Ayudantía 3

Repaso Clingo + Apoyo T2

Por Ignacio Villanueva y
Jerónimo Infante

30 de agosto de 2024

Contenidos

1. Repaso Clingo
2. Repaso Modelación
3. Apoyo T2

Repaso Clingo

Reglas

Body con varios átomos

- Generan una **conjunción** de proposiciones; es decir, se debe cumplir todo en *Body* para que la regla se exija

```
a.                  % a se encuentra en el modelo
b.                  % b se encuentra en el modelo
c :- a, b.          % c está sólo si a y b lo están
d :- a, m.          % d está sólo si a y m lo están
```

El modelo es **{a, b, c}**

Reglas

Head con varios átomos

- Generan una **disyunción** de proposiciones; es decir, cuando se cumple el *Body*, se cumple sólo uno de los átomos del *Head*
- A excepción de que se fuerce la presencia de más átomos

```
p.
q, r, k :- p.
```

Los modelos son **{p,q}, {p,r}, {p,k}.**

Reglas

Predicados con variables

```
pajaro(carpintero).
pajaro(martin_pescador).
pajaro(condor).
vuela(Z) :- pajaro(Z).
```

Las variables nos permiten **generalizar** reglas:
“Todo pájaro vuela”

*Las variables siempre se escriben en mayúscula y términos en minúscula

Restricciones de Cardinalidad

- En el contexto de la Head de una regla, estas permiten elegir **distintas combinaciones** de átomos o predicados para que aparezcan en los modelos.
- Por ejemplo, para el programa:

```
p.
{q; r} :- p.     % Si p está en el modelo,
                 % alguna combinación entre q y r también lo está
```

Las combinaciones pueden ser **{p}**, **{p,q}**, {**p,r}** y **{p,q,r}**.

Restricciones de Cardinalidad

Limitando combinaciones

- Por defecto, Clingo prueba con todas las combinaciones posibles.
- Puede limitarse el número de elementos a incluir rodeando con números el conjunto de la restricción:

```
p.
1{q; r; s}2 :- p.     % Si p está en el modelo, alguna combinación
                      % de 1 a 2 elementos entre q, r y s
                      % también lo está
```

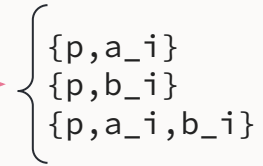
Ahora, las combinaciones pueden ser **{p;q}**, **{p;r}**, **{p;s}**, **{p,q;r}**, **{p;r;s}** y **{p;q;s}** (6 modelos).

Restricciones de Cardinalidad

Ejercicio

- Supongamos que tenemos un programa con N líneas del tipo:

```
p.
1 {a_1, b_1} 2 :- p.
1 {a_2, b_2} 2 :- p.
(...)
1 {a_n, b_n} 2 :- p.
```



× N veces *

El programa genera **3^N modelos distintos**

***** Para cada línea, tres combinaciones posibles

Restricciones de Cardinalidad

Condiciones dentro de las restricciones

- Al poner el carácter ":" es posible crear condiciones dentro de las restricciones de cardinalidad para generar reglas más complejas.

```
num(0..5).
3{seleccionado(X) : num(X)}3.     % selecciona 3 tal que sean num
#show seleccionado/1.             % muestra los seleccionados
```

Hay 20 modelos posibles.

Restricciones de Cardinalidad

Condiciones dentro de las restricciones

```
#const n = 5.

tiempo(1..n).
persona(pedro).

1{estudia(P, T) : tiempo(T)}5 :- persona(P).
```

¿Qué simula el programa anterior?

Restricciones de Cardinalidad

Condiciones dentro de las restricciones

```
#const n = 5.

tiempo(1..n).
persona(pedro).

1{estudia(P, T) : tiempo(T)}5 :- persona(P).
```

¡Incluye todos los modelos en los que Pedro estudia entre los tiempos 1 y 5!

Negación

- Podemos formar nuevos predicados:

```
desmayarse(P):- not desayunar(P), not dormir(P).
```

- Podemos evitar redundancias:

```
catedra(X,Y):- profesor(X), profesor(Y), not catedra(Y,X), X!=Y.
```

Repaso Modelación

Modelación

Algunos tips

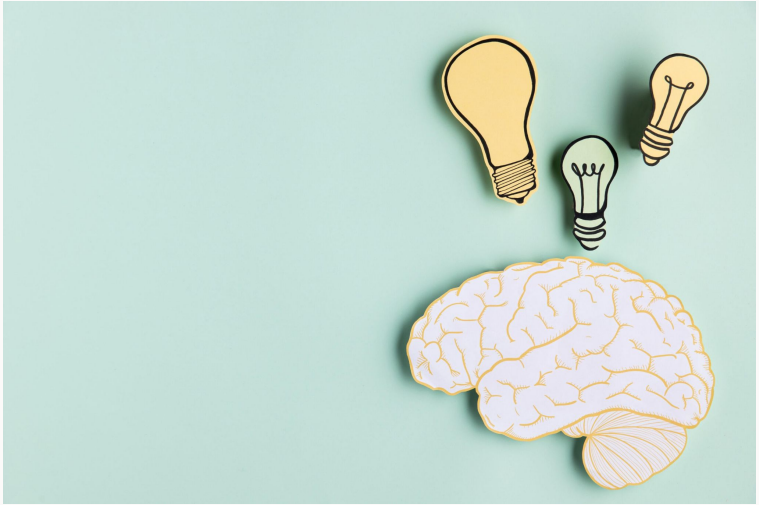
- Dado que Clingo es un lenguaje declarativo, pensar en el problema **resuelto**, no en cómo resolverlo.
- Probar que las reglas funcionan **individualmente** sirve para entender qué funciona y qué no.
- Soltar la mano, especialmente pasando predicados lógicos a Clingo.
- ¡Ejercitar! Hay muchos ejemplos básicos, medios y avanzados.

Apoyo T2

Parte 1: Reflexión y Teoría

Algunos tips

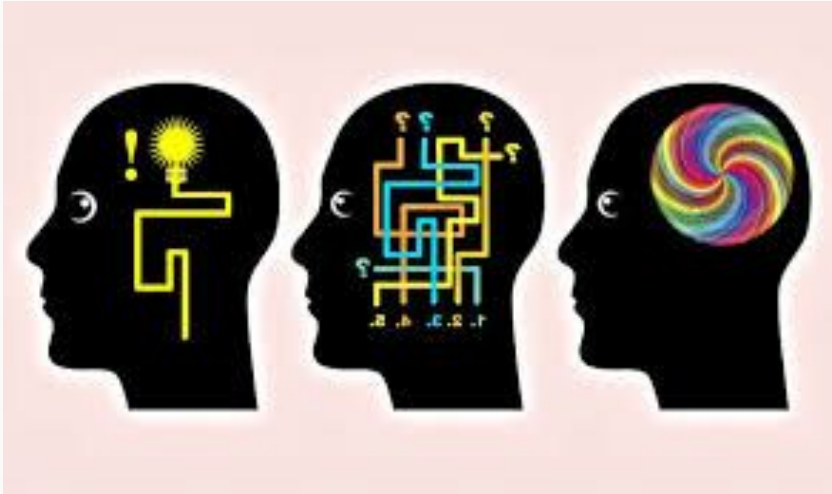
- Sean claros y concretos.
- Vean el video con atención.



Parte 2: Teoría de ASP

Algunos tips

- Lean y escriban las definiciones.
- Piensen en cómo usar las definiciones para responder la pregunta.
- Escriban lo más claro posible. No se les pide rigurosidad matemática, pero que se entienda la idea.



Parte 2: Teoría de ASP

Pregunta 1.a, 2 y 3:

Definición

Un conjunto A que cumple una propiedad P es un conjunto **minimal** que cumple P ssi no existe un subconjunto propio de A que cumpla P.

Definición

M es un modelo de un programa Π si es un conjunto minimal que satisface que para cada regla $Head \leftarrow Tail \in \Pi$ tal que $Tail \subseteq M$, se cumple que $Head \cap M \neq \emptyset$.

Parte 2: Teoría de ASP

Para construir un modelo M de un programa básico Π:

1. Agregamos todos los hechos del programa Π a M.
2. Si hay una regla $Head \leftarrow Tail$ en Π que cumple:
   (a) $Head$ no está contenido en M
   (b) Todos los elementos de $Tail$ están en M

   entonces agregamos $Head$ a M.
3. Si el paso 2 agrega algo a M volvemos al paso 2. En caso contrario, terminar.

Parte 2: Teoría de ASP

Pregunta 1.b:

Definición (Reducción)

La *reducción* un programa Π relativa a un conjunto X, denotada por Π^X es la que resulta de hacer:

1. $\Pi^X := \Pi$
2. **Borrar** toda regla $Head \leftarrow Pos \cup not(Neg)$ de Π^X cuando $Neg \cap X \neq \emptyset$.
3. **Reemplazar** cada regla $Head \leftarrow Pos \cup not(Neg)$ en Π^X por $Head \leftarrow Pos$ cuando $Neg \cap X = \emptyset$.

Definición (Modelo de un programa con negación)

X es un modelo de un programa con negación Π ssi X es un modelo para Π^X.

Parte 2: Teoría de ASP

Pregunta 2 y 3:

Definición

Un conjunto A que cumple una propiedad P es un conjunto **minimal** que cumple P ssi no existe un subconjunto propio de A que cumpla P.

Definición

M es un modelo de un programa Π si es un conjunto minimal que satisface que para cada regla $Head \leftarrow Tail \in \Pi$ tal que $Tail \subseteq M$, se cumple que $Head \cap M \neq \emptyset$.

Parte 2: Teoría de ASP

Para construir un modelo M de un programa básico Π:

1. Agregamos todos los hechos del programa Π a M.
2. Si hay una regla $Head \leftarrow Tail$ en Π que cumple:
   (a) $Head$ no está contenido en M
   (b) Todos los elementos de $Tail$ están en M

   entonces agregamos $Head$ a M.
3. Si el paso 2 agrega algo a M volvemos al paso 2. En caso contrario, terminar.

Parte 2: Teoría de ASP

Suponga que deseamos demostrar una afirmación $\forall x.\, P(x)$ sobre $\mathbb{N}$.

Principio de inducción

Para una afirmación $P(x)$ sobre los naturales, si $P(x)$ cumple que:

1. $P(0)$ es verdadero,
2. si $P(n)$ es verdadero, entonces $P(n+1)$ es verdadero,

entonces para todo n en los naturales se tiene que $P(n)$ es verdadero.

Notación

- $P(0)$ se llama el **caso base**.
- En el paso 2.
  - $P(n)$ se llama la **hipótesis de inducción**.
  - $P(n+1)$ se llama la **tesis de inducción** o paso inductivo.

Fuente: Clases Matemáticas Discretas Profesor Cristián Riveros

Parte 2: Teoría de ASP

Definición

Para dos conjuntos A y B, diremos que A es **subconjunto** de B si, y solo si:

$$\forall x. \quad x \in A \rightarrow x \in B$$

Si A es subconjunto de B escribiremos $A \subseteq B$.

Fuente: Clases
Matemáticas Discretas
Profesor Cristián Riveros

Parte 2: Teoría de ASP

Ojo:

- La pregunta 2 no incluye problemas con cabeza vacía y solo de tamaño 1.
- La pregunta 3 incluye problemas con cabezas de tamaño mayor o igual a 1.

Parte 3: DCConecta

Algunos tips

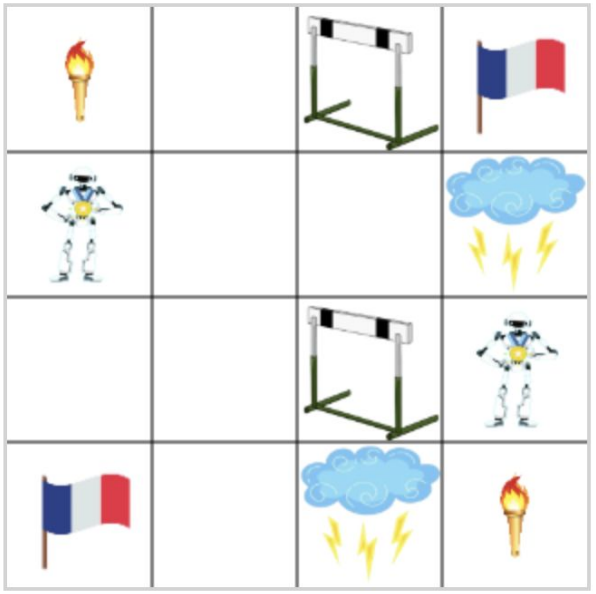
- Dibujen el problema, el grafo.
- Rayen sus dibujos, prueben sus soluciones en papel.

OJO: Para conectados por n/3, piensen en una solución recursiva inductiva.

Parte 4: DCCarrera Olímpica

¿Qué es?

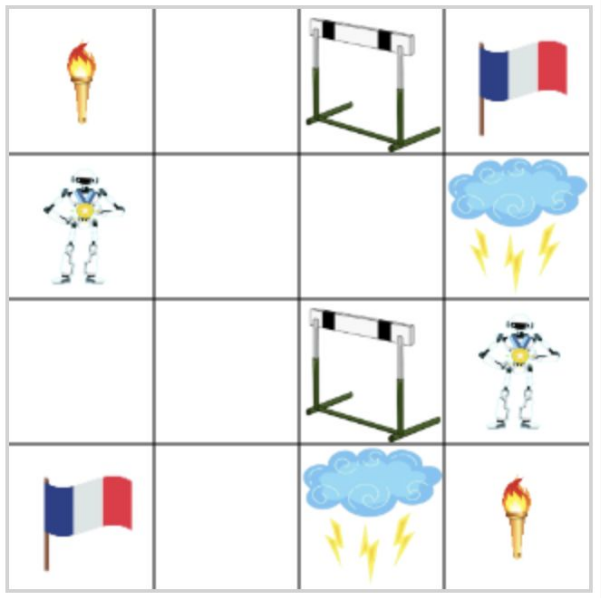
- Un juego de simulación que consiste en implementar que los robots presentes en el mapa lleguen a la meta.
- Se presentan algunas variantes, como las vallas, los obstáculos de clima, y las antorchas.



Parte 4: DCCarrera Olímpica

Algunos tips

- Revisar el código que se entrega para entender los predicados ya definidos.
- Pensar en cómo sería una solución completa, más que ir regla por regla.



Tips Generales

- Con Clingo, partir con una solución relajada para agregar poco a poco las demás reglas. Obtener un modelo (satisfacible) es un buen avance.
- Revisar las issues, y **preguntar** si es que necesitan.
- Comentar con más personas (OJO! Las preguntas, **no las soluciones**).
- Tiempo pensando > tiempo programando.

Ayudantía 3

Repaso Clingo + Apoyo T2

Por Ignacio Villanueva y Jerónimo Infante

30 de agosto de 2024