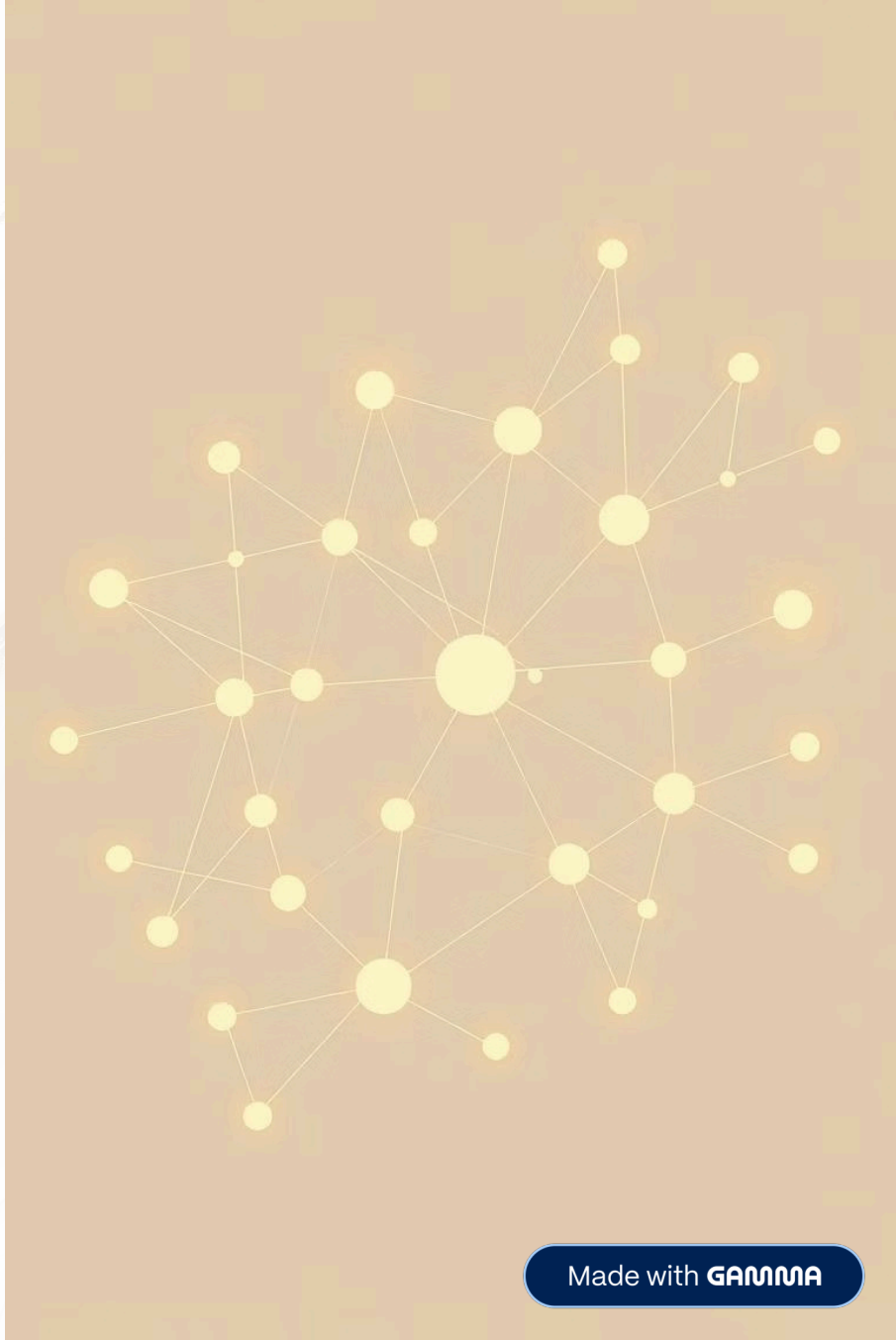
TP4: Representación del Conocimiento e Inferencia

3 de septiembre de 2025



Temas Clave del TP4

Representación del Conocimiento

Y Razonamiento Lógico.

Estrategias de Resolución

Encadenamiento hacia Adelante, Encadenamiento hacia Atrás y Resolución por Contradicción.

Representación Basada en Circuitos

Diseño lógico de proposiciones.

Ejercicios Teóricos: Inferencia

¿Qué es una inferencia?

Es el proceso de usar un modelo de IA entrenado para hacer predicciones o tomar decisiones sobre datos nuevos.

¿Cómo se verifica?

Se evalúa el desempeño del modelo con datos de validación y prueba que no ha visto. Métricas como precisión y exactitud cuantifican si el modelo generaliza correctamente su base de conocimiento.

Encadenamiento hacia Adelante

Probando que $a = True$ a partir de la base de conocimiento (axiomas 5 y 6).

01

Paso 1: Regla 2

$(d \wedge e \rightarrow b)$ entonces $b = True$, ya que d y e son siempre verdaderas.

02

Paso 2: Regla 4

$(e \rightarrow c)$ entonces $c = True$, ya que e siempre es verdadera.

03

Paso 3: Regla 1

$(b \wedge c \rightarrow a)$ entonces $a = True$.

La propiedad clave para esta inferencia es la completitud: todo enunciado lógicamente derivable debe serlo por el algoritmo.

Encadenamiento hacia Atrás

Probando que $a = True$ asumiendo la hipótesis $a = True$.

01

Paso 1: Hipótesis

Asumimos $a = True$. Para que la Regla 1 ($b \wedge c \rightarrow a$) sea cierta, b y c deben ser verdaderos.

02

Paso 2: Probar b

Usando Regla 2 ($d \wedge e \rightarrow b$). Como d y e son axiomas, $b = True$.

03

Paso 3: Probar c

Usando Regla 4 ($e \rightarrow c$). Como e es un axioma, $c = True$.

04

Conclusión

Al probar b y c como verdaderos, se demuestra que $a = True$.

Demostración por Contradicción

Convertimos la base de conocimiento a Forma Normal Conjuntiva (FNC) y buscamos la cláusula vacía {}.

01

Paso 1: Resolución

Resolvemos C5 (b) con C2 ($\neg b \vee \neg a$) para obtener $\neg a$.

02

Paso 2: Nueva Resolución

Resolvemos C5 (b) con C1 ($\neg b \vee c$) para obtener c.

03

Paso 3: Otra Resolución

Resolvemos C8 (c) con C3 ($\neg c \vee a$) para obtener a.

04

Paso 4: Contradicción

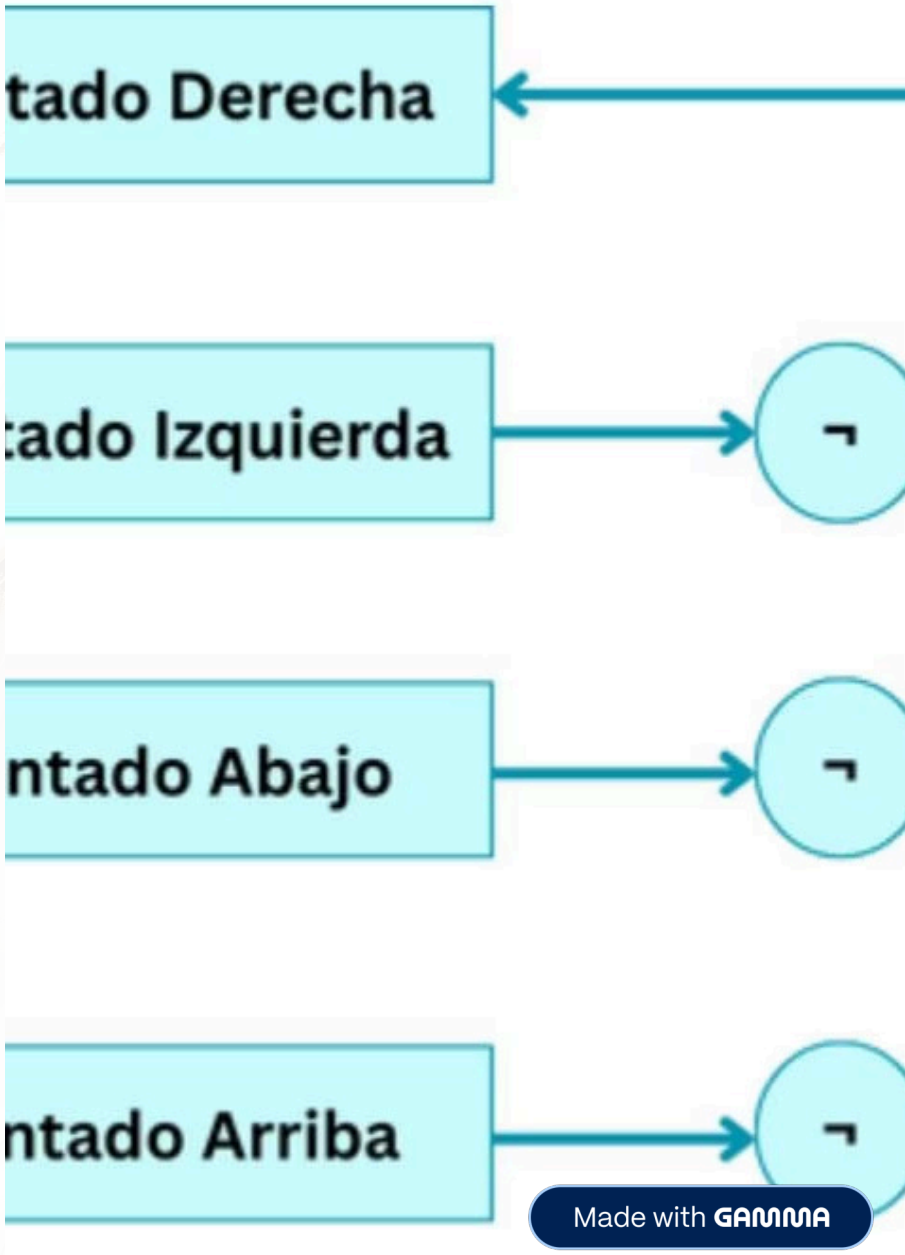
Resolvemos C9 (a) con nuestra suposición inicial C6 ($\neg a$), resultando en la cláusula vacía {}.

La derivación de la cláusula vacía demuestra que nuestra suposición inicial ($\neg a$) es falsa, por lo tanto, a es verdadera.

Lógica Proposicional Basada en Circuitos

Representación de "OrientadoDerecha" y "Agente ubicado en la casilla [1,2]" para el mundo de Wumpus de 4x4.

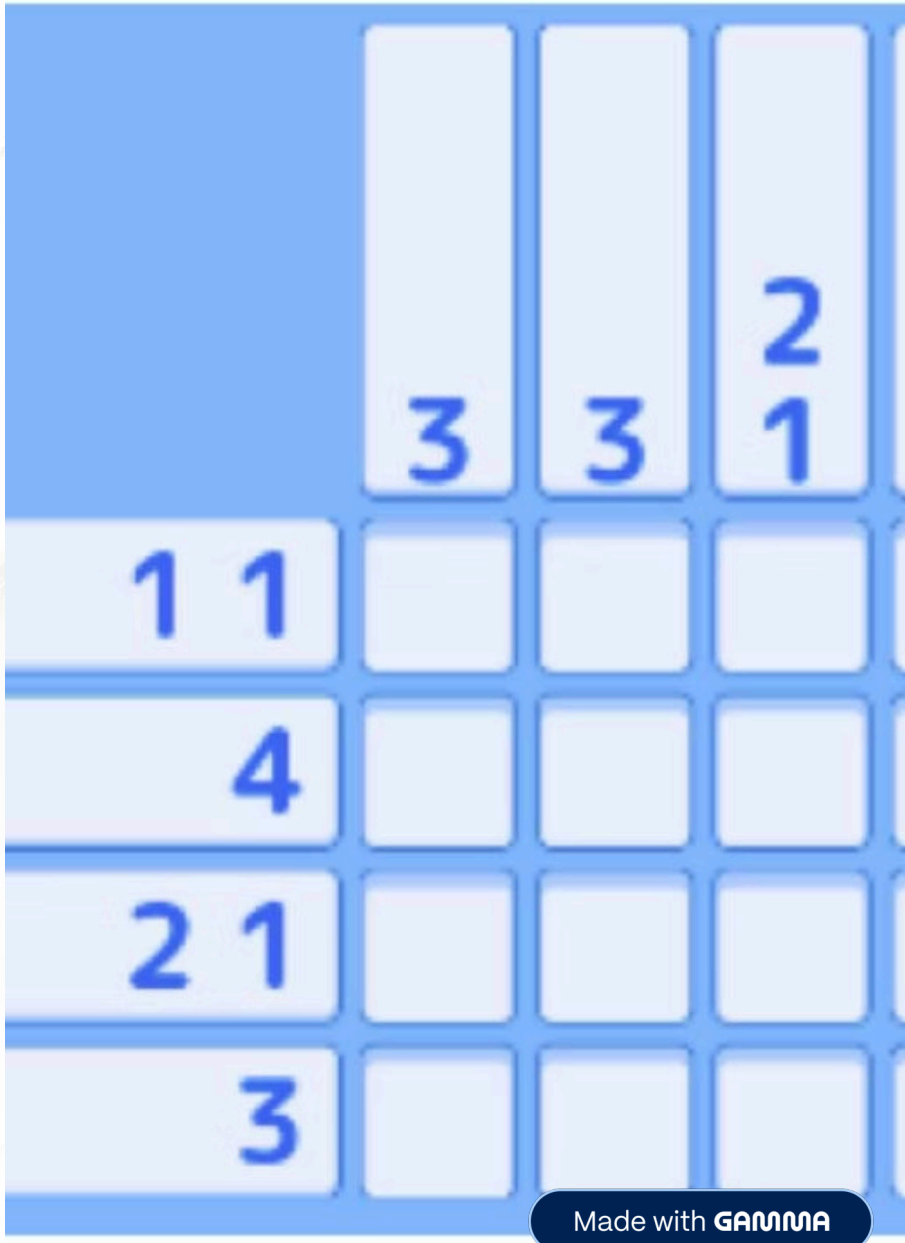
Este circuito visualiza las condiciones lógicas para la orientación del agente y su posición específica en el tablero.



Resolución de Nonograma

Aplicando reglas de inferencia para completar el tablero.

- **R1:** Fila 2 ($P_{2,1} \land P_{2,2} \land P_{2,3} \land P_{2,4}$)
- **R2:** Fila 3 ($P_{3,1} \land P_{3,2} \land \neg P_{3,3} \land P_{3,4}$)
- **R3:** Columna 3 ($P_{1,3} \land P_{2,3} \land \neg P_{3,3} \land P_{4,3}$)
- **R4:** $P_{1,3} \rightarrow (\neg P_{1,2} \land \neg P_{1,4})$
- **R5:** $\neg P_{1,2} \rightarrow (P_{1,1} \land P_{1,3})$
- **R6:** $\neg P_{1,2} \rightarrow (P_{2,2} \land P_{3,2} \land P_{4,2})$
- **R7:** $\neg P_{1,4} \rightarrow (P_{2,4} \land P_{3,4} \land P_{4,4})$
- **R8:** $(P_{4,2} \land P_{4,3} \land P_{4,4}) \rightarrow \neg P_{4,1}$



Motores de Inferencia Implementados

Encadenamiento hacia Adelante

Deriva nuevas conclusiones a partir de hechos conocidos y reglas, avanzando desde las premisas hacia el objetivo.

```
def forward_chaining(reglas, hechos_iniciales, objetivo):
    hechos_conocidos = set(hechos_iniciales)
    while True:
        nuevos_hechos_derivados = False
        for premisa, conclusion in reglas:
            if premisa.issubset(hechos_conocidos):
                if conclusion not in hechos_conocidos:
                    hechos_conocidos.add(conclusion)
                    nuevos_hechos_derivados = True
        if not nuevos_hechos_derivados:
            break
    return objetivo in hechos_conocidos
```

Encadenamiento hacia Atrás

Parte del objetivo y busca reglas que puedan probarlo, trabajando hacia atrás hasta encontrar hechos iniciales.

```
def backward_chaining(reglas, hechos_iniciales, objetivo):
 if objetivo in hechos_iniciales:
 return True
 reglas_posibles = [r for r in reglas if r[1] == objetivo]
 if not reglas_posibles:
 return False
 for premisa, _ in reglas_posibles:
 todos_subobjetivos_probados = True
 for sub_objetivo in premisa:
 if not backward_chaining(reglas, hechos_iniciales,
sub_objetivo):
 todos_subobjetivos_probados = False
 break
 if todos_subobjetivos_probados:
 return True
 return False
```

Motor de Inconsistencia por Resolución

Detecta si un conjunto de proposiciones es inconsistente derivando la cláusula vacía.

```
def motor_inconsistencia_resolucion(clausulas):
    clausulas_conocidas = set(clausulas)
    while True:
        nuevas_clausulas = set()
        pares_clausulas = itertools.combinations(clausulas_conocidas, 2)
        for c1, c2 in pares_clausulas:
            resultado_resolucion = resolver_clausulas(c1, c2)
            if resultado_resolucion is not None:
                if not resultado_resolucion: # La cláusula vacía {}
                    return True # Inconsistencia detectada
                if resultado_resolucion not in clausulas_conocidas:
                    nuevas_clausulas.add(resultado_resolucion)
        if not nuevas_clausulas:
            return False # No se encontraron nuevas cláusulas
        clausulas_conocidas.update(nuevas_clausulas)
```

Bibliografía

- Russell, S. & Norvig, P. (2004) Inteligencia Artificial: Un Enfoque Moderno. Pearson Educación S.A. (2a Ed.) Madrid, España
- Poole, D. & Mackworth, A. (2023) Artificial Intelligence: Foundations of Computational Agents. Cambridge University Press (3a Ed.) Vancouver, Canada