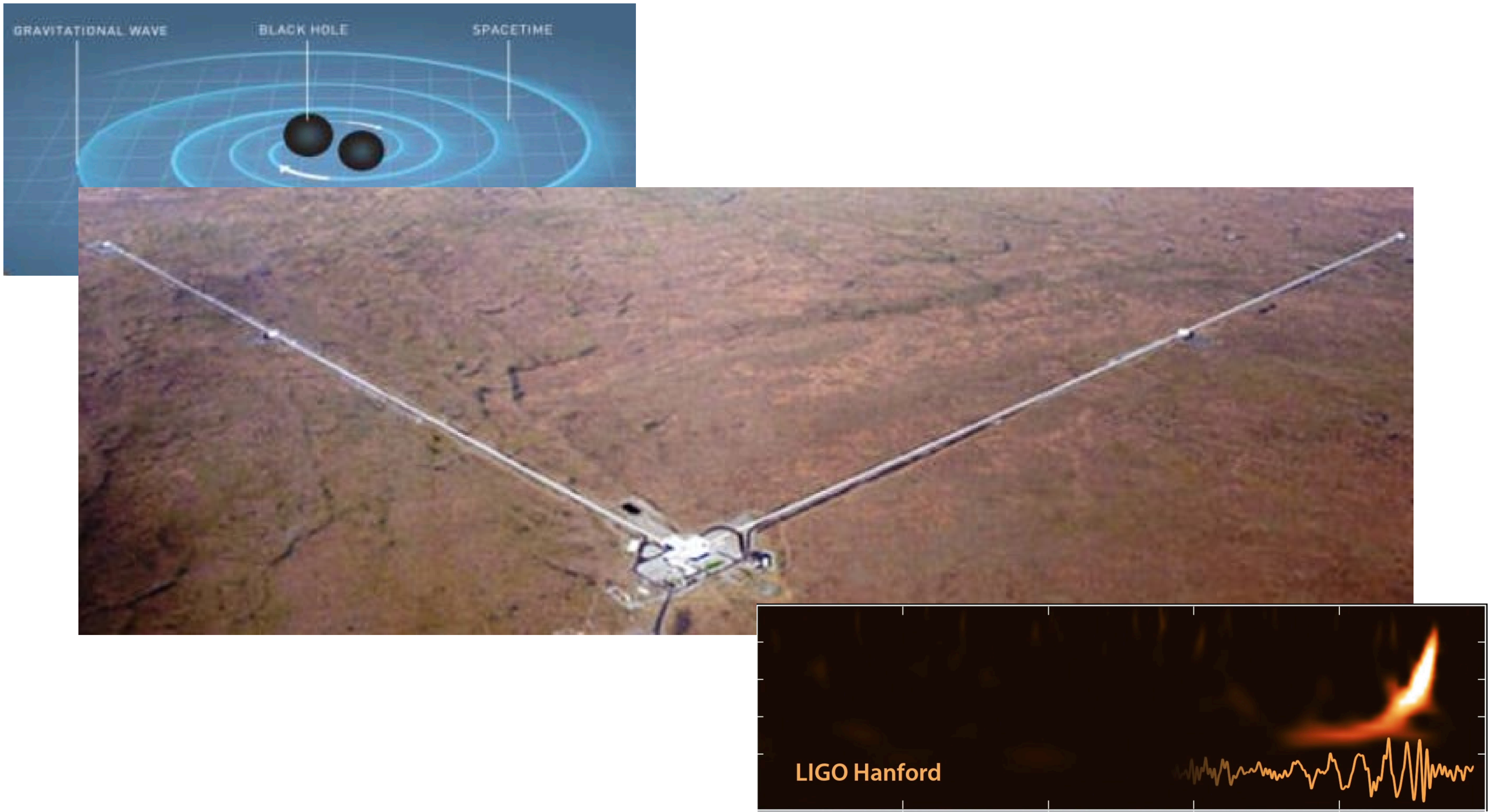
Índice

- Conceptos básicos de conocimiento, lógica, razonamiento
- Lógica proposicional: sintaxis y semántica, ejemplos en los mundos del Wumpus y de Pacman.

Agentes que saben cosas

- Los agentes adquieren **conocimiento** a través de la percepción, el aprendizaje o el lenguaje
  - Conocimiento de los efectos de las acciones ("modelo de transición")
  - Conocimiento de cómo afecta el mundo a los sensores ("modelo de sensor")
  - Conocimiento del estado actual del mundo
- Pueden realizar un seguimiento de un mundo parcialmente observable
- Pueden formular planes para alcanzar metas.
- Pueden diseñar y construir detectores de ondas gravitacionales.....

LIGO



Conocimiento

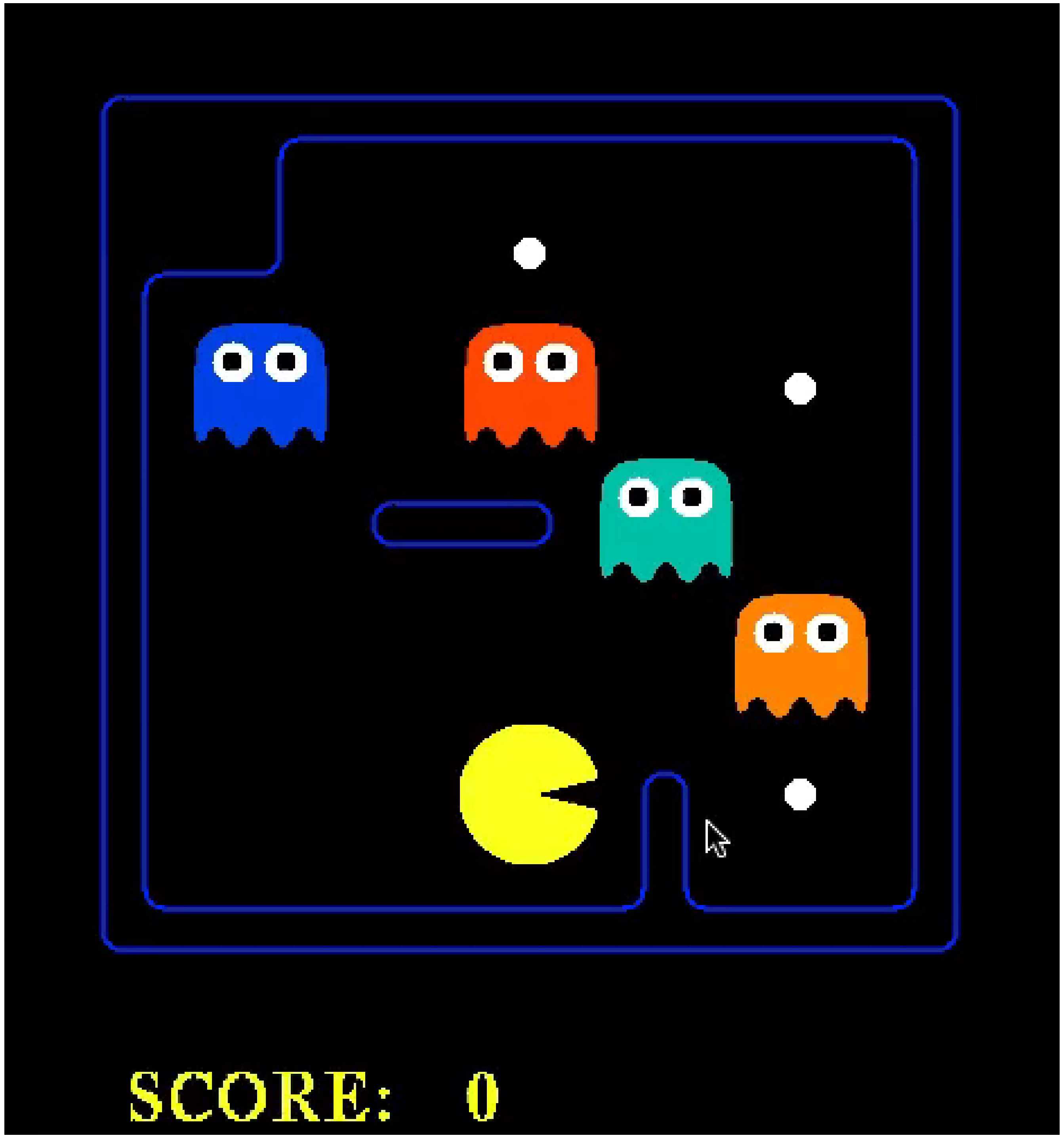
- **Base de conocimiento (KB)** = conjunto de *sentencias* (que representan afirmaciones sobre el mundo) en un lenguaje formal
- Enfoque declarativo para crear un agente (u otro sistema):
  - **Dile** lo que necesita saber (o pídele que **aprenda** el conocimiento)
  - A continuación, puede **preguntarse** qué hacer: las respuestas deben obtenerse de la base de conocimiento
- Los agentes pueden verse a **nivel de conocimiento**
  es decir, lo que **saben**, se implemente como se implemente
- Un único algoritmo de inferencia puede responder a cualquier pregunta que se pueda responder

Base de conocimiento	Hechos específicos del dominio
Motor de inferencia	Código genérico

Agente basado en conocimiento

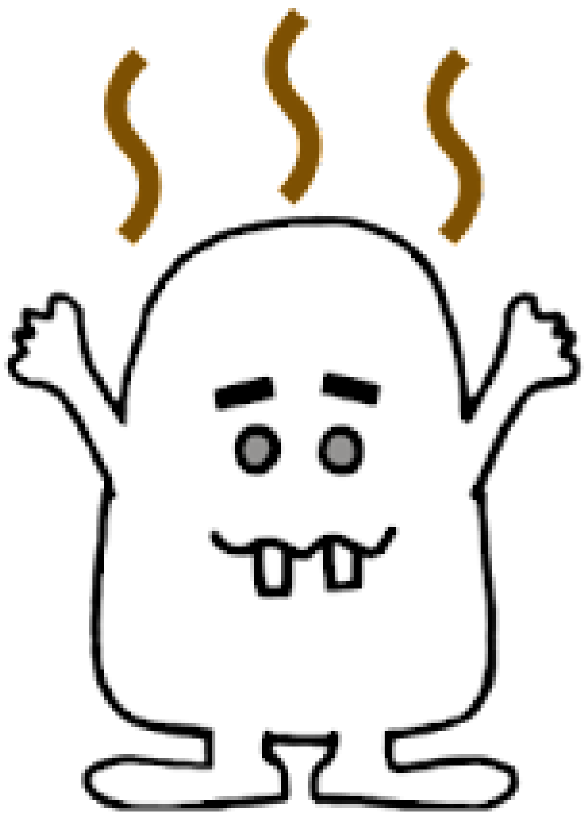
función AGENTE-BC(*percepción*) **devuelve** una *acción*
  variables estáticas: *BC*, una base de conocimiento
    t, un contador, inicializado a 0, que indica el tiempo

  DECIR(*BC*, CONSTRUIR-SENTENCIA-DE-PERCEPCIÓN(*percepción*, *t*))
  acción ← PREGUNTAR(*BC*, PEDIR-Acción(*t*))
  DECIR(*BC*, CONSTRUIR-SENTENCIA-DE-ACCIÓN(*acción*, *t*))
  $t \leftarrow t + 1$
  devolver *acción*


SCORE: 0

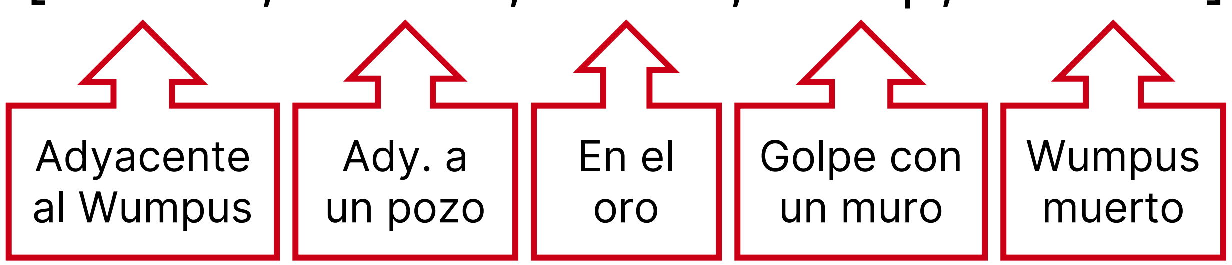
El mundo de Wumpus

- Érase una vez una cueva compuesta por habitaciones conectadas mediante pasillos.
- Escondido en algún lugar de la cueva está el **Wumpus**, una bestia que se come a cualquiera que entre en su habitación.
  - El Wumpus puede ser derribado por la flecha del agente, y éste sólo dispone de una.
- Algunas habitaciones contienen pozos sin fondo que atrapan a aquel que deambula por dichas habitaciones (menos al Wumpus)
- La única motivación para entrar en la cueva es la posibilidad de encontrar una **pila de oro**.



El mundo de Wumpus: PEAS

- Medida de Rendimiento:
  - +1000 por salir de la cueva con el oro
  - -1000 por caer a un pozo o ser comido por Wumpus
  - -1 por cada movimiento, -10 por usar el arco
- Entorno:
  - Un grid de 4x4. [1, 1] es la entrada.
  - Los elementos están aleatoriamente distribuidos
- Actuadores:
  - {Forward, TurnLeft (90º), TurnRight (90º), Grab, Shoot, Climb}
- Sensores:
  - [Stench, Breeze, Glitter, Bump, Scream]

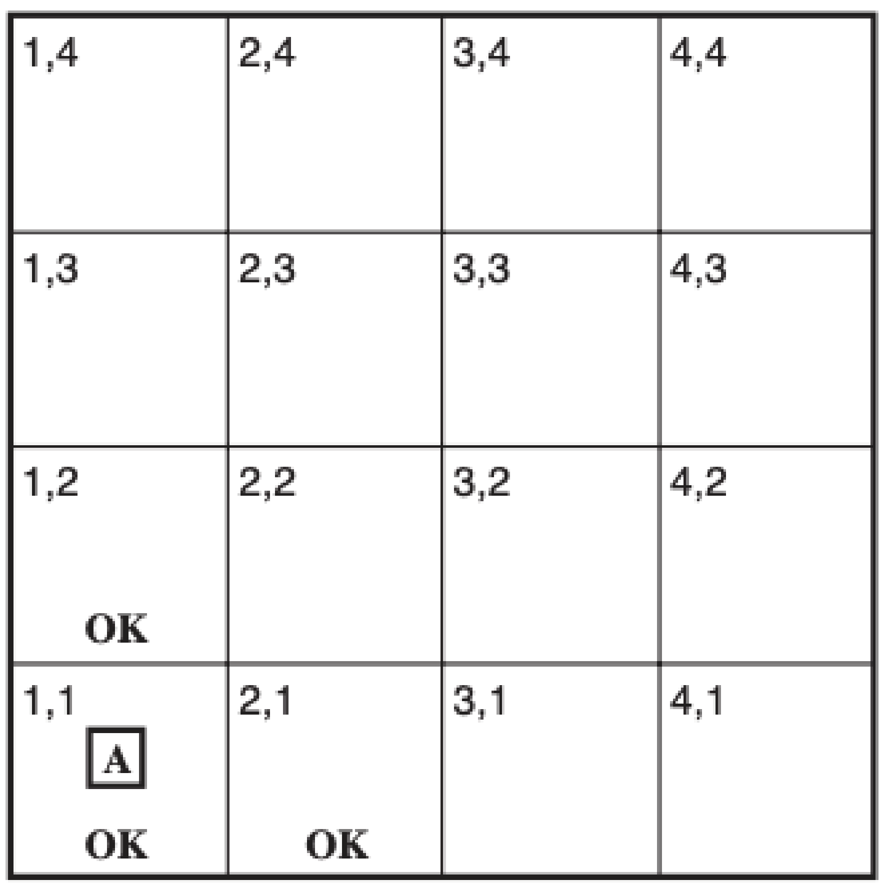


El mundo de Wumpus: PEAS

- ¿Parcial o totalmente observable?
- ¿Agente único o multiagente?
- ¿Determinista o estocástico?
- ¿Estático o dinámico?
- ¿Discreto o continuo?

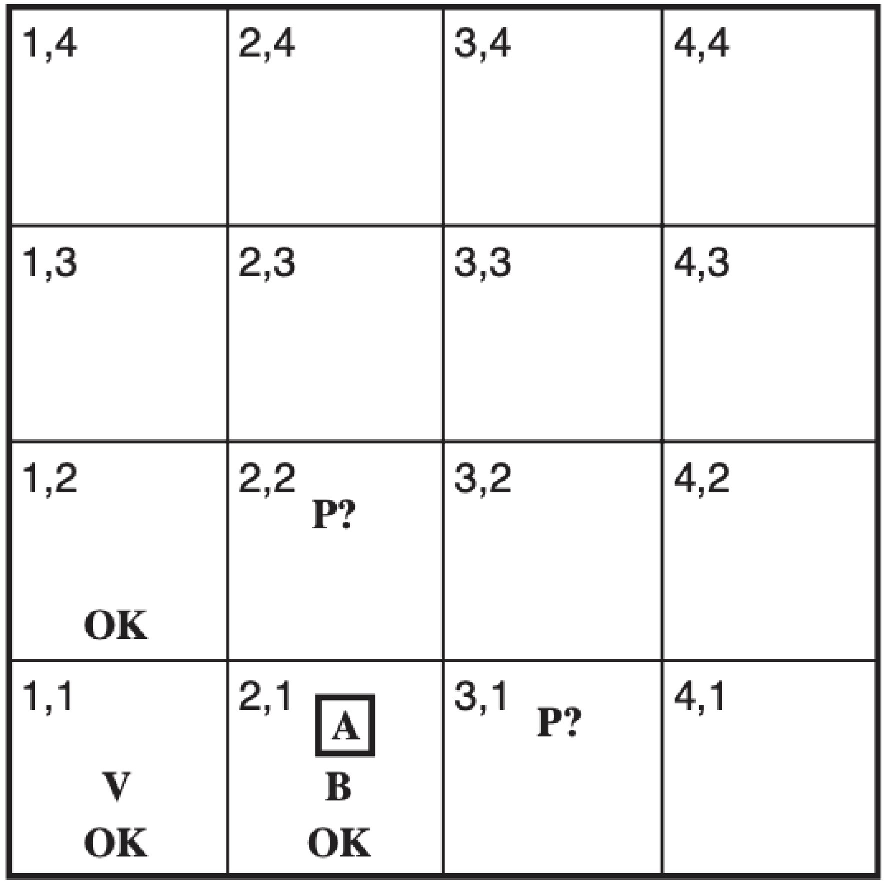
El mundo de Wumpus

A = *Agent*
B = *Breeze*
G = *Glitter, Gold*
OK = *Safe square*
P = *Pit*
S = *Stench*
V = *Visited*
W = *Wumpus*



- **Acción:** (inicial)
- **Percepción:** $[None, None, None, None, None]$

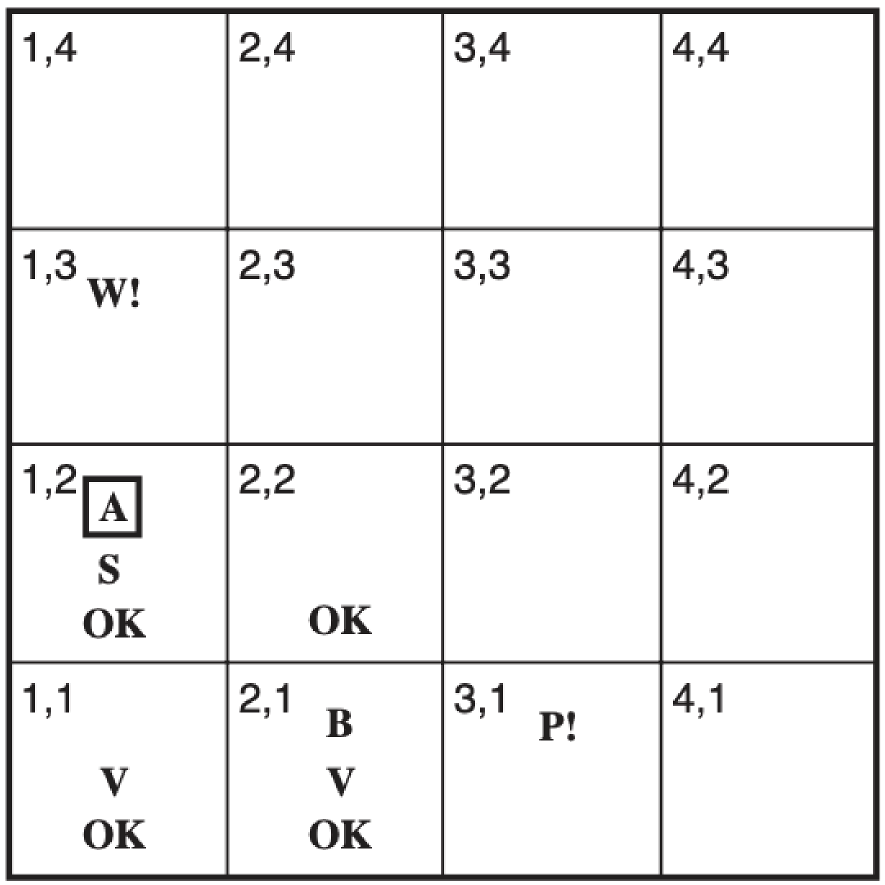
El mundo de Wumpus



- **Acción:** F
- **Percepción:** $[None, Breeze, None, None, None]$

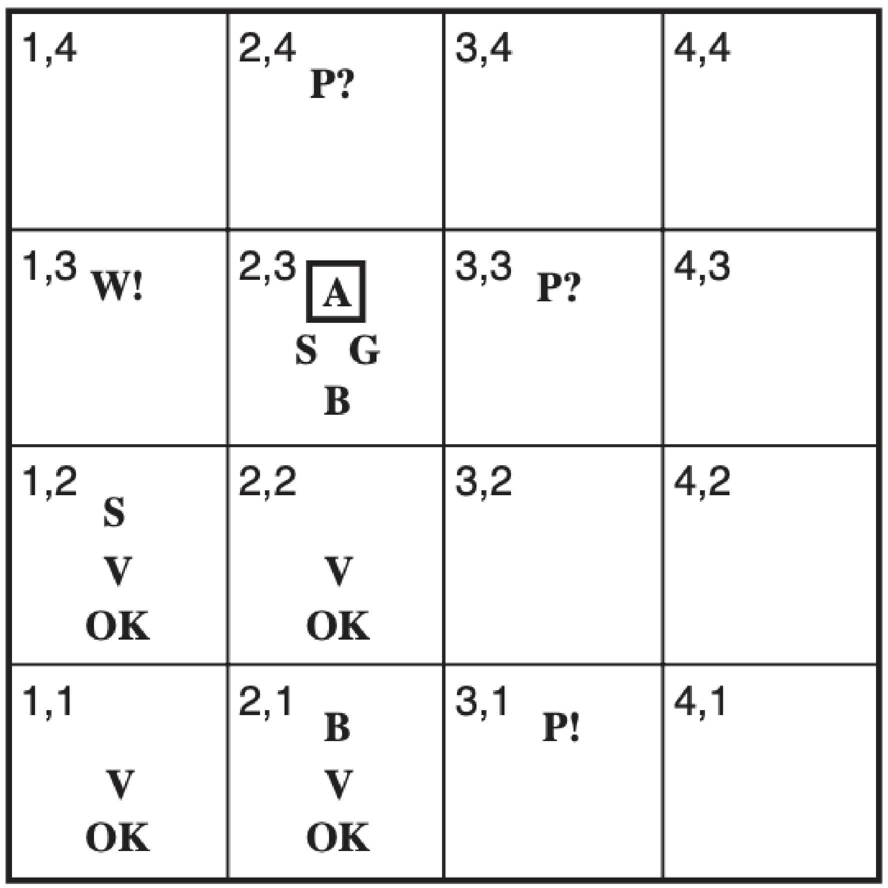
El mundo de Wumpus

$\boxed{\text{A}}$ = *Agent*
B = *Breeze*
G = *Glitter, Gold*
OK = *Safe square*
P = *Pit*
S = *Stench*
V = *Visited*
W = *Wumpus*



- **Acción:** (L, L, F) (R, F)
- **Percepción:** $[Stench, None, None, None, None]$

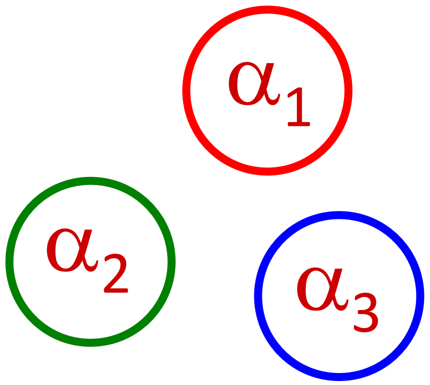
El mundo de Wumpus



- **Acción:** (R, F), (L, F)
- **Percepción**: $[Stench, Breeze, Glitter, None, None]$

Lógica

- **Sintaxis**: ¿Qué sentencias están permitidas?
- **Semántica:**
  - ¿Cuáles son los **mundos posibles**?
  - ¿Qué frases son **verdaderas** en qué mundos (modelos)? (es decir, **definición** de verdad)
    - Se dice que el modelo m “satisface α” o “es un modelo de α”
    - $M(\alpha)$ representa todos los modelos de α



Sintaxisland

Semanticaland

Diferentes tipos de lógica

- Lógica proposicional
  - Sintaxis: $P \vee (\neg Q \wedge R)$; $X_1 \Leftrightarrow (\text{Lluvioso} \Rightarrow \neg \text{Soleado})$
  - Mundo posible: {P=verdadero,Q=verdadero,R=falso,S=verdadero} o 1101
  - Semántica: $\alpha \wedge \beta$ es verdad en un mundo sii (si y sólo si) α es verdad y β es verdad (etc.)
- Lógica de primer orden
  - Sintaxis: $\forall x\ \exists y\ P(x,y) \wedge \neg Q(Joe,f(x)) \Rightarrow f(x)=f(y)$
  - Mundo posible: Objetos o_1, o_2, o_3; P aplica a <o_1,o_2>; Q aplica a <o_3, o_2>; $f(o_1)=o_1$; Juan=o_3; etc.
  - Semántica: $\phi(\sigma)$ es verdad en un mundo si $\sigma=o_j$ y ϕ aplica a o_j; etc.

Diferentes tipos de lógica (continúa)

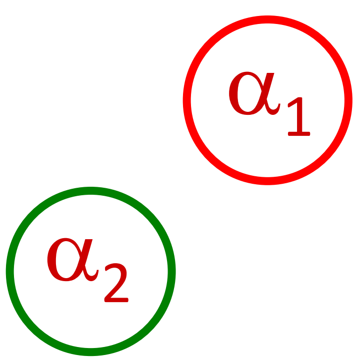
- Bases de datos relacionales:
  - Sintaxis: sentencias relacionales básicas, p. ej., *Hermano*(*Alice, Bob*)
  - Mundos posibles: objetos (tipados) y relaciones (tipadas)
  - Semántica: las sentencias en la base de datos son verdad, todo lo demás es falso
    - No pueden expresar disyunción, implicación, principios universales, etc.
    - Lenguaje de consulta (SQL, etc.), normalmente alguna variante de la lógica de primer orden
    - A menudo se complementa con lenguajes de reglas de primer orden, como Datalog.
  - Grafos de conocimiento (aproximadamente: BD relacional + ontología de tipos y relaciones)
    - Google Knowledge Graph: 5.000 millones de entidades, 500.000 millones de hechos, >30% de las consultas
    - Red de Facebook: 2.930 millones de personas, billones de publicaciones, tal vez cuatrillones de hechos

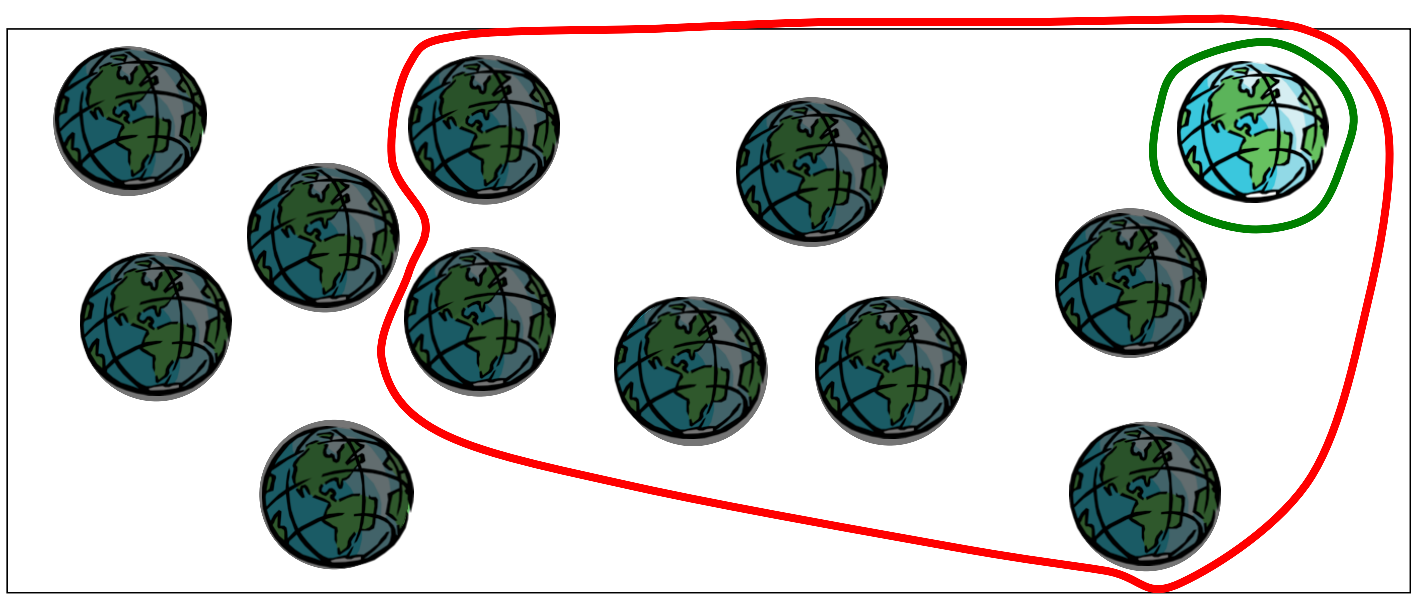
Inferencia: implicación

- ***Implicación***: $\alpha \vDash \beta$ ("α **implica** β" o "β **se sigue de** α") si y sólo si <u>en todos los mundos donde α es verdad, β también es verdad</u>
  - Es decir, los mundos α son un **<u>subconjunto</u>** de los mundos β:

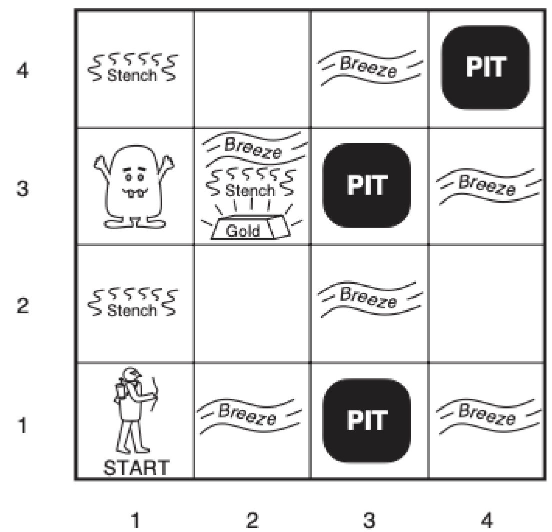
$$M(\alpha) \subseteq M(\beta)$$

- En el ejemplo, $\alpha_2 \vDash \alpha_1$
- (Pongamos que α_2 es $\neg Q \wedge R \wedge S \wedge W$
  α_1 es $\neg Q$)





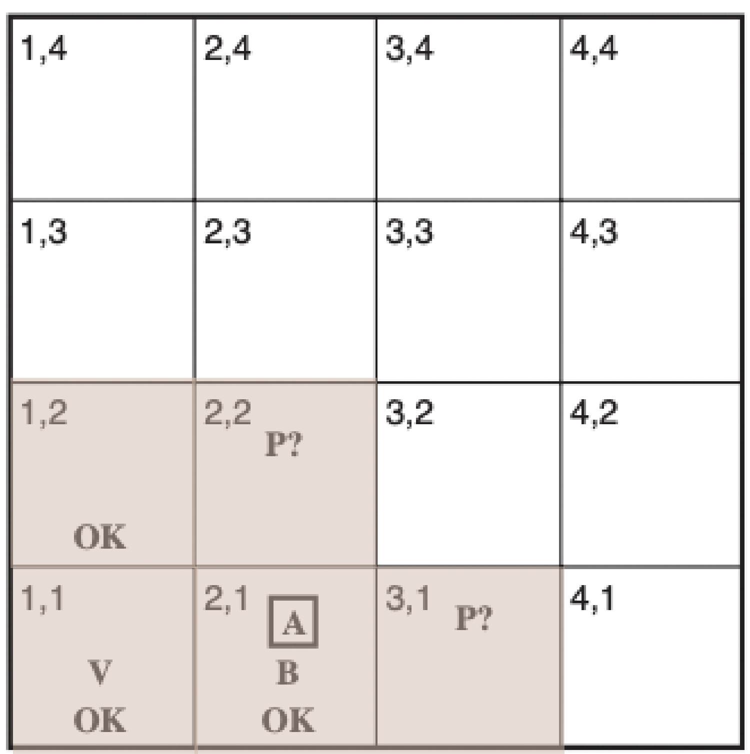
Mundo de Wumpus

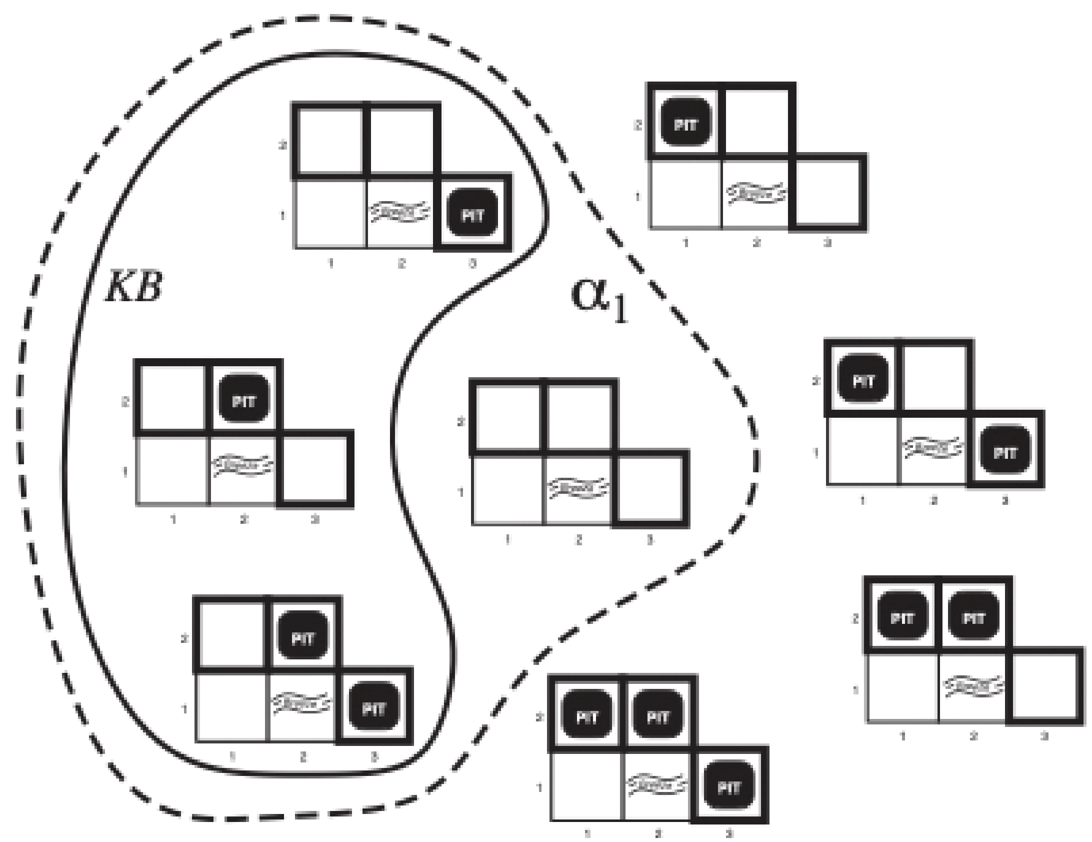


Modelos de la presencia de pozos en
[1, 2], [2, 2], [3, 1]

Percepción en [2,1]: [None, *Breeze*, None, None, None]

A = *Agent*
B = *Breeze*
G = *Glitter, Gold*
OK = *Safe square*
P = *Pit*
S = *Stench*
V = *Visited*
W = *Wumpus*

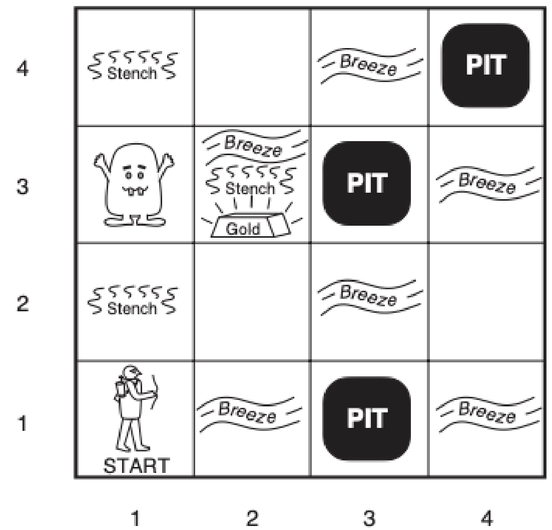




La línea punteada es $M(\alpha_1)$, donde $\alpha_1 =$ “no hay pozo en [1, 2]”

$\boxed{KB \models \alpha_1}$

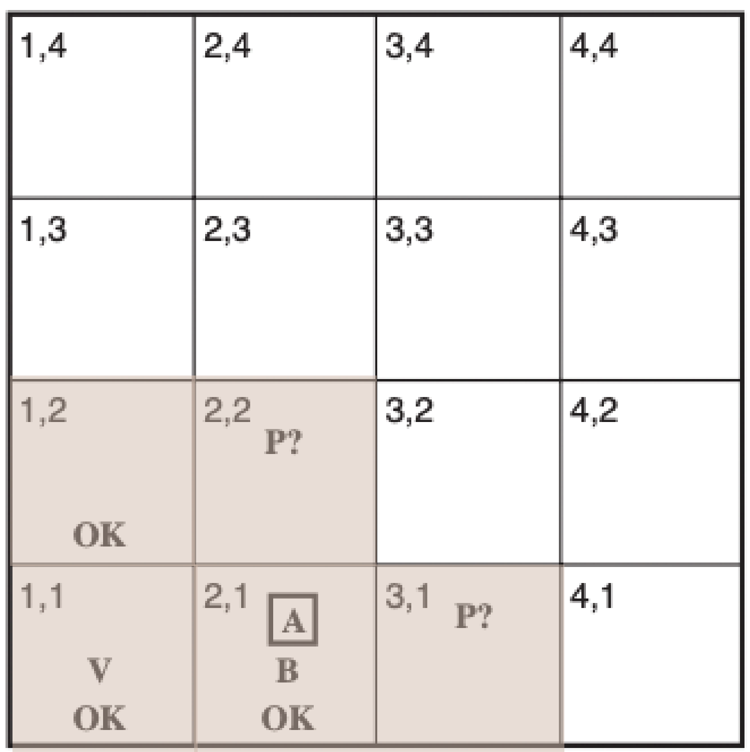
Mundo de Wumpus

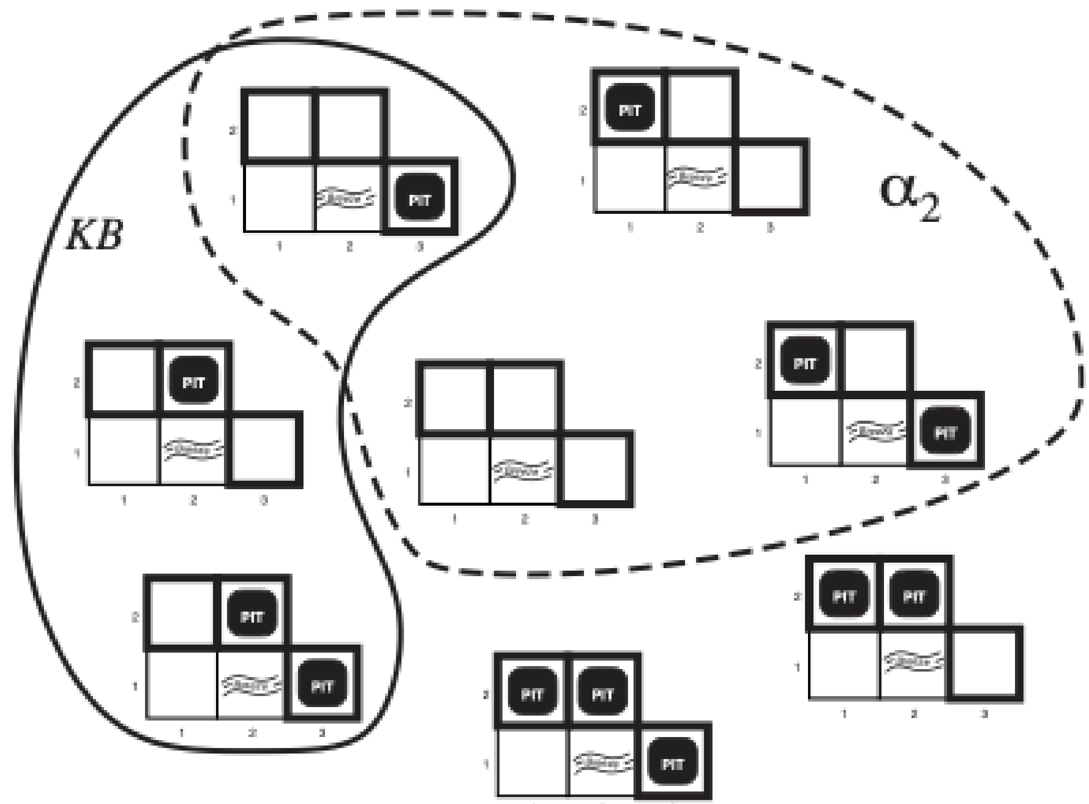


Modelos de la presencia de pozos en
[1, 2], [2, 2], [3, 1]

Percepción en [2,1]: [None, *Breeze*, None, None, None]

A = *Agent*
B = *Breeze*
G = *Glitter, Gold*
OK = *Safe square*
P = *Pit*
S = *Stench*
V = *Visited*
W = *Wumpus*





La línea punteada es $M(\alpha_2)$, donde $\alpha_2 =$ “no hay pozo en [2, 2]”

$$KB \nvDash \alpha_2$$

Inferencia: pruebas (demostraciones)

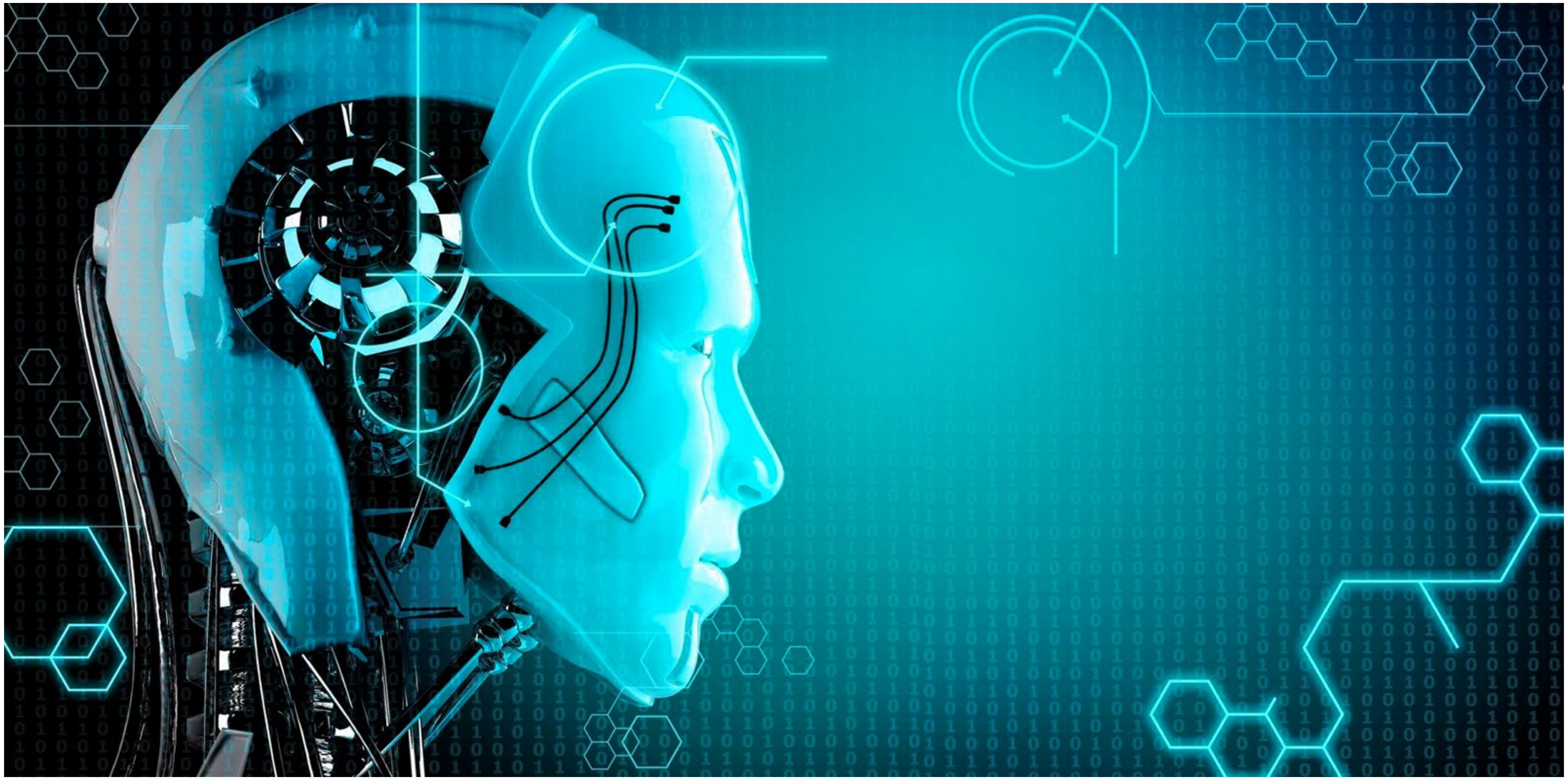
- Una prueba es una **demostración** de implicación entre α y β
- Algoritmo **sólido *(sound)***: deriva solo sentencias implicadas (no se inventa cosas)
- Algoritmo **completo**: puede derivar cualquier sentencia que esté implicada

Inferencia: pruebas (demostraciones)

- Método 1: ***comprobación de modelos*** (ejemplo de Wumpus)
  - Para cada mundo posible, si α es verdad, asegura que β también lo sea
  - OK para lógica proposicional (cantidad finita de mundos); no tan fácil para lógica de primer orden
- Método 2: ***demostración de teoremas*** (como en matemáticas)
  - Búsqueda de una secuencia de pasos de prueba (aplicación de las **reglas de inferencia**) que lleven de α a β
  - Por ejemplo, de P y $(P \Rightarrow Q)$, inferir Q por ***modus ponens*** ("el modo que, al afirmar, afirma)

Inteligencia Artificial

Lógica proposicional



[Transparencias adaptadas de Dan Klein and Pieter Abbeel: CS188 Intro to AI, UC Berkeley (ai.berkeley.edu)]

Universidad Rey Juan Carlos

Sintaxis de la lógica proposicional

- Sintaxis: sentencias permitidas
- Dado: un conjunto de símbolos de proposición $\{X_1,X_2,\ldots, X_n\}$
  - (a menudo se añaden Verdadero y Falso por conveniencia)
- Las sentencias complejas se construyen a partir de sentencias más simples mediante paréntesis y conectores lógicos:
  - ¬ (no): negación
  - ∧ (y): conjunción
  - ∨ (o): disyunción
  - ⇒ (implica): implicación
  - ⇔ (si y sólo si): bicondicional

Sintaxis de la lógica proposicional

- Dado: un conjunto de símbolos de proposición $\{X_1, X_2, \ldots, X_n\}$
  - (a menudo se añaden Verdadero y Falso por conveniencia)
- X_i es una sentencia
- Si α es una sentencia, entonces $\neg\alpha$ es una sentencia
- Si α y β son sentencias, entonces $\alpha \wedge \beta$ es una sentencia
- Si α y β son sentencias, entonces $\alpha \vee \beta$ es una sentencia
- Si α y β son sentencias, entonces $\alpha \Rightarrow \beta$ es una sentencia
- Si α y β son sentencias, entonces $\alpha \Leftrightarrow \beta$ es una sentencia
- ¡Y no hay más sentencias!

Semántica de la lógica proposicional

- Semántica: determinar la verdad de una sentencia con respecto a un modelo en particular
- Sea *m* un modelo que asigna verdadero o falso a $\{X_1, X_2, \ldots, X_n\}$ (definición de modelo)
- Si α es un símbolo, entonces su valor de verdad se da en *m*
- $\neg\alpha$ es verdad en *m* sii α es falso en *m*
- $\alpha \wedge \beta$ es verdad en *m* sii α es verdad en *m* y β es verdad en *m*
- $\alpha \vee \beta$ es verdad en *m* sii α es verdad en *m* o β es verdad en *m*
- $\alpha \Rightarrow \beta$ es verdad en *m* sii α es falso en *m* o β es verdad en *m*
- $\alpha \Leftrightarrow \beta$ es verdad en *m* sii $\alpha \Rightarrow \beta$ es verdad en *m* y $\beta \Rightarrow \alpha$ es verdad en *m*

Tablas de verdad

P	Q	$\neg P$	$P \wedge Q$	$P \vee Q$	$P \Rightarrow Q$	$P \Leftrightarrow Q$
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*

Ejemplo

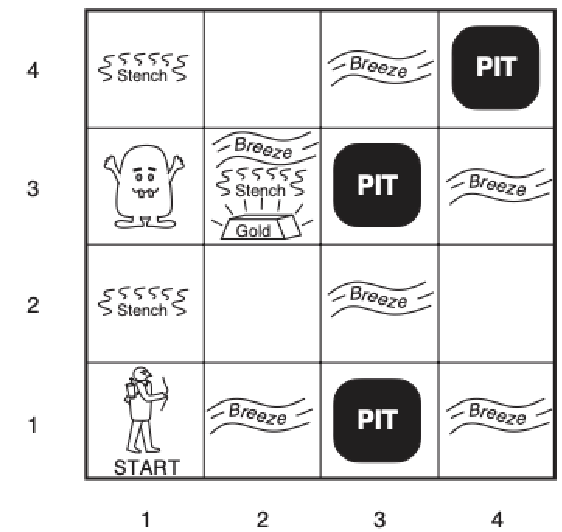
- Sea *m* {A=verdadero, B=verdadero, C=falso, D=falso}
- Sea α la sentencia $(A \wedge B) \vee (C \wedge \neg D)$, ¿verdadera o falsa en *m*?
- α es verdadero in *m* sii $(A \wedge B)$ es verdadero en *m* o $(C \wedge \neg D)$ es verdadero en *m*
  - $(A \wedge B)$ es verdadero en *m* sii A es verdadero en *m* y B es verdadero en *m*
  - $(A \wedge B)$ es verdadero en *m*
- α es verdadero en *m*

Ejemplo

- Sea α la sentencia $(A \wedge B) \vee (C \wedge \neg D)$
- ¿En cuántos modelos es verdadero α?

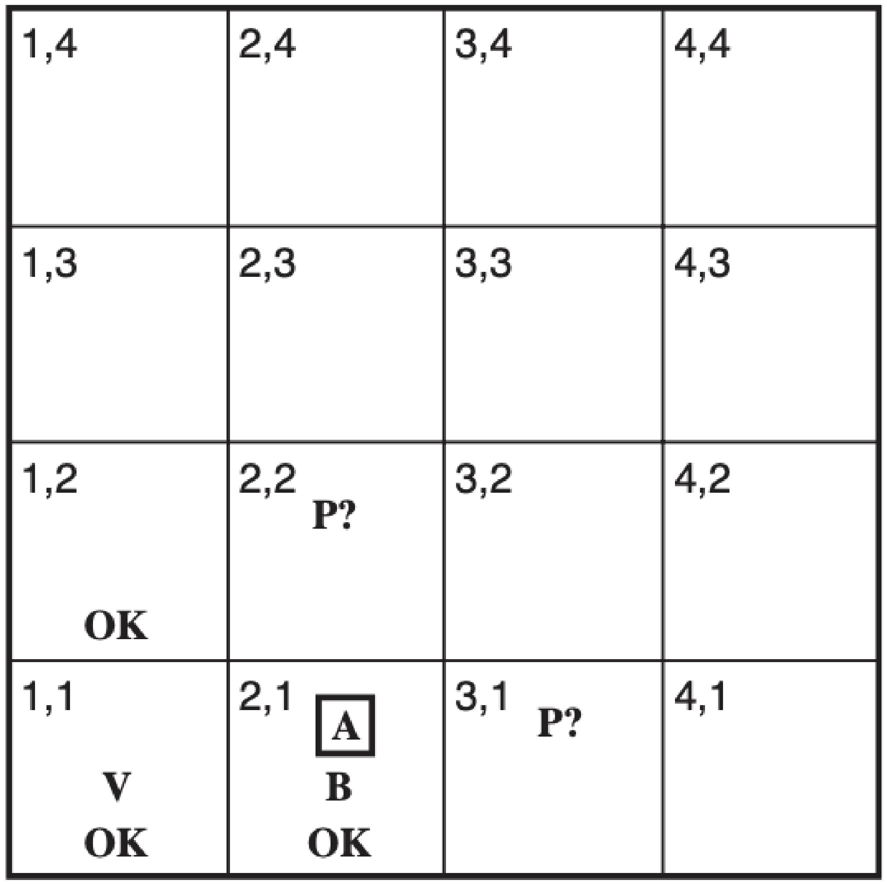
Una KB simple (Wumpus)

- $P_{x,y}$ es verdadero si hay un pozo en [x, y].
- $W_{x,y}$ es verdadero si hay un Wumpus en [x, y], vivo o muerto.
- $B_{x,y}$ es verdadero si el agente percibe una brisa en [x, y].
- $S_{x,y}$ es verdadera si el agente percibe un hedor en [x, y].



Sentencias para derivar $\neg P_{1,2}$

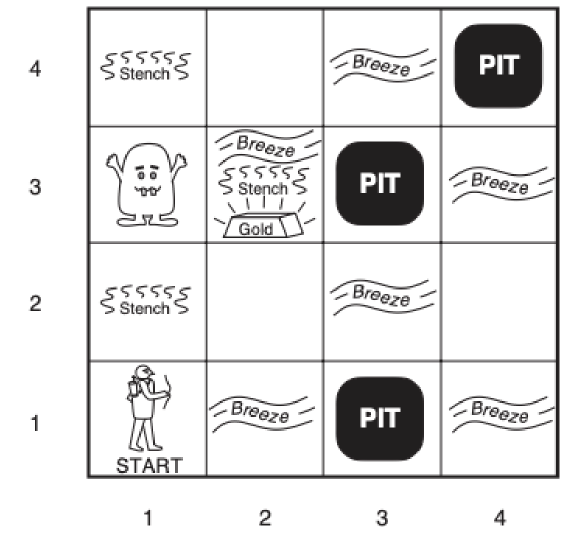
- No puede haber pozo en la casilla de salida
  - $R_1: \neg P_{1,1}$
- Por cada casilla, establecemos la condición de que haya brisa
  - $R_2: B_{1,1} \Leftrightarrow (P_{1,2} \vee P_{2,1})$
  - $R_3: B_{2,1} \Leftrightarrow (P_{1,1} \vee P_{2,2} \vee P_{3,1})$
- Las percepciones que llevamos hasta el momento
  - $R_4: \neg B_{1,1}$
  - $R_5: B_{2,1}$



Un proceso de inferencia simple (Wumpus)

- El objetivo es decidir $KB \vDash \alpha$ para algún α
- Por ejemplo, pongamos que α es $\neg P_{1,2}$
- Veamos los símbolos relevantes:

  $B_{1,1}, B_{2,1}, P_{1,1}, P_{1,2}, P_{2,1} P_{2,2}, P_{3,1}$
- KB es verdadero si $R_1 \dots R_5$ son verdaderas



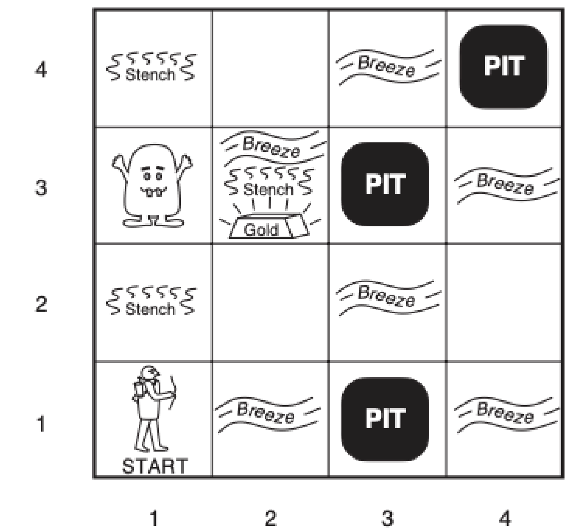
$B_{1,1}$	$B_{2,1}$	$P_{1,1}$	$P_{1,2}$	$P_{2,1}$	$P_{2,2}$	$P_{3,1}$	R_1	R_2	R_3	R_4	R_5	BC
falso *falso* ⋮ *falso*	*falso* *falso* ⋮ *verdadero*	*falso* *falso* ⋮ *falso*	*falso* *falso* ⋮ *falso*	*falso* *falso* ⋮ *falso*	*falso* *falso* ⋮ *falso*	*falso* *verdadero* ⋮ *falso*	*verdadero* *verdadero* ⋮ *verdadero*	*verdadero* *verdadero* ⋮ *verdadero*	*verdadero* *falso* ⋮ *falso*	*verdadero* *verdadero* ⋮ *verdadero*	*falso* *falso* ⋮ *verdadero*	*falso* *falso* ⋮ *falso*
falso *falso* *falso*	*verdadero* *verdadero* *verdadero*	*falso* *falso* *falso*	*falso* *falso* *falso*	*falso* *falso* *falso*	*falso* *verdadero* *verdadero*	*verdadero* *falso* *verdadero*	*verdadero* *verdadero* *verdadero*	*verdadero* *verdadero* *verdadero*	*verdadero* *verdadero* *verdadero*	*verdadero* *verdadero* *verdadero*	*verdadero* *verdadero* *verdadero*	*verdadero* *verdadero* *verdadero*
falso ⋮ *verdadero*	*verdadero* ⋮ *verdadero*	*falso* ⋮ *verdadero*	*falso* ⋮ *verdadero*	*verdadero* ⋮ *verdadero*	*falso* ⋮ *verdadero*	*falso* ⋮ *verdadero*	*verdadero* ⋮ *falso*	*falso* ⋮ *verdadero*	*falso* ⋮ *verdadero*	*verdadero* ⋮ *falso*	*verdadero* ⋮ *verdadero*	*falso* ⋮ *falso*

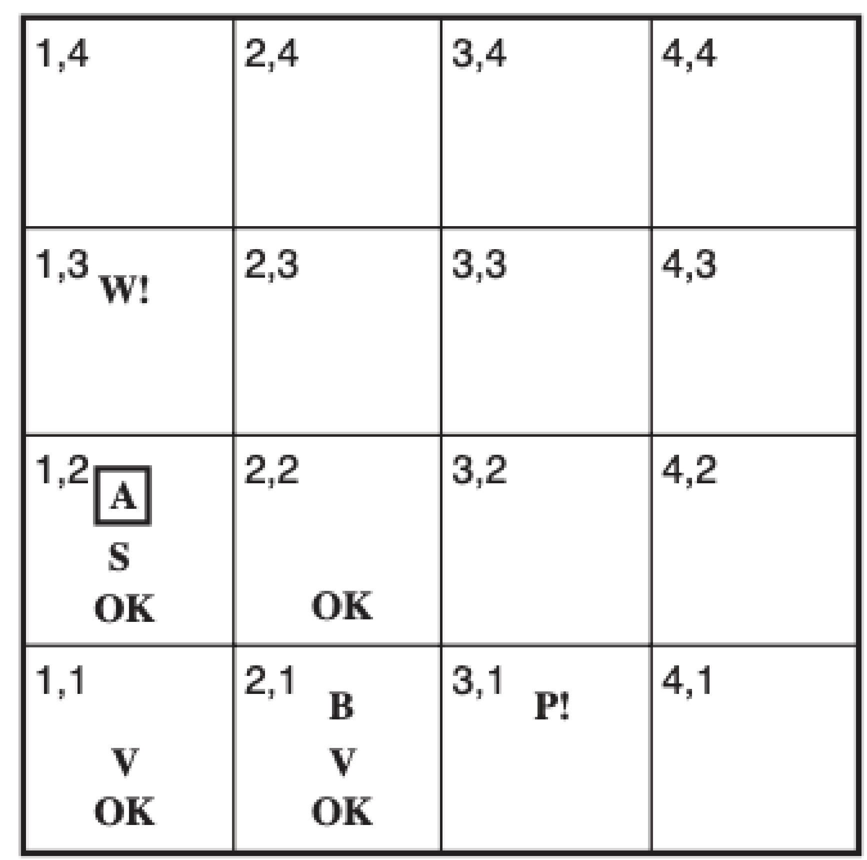
Ejercicio

Supón que el agente ha progresado hasta la situación que se muestra en la figura de abajo, sin haber percibido nada en [1,1], una brisa en [2,1] y un hedor en [1,2], y ahora se ocupa del contenido de [1,3], [2,2] y [3,1]. Cada uno de estos puede contener un pozo y, como máximo, uno puede contener un Wumpus. Construye el conjunto de mundos posibles (deberías encontrar 32 de ellos). Marca los mundos en los que la KB es verdadera y aquellos en los que cada una de las siguientes sentencias es verdadera:

- α_2 ="No hay hoyo en [2, 2]"
- α_3 ="No hay hoyo en [1, 3]"

Por tanto, demuestra que $KB \vDash \alpha_2$ y $KB \vDash \alpha_3$.



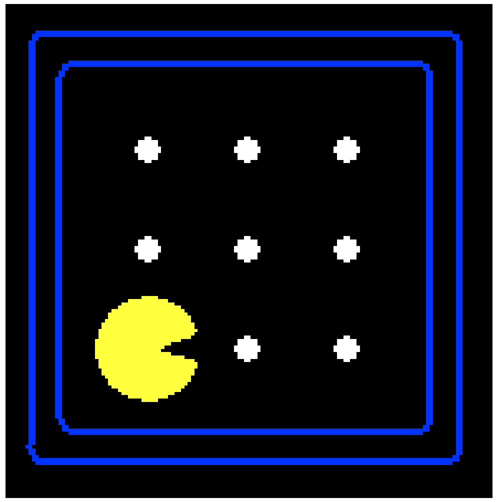


Vámonos ahora al mundo de Pacman...

- Vamos a decirle al agente lógico lo que sabemos sobre la física de Pacman (*PacPhysics*)
- Y, después, a preguntarle qué acciones tienen que ser verdad para conseguir el objetivo

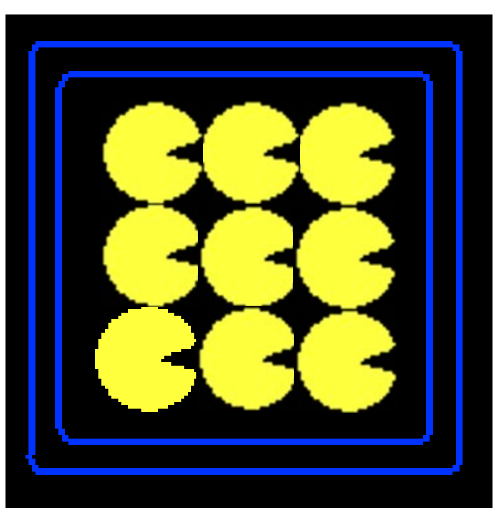
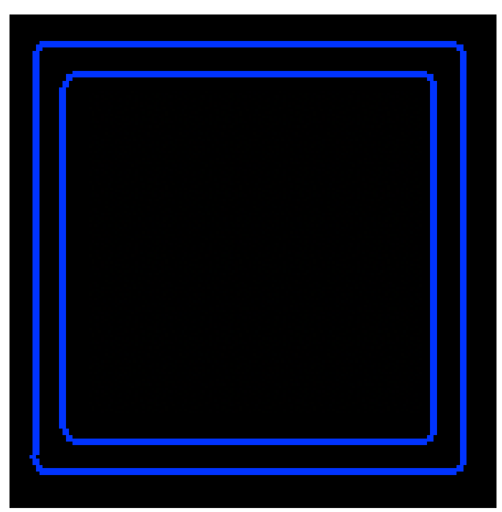
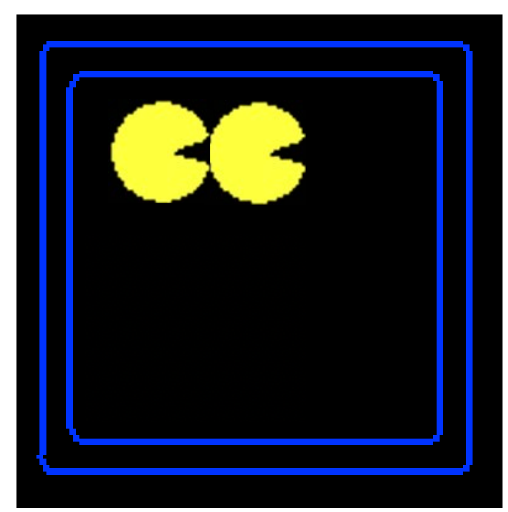
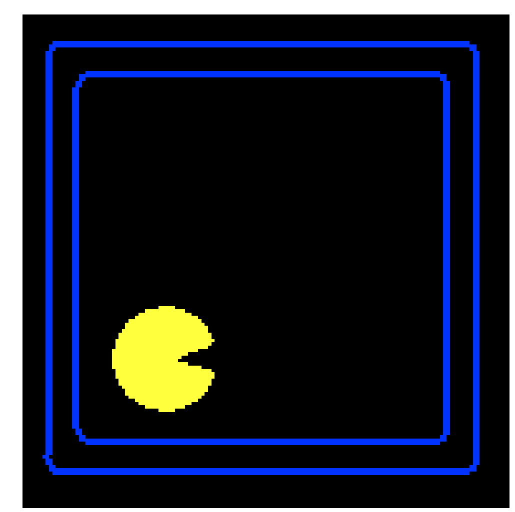
Pacman parcialmente observable

- Pacman conoce el mapa, pero percibe solo el muro/el hueco en su NSEO
- Formulación: ***¿qué variables necesitamos?***
  - Posiciones de la pared
    - Wall_0,0 hay una pared en [0,0]
    - Wall_0,1 hay una pared en [0,1], etc. (*N* símbolos para *N* posiciones)
  - Percepciones
    - ~~Blocked_W (bloqueado por una pared a mi oeste) etc.~~
    - Blocked_W_0 (bloqueado por una pared a mi oeste ***en el tiempo 0***) etc. (4 *T* símbolos para *T* pasos temporales)
  - Acciones
    - W_0 (Pacman se mueve hacia el oeste en el tiempo 0), E_0 etc. (4 *T* símbolos)
  - Posición de Pacman
    - At_0,0_0 (Pacman es en [0,0] en el tiempo 0), At_0,1_0 etc. (*NT* símbolos)



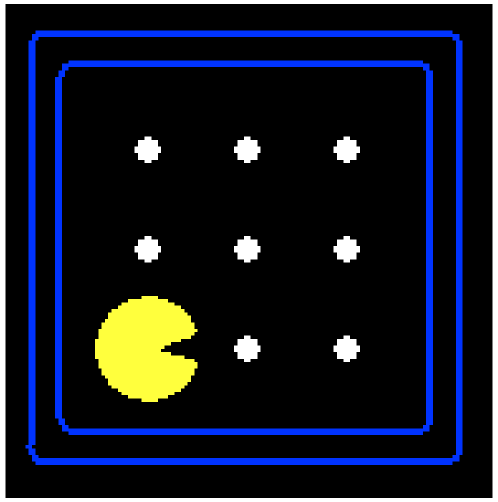
¿Cuántos mundos posibles?

- N posiciones, T pasos temporales → $N + 4T + 4T + NT = O(NT)$ variables
- ¡$O(2^{NT})$ mundos posibles!
- N=200, T=400 → ~10^{24000} mundos
- Cada mundo es una “historia” completa
  - Pero la mayoría son muy raros

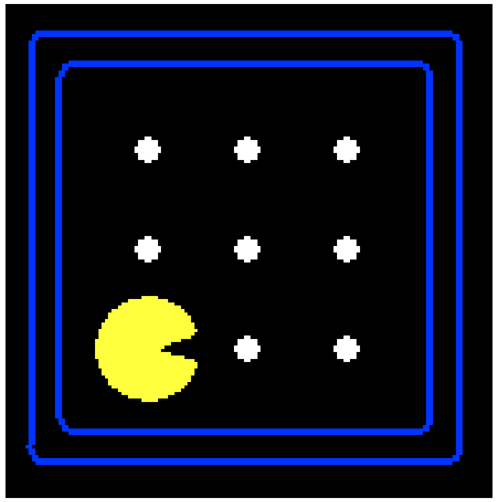
Base de conocimiento (KB) de Pacman: mapa

- Pacman sabe dónde están las paredes:
  - $Wall\_0,0 \wedge Wall\_0,1 \wedge Wall\_0,2 \wedge Wall\_0,3 \wedge Wall\_0,4 \wedge Wall\_1,4 \wedge \ldots$
- Pacman sabe dónde no están las paredes
  - $\neg Wall\_1,1 \wedge \neg Wall\_1,2 \wedge \neg Wall\_1,3 \wedge \neg Wall\_2,1 \wedge \neg Wall\_2,2 \wedge \ldots$



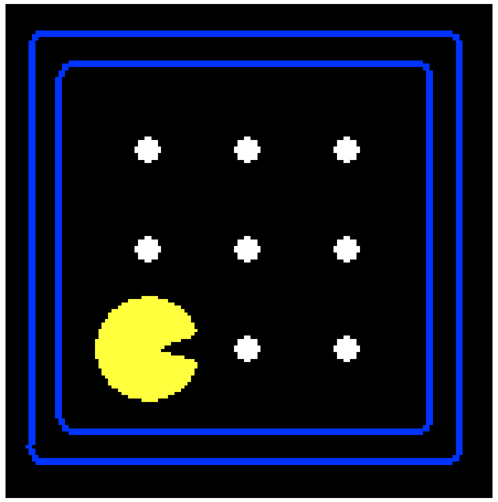
KB de Pacman: estado inicial

- Pacman no sabe dónde está
- Pero sabe que está en alguna parte
  - At_1,1_0 ∨ At_1,2_0 ∨ At_1,3_0 ∨ At_2,1_0 ∨ ...
- Y sabe que no está en más de un sitio
  - ¬ (At_1,1_0 ∧ At_1,2_0) ∧ ¬ (At_1,1_0 ∧ At_1,3_0) ...



KB de Pacman: modelo del sensor

- Enuncia hechos sobre cómo surgen las percepciones de Pacman...
  - <Variable percibida en t> ⇔ <alguna condición del mundo en t>
- Pacman percibe un muro al oeste en el tiempo *t* ***si y solo si*** está en *x,y* y hay una pared en *x-1,y*
  - Blocked_W_0 ⇔ ((At_1,1_0 ∧ Wall_0,1) v
    (At_1,2_0 ∧ Wall_0,2) v
    (At_1,3_0 ∧ Wall_0,3) v ....)
  - 4T sentencias, cada una de tamaño *O*(*N*)
  - Nota: son válidas para cualquier mapa



KB de Pacman: modelo de transición

- ¿Cómo obtiene cada **variable de estado** su valor en cada instante?
  - Aquí nos importan las variables de posicionamiento, p. ej., At_3,3_17
- Una variable de estado X toma su valor de acuerdo a un **axioma de sucesión de estados**
  - X_t ⇔ [X_t-1 ∧ ¬(alguna acción_t-1 lo hizo falso)] v
    [¬X_t-1 ∧ (alguna acción_t-1 lo hizo verdadero)]
- Para la localización de Pacman:

At_3,3_17 ⇔ [At_3,3_16 ∧ ¬((¬Wall_3,4 ∧ N_16) v (¬Wall_4,3 ∧ E_16) v ...)]
v [¬At_3,3_16 ∧ ((At_3,2_16 ∧ ¬Wall_3,3 ∧ N_16) v
(At_2,3_16 ∧ ¬Wall_3,3 ∧ N_16) v ...)]

¿Cuántas sentencias?

- La gran mayoría de la KB está ocupada por sentencias del modelo de transición O(NT)
  - Cada una de unas 10 líneas de texto
  - *N*=200, *T*=400 => ~800,000 líneas de texto, o 20,000 páginas
- La lógica proposicional tiene un **poder expresivo limitado**
- ¿Realmente vamos a escribir 20.000 páginas de sentencias?
- No, pero nuestro código va a generarlas
- (La búsqueda en el espacio de estados utiliza estados atómicos: ¿cómo mantener pequeña la representación del modelo de transición?)
  - Usando lenguajes expresivos como Python para representar lo que sabemos

Algunas tareas de razonamiento

- ***Localización*** con un mapa y percepción local:
  - Dadas una KB inicial, más una secuencia de percepciones y acciones, ¿dónde estoy?
- ***Mapeado*** con un sensor de localización:
  - Dadas una KB inicial, más una secuencia de percepciones y acciones, ¿cuál es el mapa?
- ***SLAM***
  - Dadas ..., ¿dónde estoy y cuál es el mapa?
- ***Planificación***:
  - Dadas ..., ¿qué secuencia de acciones está garantizado que va a alcanzar el objetivo?
- *Todos estos usan la misma KB y el mismo algoritmo*

Resumen

- Una posible arquitectura de agentes: conocimiento + inferencia
- La lógica proporciona una forma formal de codificar el conocimiento
  - Una lógica se define por: sintaxis, conjunto de mundos posibles y condición de verdad
- Una base de conocimientos simple para, por ejemplo, Pacman cubre el estado inicial, el modelo de sensor y el modelo de transición
- La inferencia lógica calcula las relaciones de implicación entre sentencias, lo que permite resolver una amplia gama de tareas