

Tema 8: Planificación

José L. Ruiz Reina

Dpto. de Ciencias de la Computación e Inteligencia Artificial

UNIVERSIDAD DE SEVILLA

Planificación en IA

- **Planificar:** encontrar una secuencia de acciones que alcanzan un determinado objetivo si se ejecutan desde un determinado estado inicial.
- **Plan:** secuencia de acciones que consiguen el objetivo
- **Aplicaciones del mundo real:**
 - Robótica
 - Fabricación mediante ensamblado de componentes
 - Misiones espaciales
- *Planning vs Scheduling*

El problema de la planificación en IA

- Cuestiones a abordar:
 - Representación del mundo y de las acciones que lo transforman
 - Representación de planes
 - Algoritmos de búsqueda de planes
 - Minimizar los recursos consumidos por el plan
 - Tiempo en el que se realiza cada acción
 - Monitorizar la ejecución del plan, revisándolo en caso de errores o contingencias
- Por simplificar, en este capítulo supondremos:
 - Acciones deterministas, totalmente definidas por su especificación
 - Hipótesis del mundo cerrado

Planificación y búsqueda en espacio de estados

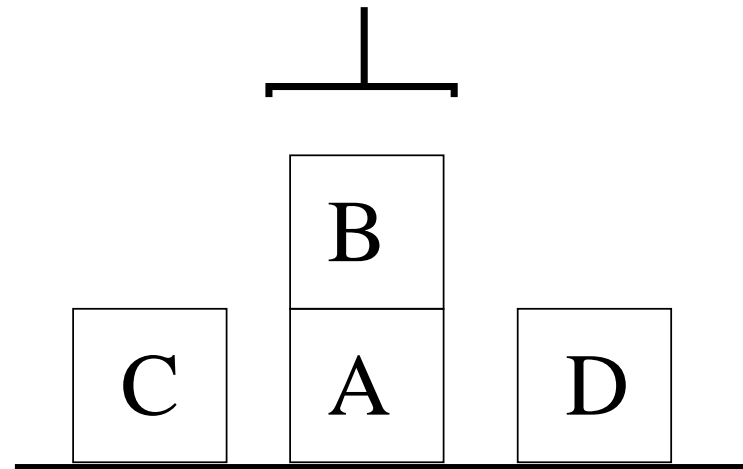
- Solucionar un problema de planificación usando los algoritmos ya vistos de búsqueda en espacio de estados *no funciona* en un problema de escala real, por los siguientes motivos:
 - Descripción de los estados en el mundo real extremadamente compleja
 - Gran cantidad de posibles acciones, muchas de ellas irrelevantes para la consecución del objetivo final
 - Las acciones sólo cambian una pequeña porción del mundo (*el problema del marco*)
 - Necesitamos heurísticas *independientes* del dominio
 - La necesidad de una acción puede detectarse sin necesidad de que se haya decidido las acciones previas (*compromiso mínimo*)
 - A veces es aconsejable descomponer en subproblemas más simples
- Idea: usar la lógica para representar estados, acciones y objetivos
 - Y algoritmos específicos que operan sobre esta representación

Un formalismo lógico: STRIPS

- Un lenguaje para representar problemas de planificación:
 - Constantes: objetos del mundo (en mayúsculas)
 - Variables para representar cualquier objeto (en minúsculas)
 - Símbolos de predicados (para expresar propiedades de los objetos)
 - Símbolos de acciones (para representar operadores)
- Terminología:
 - Átomos: fórmulas de la forma $P(o_1, \dots, o_n)$, donde P es un símbolo de predicado y cada o_i es una constante o una variable
 - Literales: átomos o negación de átomos
 - Literales cerrados: sin variables
- Estados: conjunción de literales cerrados
 - Hipótesis del mundo cerrado: las condiciones que no se mencionan se suponen falsas

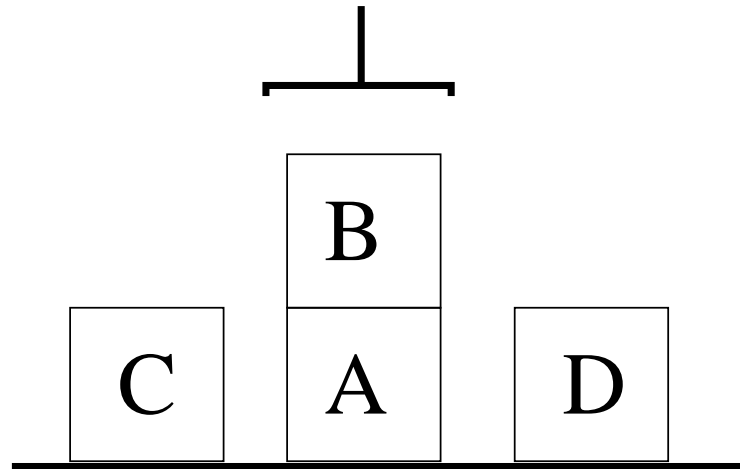
Ejemplo: el mundo de los bloques

- Ejemplo clásico en planificación.
- Elementos que intervienen:
 - Una superficie plana.
 - Una serie de bloques cúbicos.
 - Un brazo robotizado, que puede coger un bloque cada vez.
 - Un bloque puede estar sobre la mesa o apilado sobre otro bloque.



Representación de estados en el mundo de los bloques

- Descripción de un estado:



DESPEJADO(B), DESPEJADO(C), DESPEJADO(D), BRAZOLIBRE, SOBRE(B,A)
SOBRELAMESA(C), SOBRELAMESA(D), SOBRELAMESA(A)

- Predicados en el mundo de los bloques:

DESPEJADO(x), el bloque x está despejado.

BRAZOLIBRE, el brazo no agarra ningún bloque.

SOBRELAMESA(x), el bloque x está sobre la mesa.

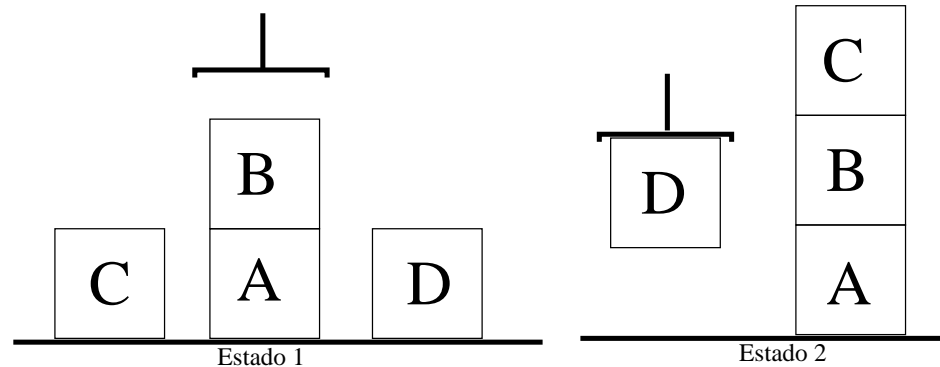
SOBRE(x,y), el bloque x está sobre el y.

AGARRADO(x), el bloque X está sujeto por el brazo.

Representación de objetivos en STRIPS

- **Objetivos:** descripción de los estados finales.
- Los objetivos se representan como conjunción de literales (con posibilidad de usar variables)
 - Las variables en los objetivos se interpretan como existencialmente cuantificadas
- **Satisfacer un objetivo:**
 - Un estado satisface un objetivo si es posible sustituir las variables del objetivo por objetos del mundo de manera que sus literales están incluidos en la descripción del estado
 - Un estado es estado final si satisface el objetivo requerido

Ejemplo de objetivos en el mundo de los bloques



- El objetivo $\text{SOBRE}(B,A)$, $\text{SOBRELAMESA}(A)$ es satisfecho por el estado 1 y por el estado 2
- El objetivo $\text{SOBRE}(x,A)$, $\text{DESPEJADO}(x)$, BRAZOLIBRE es satisfecho por el estado 1 pero no por el estado 2
- El objetivo $\text{SOBRE}(x,A)$, $\text{SOBRE}(y,x)$ no es satisfecho por el estado 1 pero sí por el estado 2

Descripción de acciones en STRIPS

- Representación de operadores o acciones:
 - $O(x_1, \dots, x_n)$ simboliza el operador O actuando sobre los objetos x_1, \dots, x_n .
- Descripción de la acción de un operador:
 - Para intentar solucionar el *problema del marco*, sólo se especifica lo que *cambia* por la acción del operador.
 - Usualmente, x_i es variable: esquemas de operadores.
- Descripción de la acción de un operador en estilo STRIPS, mediante tres listas:
 - Precondición: lista de literales que deben ser ciertos para que el operador pueda ser aplicado.
 - Borrado: lista de átomos que dejan de ser ciertos una vez se ha aplicado el operador.
 - Adición: lista de átomos que pasan a ser ciertos una vez se ha aplicado el operador.
 - Es usual unir la lista de borrado y la de adición en lo que se denomina una *lista de efectos* (en la que los elementos de la lista de borrado aparecen negados)

Operadores en el mundo de los bloques (I)

- Colocar un bloque sobre otro:

APILAR(x,y)

P: DESPEJADO(y), AGARRADO(x)

B: DESPEJADO(y), AGARRADO(x)

A: BRAZOLIBRE, SOBRE(x,y), DESPEJADO(x)

- Quitar un bloque que estaba sobre otro:

DESAPILAR(x,y)

P: SOBRE(x,y), DESPEJADO(x), BRAZOLIBRE

B: SOBRE(x,y), DESPEJADO(x), BRAZOLIBRE

A: AGARRADO(x), DESPEJADO(y)

Operadores en el mundo de los bloques (II)

- Agarrar un bloque con el robot:

AGARRAR(x)

P: DESPEJADO(x), SOBRELAMESA(x), BRAZOLIBRE

B: DESPEJADO(x), SOBRELAMESA(x), BRAZOLIBRE

A: AGARRADO(x)

- Bajar un bloque hasta la superficie:

BAJAR(x)

P: AGARRADO(x)

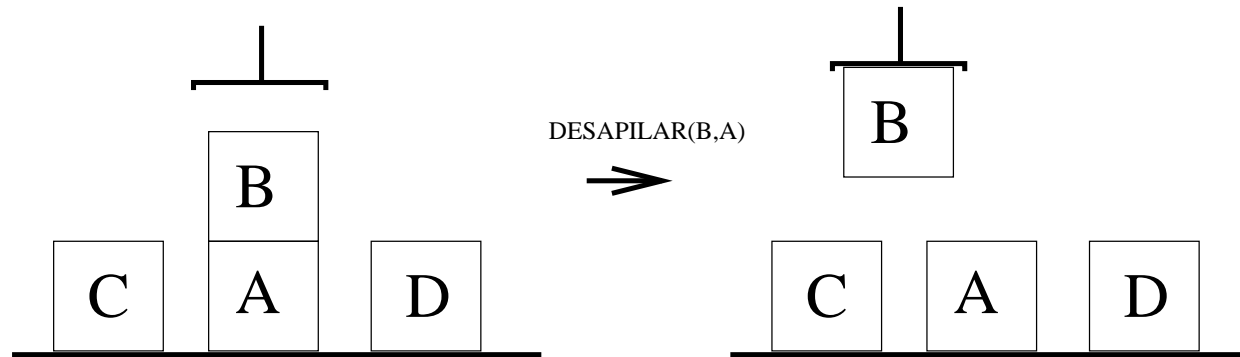
B: AGARRADO(x)

A: SOBRELAMESA(x), BRAZOLIBRE, DESPEJADO(x)

Aplicabilidad y aplicación de operadores

- Un operador es *aplicable* a un estado si éste satisface su precondition
 - Si aparecen variables en la precondition, la aplicabilidad se define *respecto de la sustitución* θ usada para satisfacer la precondition
 - Por abreviar, a veces la sustitución usada aparecerá implícita al hablar del operador
 - Por ejemplo, hablaremos de $\text{DESAPILAR}(B, A)$ para referirnos a $\text{DESAPILAR}(x, y)$ con la sustitución $\{x/A, y/B\}$
- El resultado de aplicar un operador aplicable (respecto de una sustitución θ) a un estado E es el estado resultante de:
 - Eliminar de E los átomos, instanciados por θ , de la lista de borrado (si estuvieran)
 - Añadir a E los átomos, instanciados por θ , de la lista de adición (si no estuvieran)
- **Plan:** secuencia de operadores aplicables, completamente instanciados
- **Solución:** plan que a partir del estado inicial obtiene un estado que satisface el objetivo

Ejemplo de aplicación de operador (I)



* Estado antes de aplicar DESAPILAR(B,A):

$E = \{\text{DESPEJADO}(B), \text{DESPEJADO}(C), \text{DESPEJADO}(D), \text{BRAZOLIBRE}, \text{SOBRE}(B,A), \text{SOBRELAMESA}(C), \text{SOBRELAMESA}(D), \text{SOBRELAMESA}(A)\}$

* Precondiciones de DESAPILAR(B,A):

$P = \{\text{SOBRE}(B,A), \text{DESPEJADO}(B), \text{BRAZOLIBRE}\}$

--- Condiciones satisfechas en el estado (operador aplicable) ----

* Listas de borrado y adición de DESAPILAR(B,A):

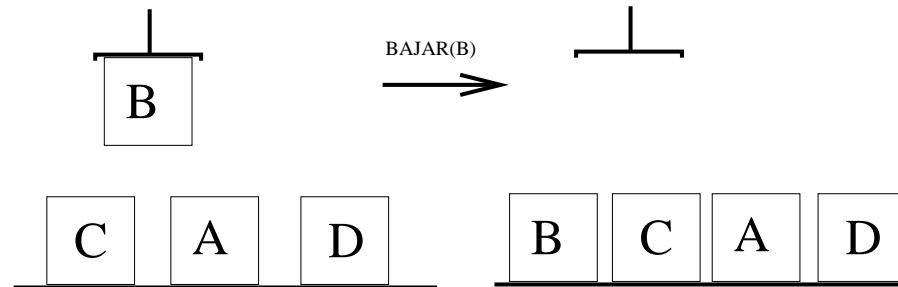
$B = \{\text{SOBRE}(B,A), \text{DESPEJADO}(B), \text{BRAZOLIBRE}\}$

$A = \{\text{AGARRADO}(B), \text{DESPEJADO}(A)\}$

* Estado después de aplicar DESAPILAR(B,A):

$E' = (E-B)+A = \{\text{DESPEJADO}(C), \text{DESPEJADO}(D), \text{SOBRELAMESA}(C), \text{SOBRELAMESA}(D), \text{SOBRELAMESA}(A), \text{AGARRADO}(B), \text{DESPEJADO}(A)\}$

Ejemplo de aplicación de operador (II)



* Estado antes de aplicar BAJAR(B):

$E = \{\text{DESPEJADO}(C), \text{DESPEJADO}(A), \text{DESPEJADO}(D), \text{SOBRELAMESA}(C),$
 $\text{SOBRELAMESA}(A), \text{SOBRELAMESA}(D), \text{AGARRADO}(B)\}$

* Precondiciones de BAJAR(B):

$P = \{\text{AGARRADO}(B)\}$

--- Condiciones satisfechas en el estado (operador aplicable) ----

* Lista de borrado de BAJAR(B):

$B = \{\text{AGARRADO}(B)\}$

* Lista de adición de DESAPILAR(B,A):

$A = \{\text{SOBRELAMESA}(B), \text{BRAZOLIBRE}, \text{DESPEJADO}(B)\}$

* Estado después de aplicar DESAPILAR(B,A):

$E' = (E-B)+A = \{\text{DESPEJADO}(C), \text{DESPEJADO}(A), \text{DESPEJADO}(D), \text{SOBRELAMESA}(C), \text{SOBRELAMESA}(A),$
 $\text{SOBRELAMESA}(D), \text{SOBRELAMESA}(B), \text{BRAZOLIBRE}, \text{DESPEJADO}(B)\}$

Ejemplo: cambio de rueda pinchada

- **Lenguaje:**

- **Objetos:** RUEDA-REPUESTO, RUEDA-PINCHADA, EJE, MALETERO, SUELO
- **Predicado:** EN(-,-)

- **Estado inicial:**

EN(RUEDA-PINCHADA ,EJE) ,EN(RUEDA-REPUESTO ,MALETERO)

- **Estado final:**

EN(RUEDA-REPUESTO ,EJE)

Acciones en el cambio de rueda pinchada

- Sacar la rueda de repuesto del maletero

QUITAR (RUEDA-REPUESTO, MALETERO)

P: EN (RUEDA-REPUESTO, MALETERO)

B: EN (RUEDA-REPUESTO, MALETERO)

A: EN (RUEDA-REPUESTO, SUELO)

- Quitar la rueda pinchada del eje

QUITAR (RUEDA-PINCHADA, EJE)

P: EN (RUEDA-PINCHADA, EJE)

B: EN (RUEDA-PINCHADA, EJE)

A: EN (RUEDA-PINCHADA, SUELO)

- Colocar la rueda de repuesto en el eje

PONER (RUEDA-REPUESTO, EJE)

P: NOT (EN (RUEDA-PINCHADA, EJE)), EN (RUEDA-REPUESTO, SUELO)

B: EN (RUEDA-REPUESTO, SUELO)

A: EN (RUEDA-REPUESTO, EJE)

- Dejar el coche solo hasta la mañana siguiente

DEJARSOLO

P: {}

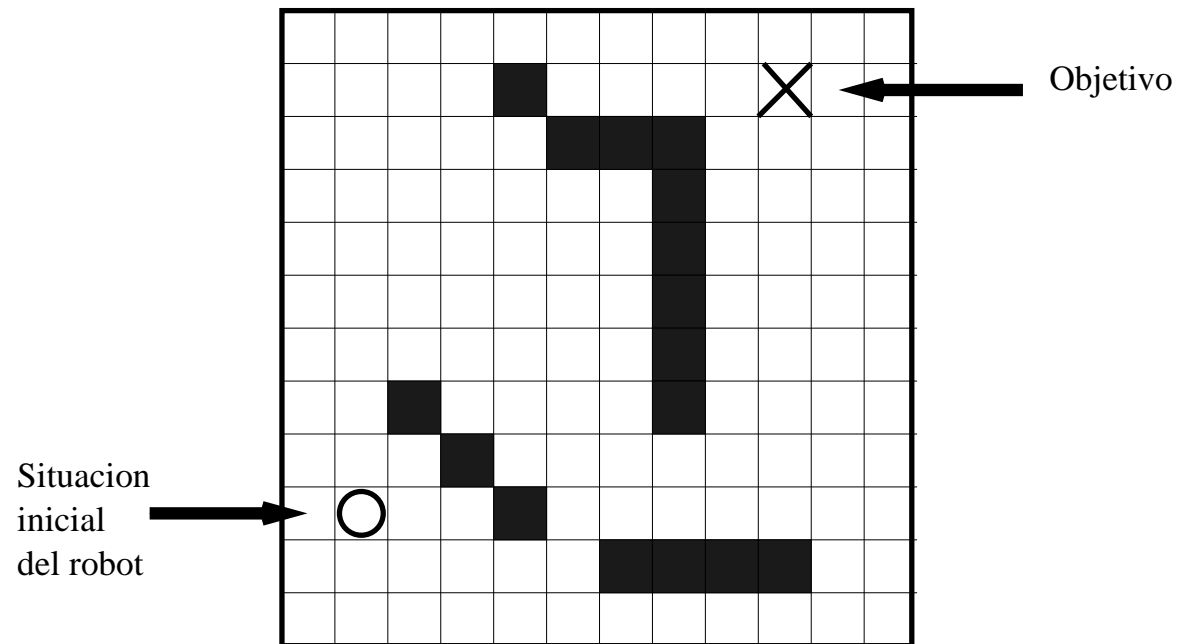
B: EN (RUEDA-REPUESTO, SUELO), EN (RUEDA-REPUESTO, EJE), EN (RUEDA-REPUESTO, MALETERO)

EN (RUEDA-PINCHADA, EJE), EN (RUEDA-PINCHADA, SUELO)

A: {}

Ejemplo: movimiento de robot por una rejilla

- Un robot ha de desplazarse por una rejilla, desde una posición inicial a una final
 - 8 movimientos posibles: N, S,E,O,NO,NE,SO,SE
 - En algunas de las rejillas existen obstáculos no franqueables



Representación del problema del movimiento de robot

- **Lenguaje:**
 - **Constantes:** números que indican coordenadas horizontales y verticales
 - **Predicados:** $\text{ROBOT-EN}(-,-)$ y $\text{LIBRE}(-,-)$
- **Estado inicial (casillas sin obstáculos y posición del robot):**
 $\text{LIBRE}(1,1), \dots, \text{LIBRE}(6,2), \text{LIBRE}(11,2), \dots, \text{LIBRE}(12,12), \text{ROBOT-EN}(2,3).$
- **Objetivo:** $\text{ROBOT-EN}(10,11)$
- **Acciones (sólo una, las siete restantes son análogos):**
 $\text{MOVER-SE}(x,y)$
 - P: $\text{ROBOT-EN}(x,y), \text{LIBRE}(x+1,y-1)$
 - B: $\text{ROBOT-EN}(x,y), \text{LIBRE}(x+1,y-1)$
 - A: $\text{ROBOT-EN}(x+1,y-1), \text{LIBRE}(x,y)$
- **En este caso, necesitamos símbolos de función (+ y -) y que el test de satisfacibilidad maneje las nociones de número siguiente y anterior**

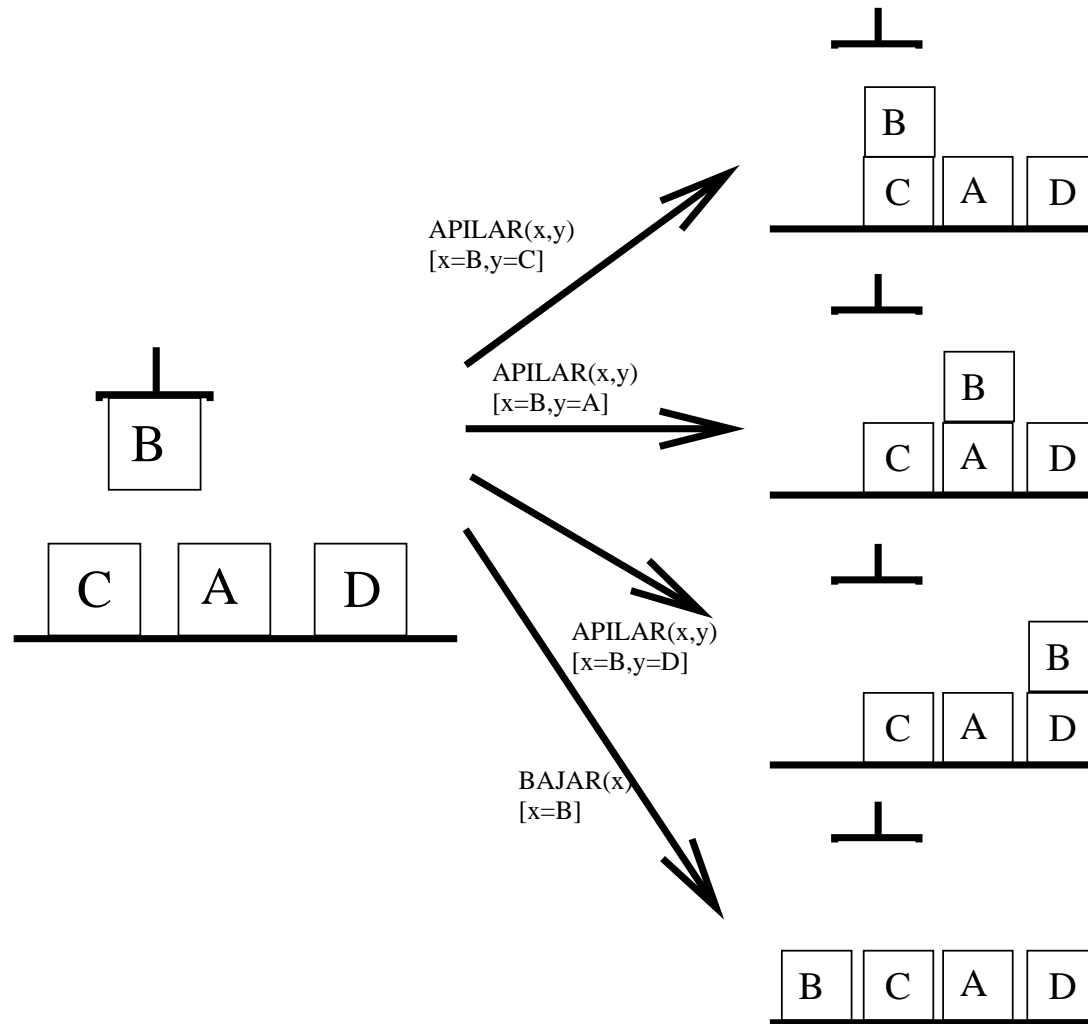
Extensiones al formalismo STRIPS

- Existen sistemas de planificación que usan un lenguaje de representación más expresivo
- Ejemplo:
 - Uso de símbolos de función
 - Manejo del símbolo de igualdad
 - Disyunciones
 - Variables con tipos
- En general existe un compromiso entre expresividad del lenguaje y simplicidad de los algoritmos que manejan la representación

Búsqueda de planes hacia adelante

- Un problema de planificación se puede plantear como un problema de espacio de estados:
 - Estados descritos mediante listas de literales.
 - Operadores como listas de precondiciones, adición y borrado.
 - Función es-estado-final descrita por un objetivo.
- La búsqueda de planes podría hacerse usando los algoritmos de búsqueda ya vistos en los temas anteriores: anchura, profundidad, primero el mejor, A^* ,...
- El uso de la representación STRIPS permite el uso de heurísticas *independientes del dominio*
 - Por ejemplo, el número de literales en el objetivo que quedan por satisfacer en un estado

Cálculo de sucesores



Cálculo de sucesores

- **Algoritmo para generar sucesores:**

FUNCION SUCESORES-ADELANTE(E)

1. Hacer SUCESORES igual a vacío
2. Para cada operador O
 - 2.1 Para cada sustitución THETA de las variables que aparecen en O tal que THETA(PRECONDICIONES(O)) está incluida en E, hacer:
 - 2.1.2 E' igual a E - THETA(BORRADO(O)) + THETA(ADICION(O))
 - 2.1.3 Añadir E' a SUCESORES
3. Devolver SUCESORES

- **Ejemplo:**

$E = \{\text{DESPEJADO}(C), \text{DESPEJADO}(A), \text{DESPEJADO}(D), \text{SOBRELAMESA}(C), \text{SOBRELAMESA}(A), \text{SOBRELAMESA}(D), \text{AGARRADO}(B)\}$

Cuatro posibles operadores aplicables:

- 1), 2) y 3): $O = \text{APILAR}(x,y)$ con $\text{THETA} = [x=B, y=C]$, $\text{THETA} = [x=B, y=A]$ y $\text{THETA} = [x=B, y=D]$, resp.
- 4): $O = \text{DEJAR}(x)$ con sustitución $\text{THETA} = [x=B]$.

Sucesor en el caso 1):

$E' = \{\text{DESPEJADO}(A), \text{DESPEJADO}(D), \text{SOBRELAMESA}(C), \text{SOBRELAMESA}(A), \text{SOBRELAMESA}(D), \text{BRAZOLIBRE}, \text{DESPEJADO}(B), \text{SOBRE}(B,C)\}$

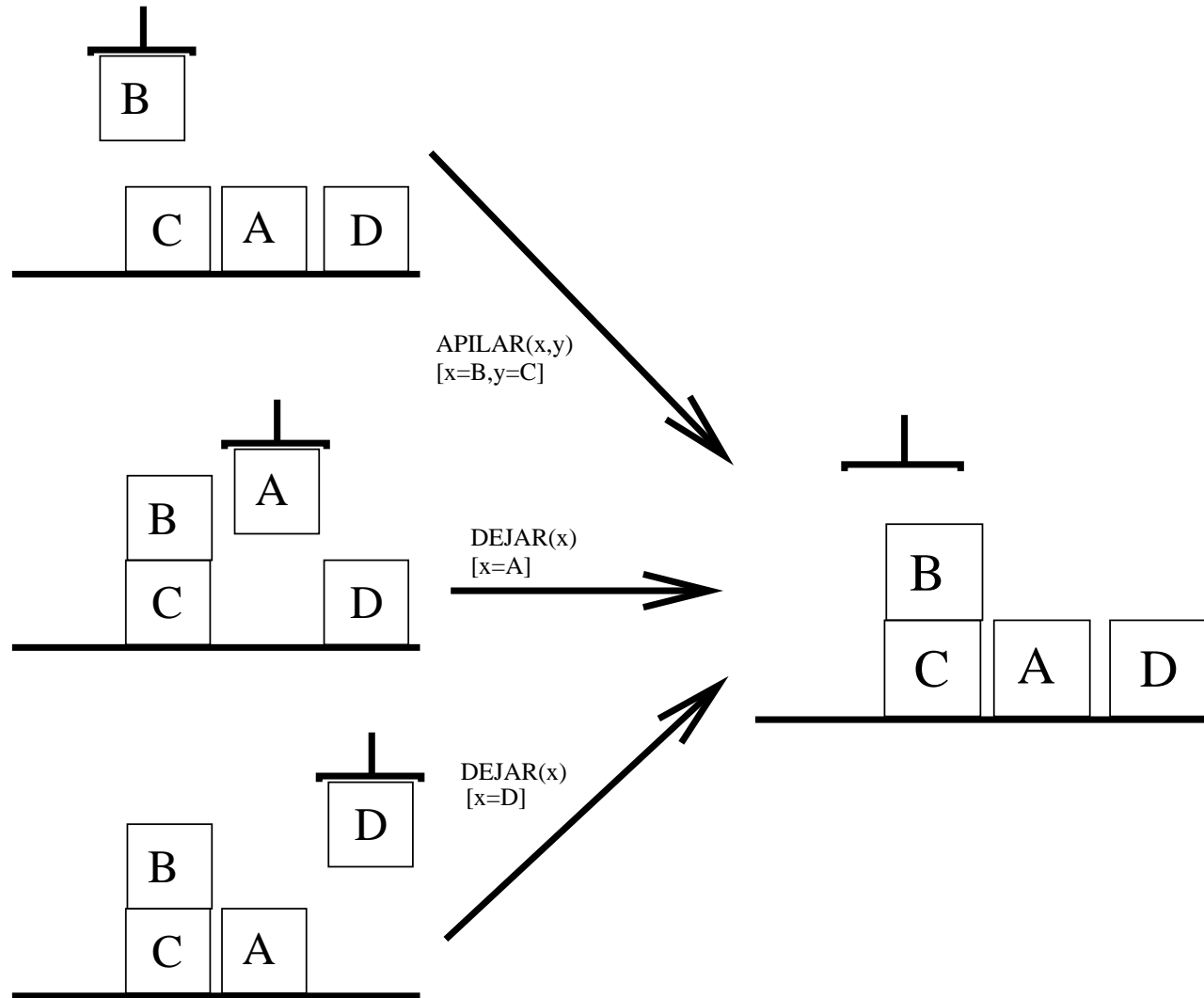
Ineficiencia de la búsqueda hacia adelante

- La búsqueda hacia adelante en un problema de planificación:
 - Es completa (si el conjunto de objetos es finitos)
 - Es ineficiente en la práctica (por las razones ya expuestas)
- El formalismo lógico para representar los estados, resuelve en cierta medida:
 - El problema del marco
 - La obtención de heurísticas independientes del dominio
- Sin embargo, persisten algunos problemas
 - El gran factor de ramificación (consideración de acciones irrelevantes)
 - La búsqueda del plan de manera secuencial
 - Planes no jerárquicos
- Para intentar resolver el primer punto, se puede realizar una *búsqueda hacia atrás*, dirigida por el objetivo

Búsqueda de planes hacia atrás

- Búsqueda hacia atrás:
 - Comienzo en un objetivo
 - En cada estado se generan todos los posibles predecesores
 - Finalizar cuando se alcanza un objetivo que es cierto en el estado inicial
- Predecesor de un *objetivo* G :
 - mediante operador O y sustitución θ
 - es cualquier estado E cuyo sucesor al aplicar O y θ satisface G .
 - Relevante: Al menos, en la lista de adición del operador está uno de los literales de G
 - Consistente: En la lista de borrado de G no aparece ningún literal de G
- Ventajas:
 - Generalmente más eficiente: menos ramificación
 - Posibilidad de aplicar heurísticas independientes del dominio (proximidad al estado inicial)

Cálculo de predecesores



Cálculo de predecesores

- **Algoritmo para generar predecesores:**

FUNCION PREDECESORES-ATRAS(G)

1. Hacer PREDECESORES igual a vacío

2. Para cada operador O

2.1 Para cada sustitución THETA de las variables que aparecen en O tal que al menos un literal L de G está en THETA(ADICION(O)), y G no tiene elementos de THETA(BORRADO(O)), hacer:

2.1.1 E igual G + THETA(PRECONDICION(O)) - THETA(ADICION(O))

2.1.2 Añadir E a PREDECESORES

3. Devolver PREDECESORES

- **Ejemplo:**

G = {DESPEJADO(B), DESPEJADO(A), DESPEJADO(D), SOBRE(B,C), BRAZOLIBRE, SOBRELAMESA(C), SOBRELAMESA(A), SOBRELAMESA(D)}

Tres posibles operadores aplicables (en regresión)

1): O=APILAR(x,y) con sustitución THETA=[x=B,y=C].

2) y 3): O=DEJAR(x) con sustituciones THETA=[x=A], s=[x=D] resp.

Predecesor en el caso 1)

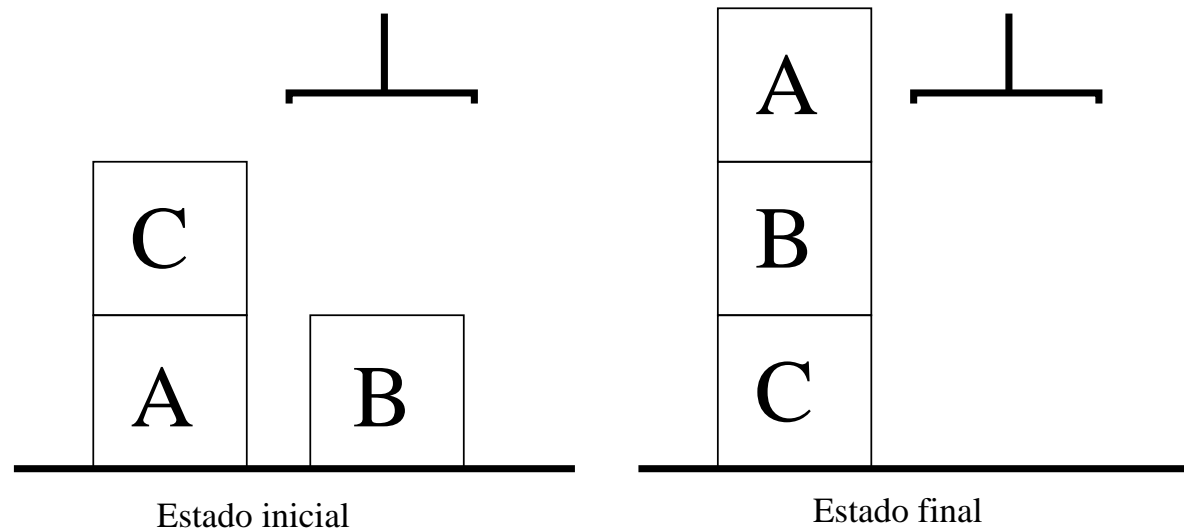
E = {DESPEJADO(B), DESPEJADO(A), DESPEJADO(D), SOBRELAMESA(C), SOBRELAMESA(A), SOBRELAMESA(D), AGARRADO(B)}

Planificación lineal

- Tanto la búsqueda de planes hacia adelante como la búsqueda de planes hacia atrás son ejemplos de algoritmos de *planificación lineal*
- En un planificador lineal, el espacio de búsqueda lo forman secuencias *totalmente ordenadas* de acciones, bien desde el inicio hacia adelante o bien desde el objetivo hacia atrás

La anomalía de Sussman

- Problema de planificación que muestra las carencias de la planificación lineal:

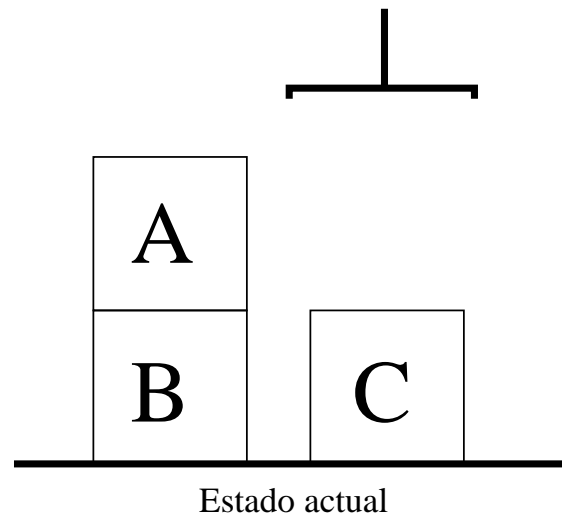


- Estados:

Estado inicial = { SOBRELAMESA(A), SOBRELAMESA(B),
DESPEJADO(B), DESPEJADO(C), BRAZOLIBRE, SOBRE(C,A) }
Objetivo = {SOBRE(A,B), SOBRE(B,C) }

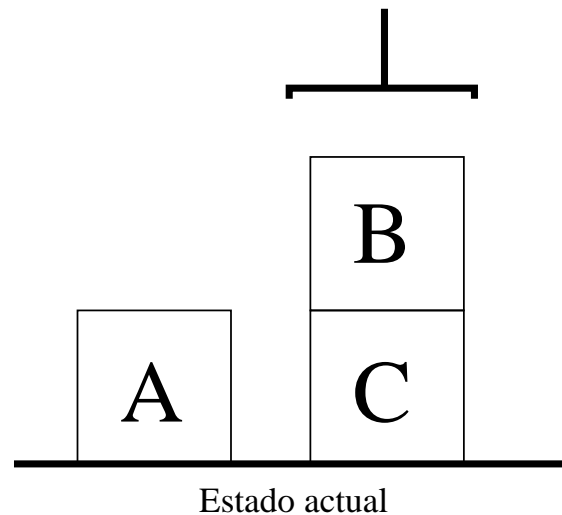
Anomalía de Sussman con un planificador lineal

- Un plan para satisfacer SOBRE(A,B):
 1. DESAPILAR(C,A)
 2. BAJAR(C)
 3. AGARRAR(A)
 4. APILAR(A,B)
- Estado intermedio:



Anomalía de Sussman con un planificador lineal

- Un plan para satisfacer $\text{SOBRE}(B,C)$:
5. $\text{DESAPILAR}(A,B)$ 6. $\text{BAJAR}(A)$ 7. $\text{AGARRAR}(B)$ 8. $\text{APILAR}(B,C)$
- Estado intermedio:



- Problema: $\text{SOBRE}(A,B)$ ha dejado de ser cierto.
 - Al satisfacer $\text{SOBRE}(B,C)$ se ha deshecho.

Deficiencias de la planificación lineal

- Causa del problema:
 - El planificador intenta resolver un objetivo cada vez.
 - En ciertos casos, hay interacción entre objetivos, deshaciéndose unos a otros.
- Este un problema general de los algoritmos de planificación lineal.

Alternativas a la planificación lineal

- Búsqueda de planes no lineales, entrelazando objetivos.

- Ejemplo:

La anomalía de Sussman se puede solucionar:

1. Comenzado a satisfacer SOBRE(A,B), en parte sólo, despejando A, (DESAPILAR(C,A), BAJAR(C)).
2. Satisfacer SOBRE(B,C), haciendo APILAR(B,C).
3. Completar SOBRE(A,B), haciendo APILAR(A,B)

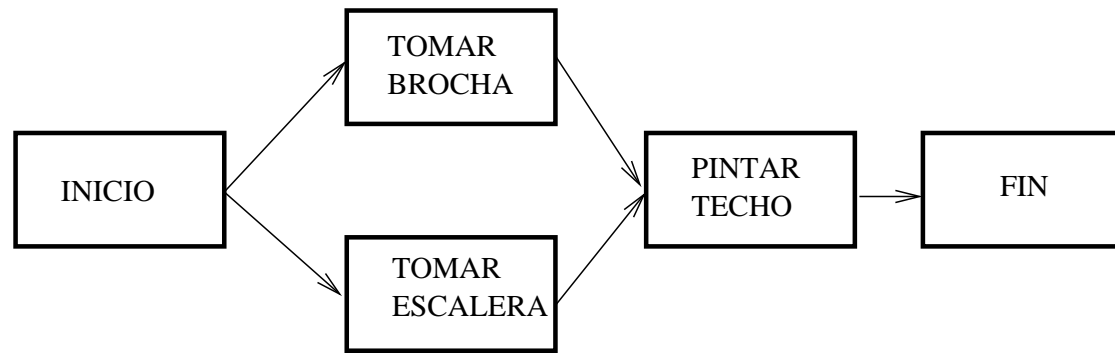
- *Principio de mínimo compromiso*: no determinar completamente el orden entre acciones, a no ser que sea estrictamente necesario

Planes parcialmente ordenados

- Plan parcialmente ordenado: un plan en el que sólo se especifican algunas de las precedencias entre sus acciones
 - Secuenciar un plan parcialmente ordenado: construir un plan secuencial que respete las precedencias obligatorias
- Idea:
 - sustituir la búsqueda en el espacio de situaciones por la búsqueda en el espacio de planes parcialmente ordenados
 - aplicar operadores que refinan los planes parciales a medida que se detecta su necesidad

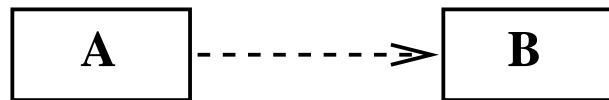
Planes parcialmente ordenados: ejemplo

- Plan parcialmente ordenado y secuenciaciones



Planes parcialmente ordenados: componentes

- Un conjunto de *acciones* que constituyen los pasos que el plan lleva a cabo, de entre los operadores del problema, cada una con sus precondiciones y efectos
 - Dos acciones especiales: INICIO (sin precondiciones y cuyo efecto es el estado inicial) y FIN (sin efectos y cuyas precondiciones son el objetivo final)
- Un conjunto de *restricciones de orden* $A \prec B$ entre acciones del plan



- Un conjunto de enlaces causales, $A \xrightarrow{p} B$, especificando la consecución, por parte de una acción, de una de las precondiciones de otra acción del plan



- Un conjunto de *precondiciones abiertas*: aquellas que no se alcanzan mediante ninguna acción del plan

Planificación de orden parcial como búsqueda

- Búsqueda en el espacio de los planes parciales
 - Importante: *los estados son los planes parciales*, en lugar de las distintas situaciones del mundo
- Estado inicial: plan cuyas únicas acciones son INICIO y FIN, con la restricción $\text{INICIO} \prec \text{FIN}$, sin enlaces causales y con todas las precondiciones de FIN abiertas
- Estados finales (o *soluciones*): planes parciales sin conflictos entre los enlaces causales, sin ciclos entre las restricciones de orden y sin precondiciones abiertas
 - Una acción C entra en *conflicto con* (o *amenaza*) un enlace causal $A \xrightarrow{p} B$, si C tiene a $\neg p$ en su lista de efectos y según las restricciones de orden, C *podría ir* después de A y antes que B
 - Nota: cualquier secuenciación de un plan parcial solución da lugar a una secuencia de acciones que partiendo del estado inicial consigue el objetivo

Planificación de orden parcial como búsqueda: operadores

- Dada una precondition abierta p de una acción B del plan, por cada acción A que tiene a p como efecto, se puede obtener un plan sucesor (*refinar*) aplicando los dos siguientes pasos (siempre que el plan resultante sea consistente):
- Paso 1: resolución de la precondition abierta, dos posibilidades
 - Establecimiento simple: si la acción A ya aparece en el plan, añadir la restricción $A \prec B$ y el enlace causal $A \xrightarrow{p} B$
 - Acción nueva: si la acción A no aparece en el plan, añadir la acción al plan, las restricciones $A \prec B$, INICIO $\prec A$ y $A \prec$ FIN, y el enlace causal $A \xrightarrow{p} B$
- Paso 2: resolución de conflictos entre el nuevo enlace causal y las acciones del plan, y, si la acción A es nueva, entre A y los los enlaces causales del plan
 - Un conflicto entre $A \xrightarrow{p} B$ y C se resuelve añadiendo la restricción $B \prec C$ (promoción) o la restricción $C \prec A$ (degradación)

Planificación de orden parcial: búsqueda

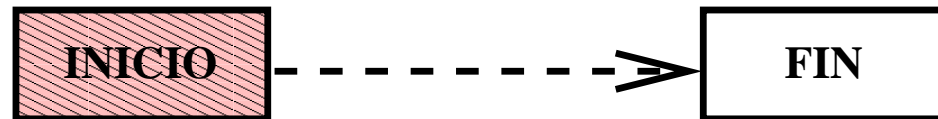
- Es posible plantear un algoritmo de planificación como una búsqueda en el espacio de estados anterior
 - Es decir, como búsqueda de una secuencia de operadores de refinamiento de planes que a partir del plan inicial obtiene un plan parcial sin conflictos entre los enlaces causales, sin ciclos entre las restricciones de orden y sin precondiciones abiertas
- Motivos para la ramificación en la búsqueda:
 - Distintas acciones que resuelven precondiciones abiertas
 - Las dos formas de resolver un conflicto (promoción y degradación)
 - Nota: no es necesario considerar un sucesor por cada precondición abierta del plan, basta ir considerarlas en pasos posteriores, siguiendo un orden dado

Anomalía de Sussman en POP

- En lo que sigue, veremos cómo la planificación de orden parcial resuelve la anomalía de Sussman
 - Nota importante: en cada momento escogeremos los operadores de refinamiento adecuados, pero en la práctica estos operadores se encuentran mediante un proceso de búsqueda entre todas las posibilidades
- Convenios en la representación gráfica del ejemplo, por claridad en la misma:
 - Algunas restricciones de orden no aparecerán, en particular la asociadas a enlaces causales
 - Las precondiciones y efectos no aparecerán. En particular, no aparecerán las precondiciones abiertas
 - Cuando una acción no tenga precondiciones abiertas, ésta aparecerá sombreada

Anomalía de Sussman en POP

- Plan inicial:

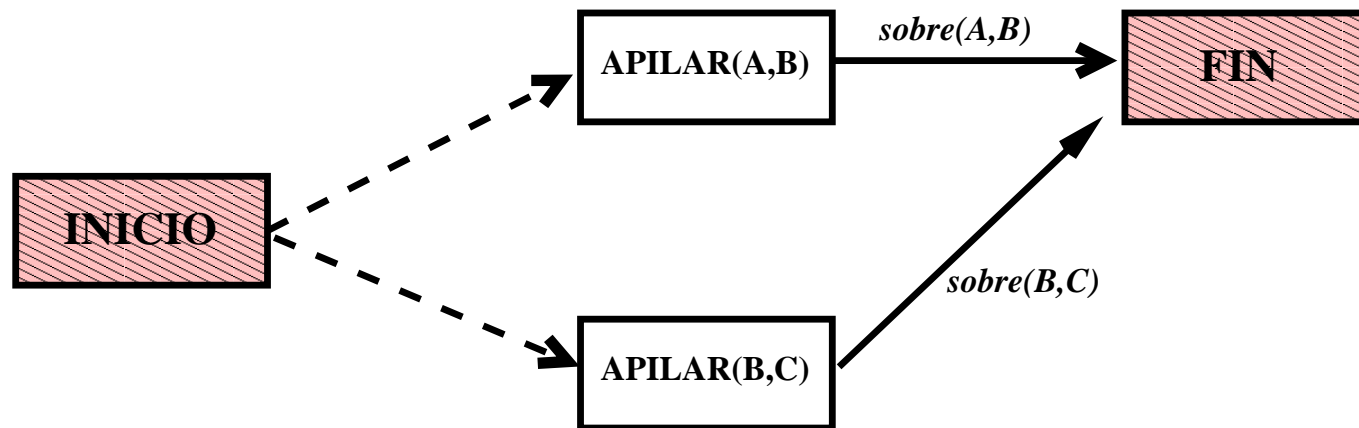


- Resolver la precondición $sobre(A,B)$ de FIN mediante la acción nueva APILAR(A,B) (es la única opción):



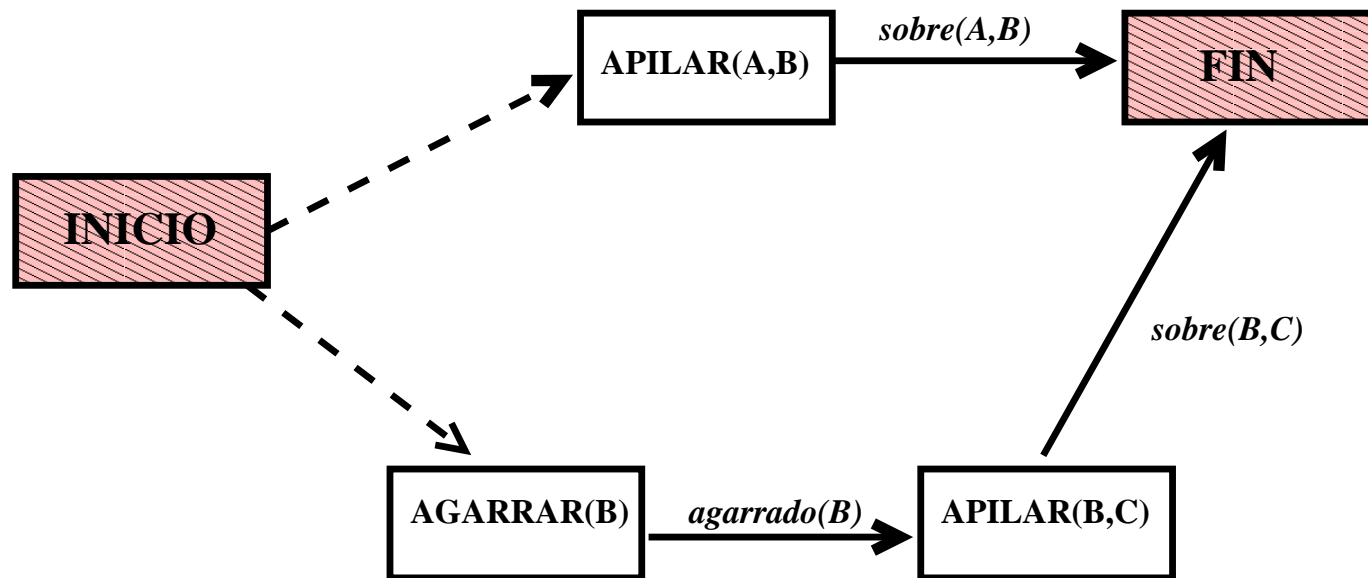
Anomalía de Sussman en POP

- Resolver la precondición $sobre(B,C)$ de FIN mediante la acción nueva APILAR(B,C) (es la única opción):



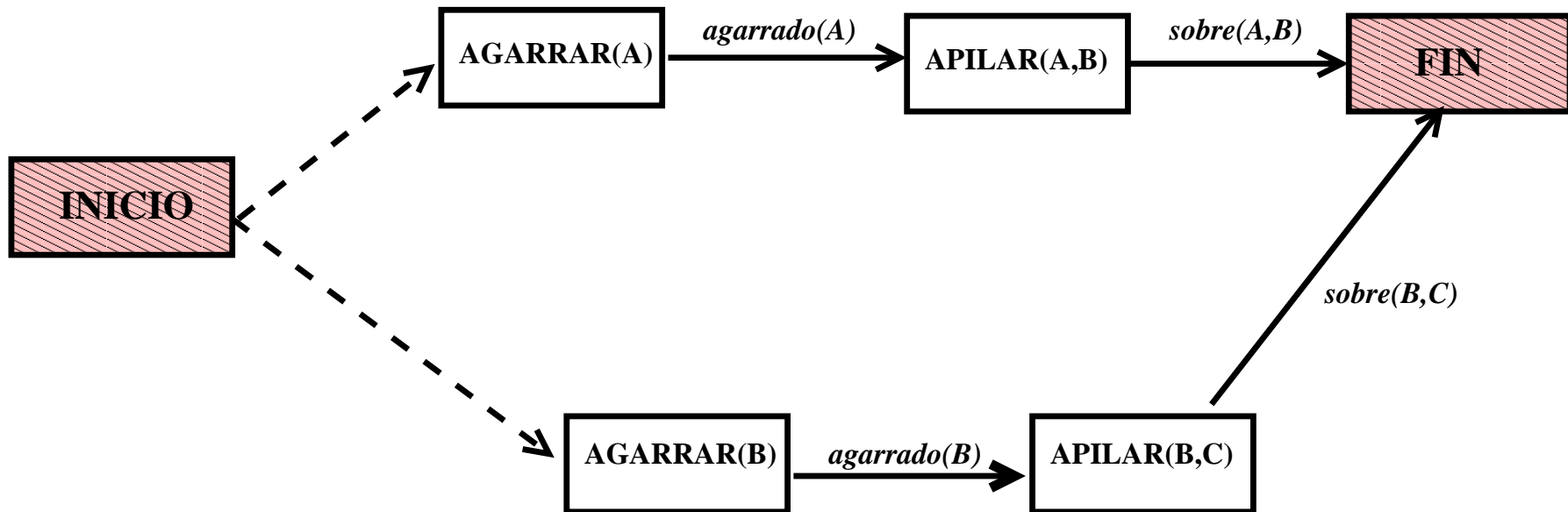
Anomalia de Sussman en POP

- Resolver la precondition $agarrado(B)$ de $APILAR(B,C)$ mediante la acción nueva $AGARRAR(B)$
 - También se podría usar $DESAPILAR$, opción que habría que considerar si la elección realizada fallara (búsqueda)



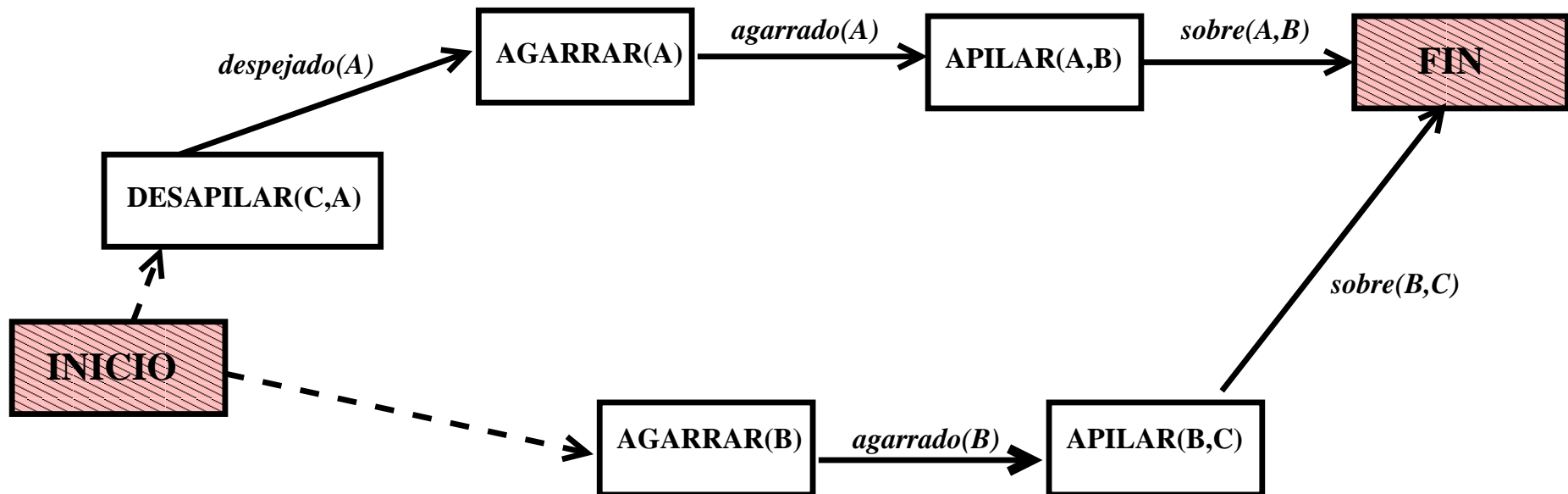
Anomalia de Sussman en POP

- Resolver la precondición $agarrado(A)$ de $APILAR(A,B)$ mediante la acción nueva $AGARRAR(A)$
 - Como en el caso anterior, también se podría usar $DESAPILAR$ (explorar mediante búsqueda)



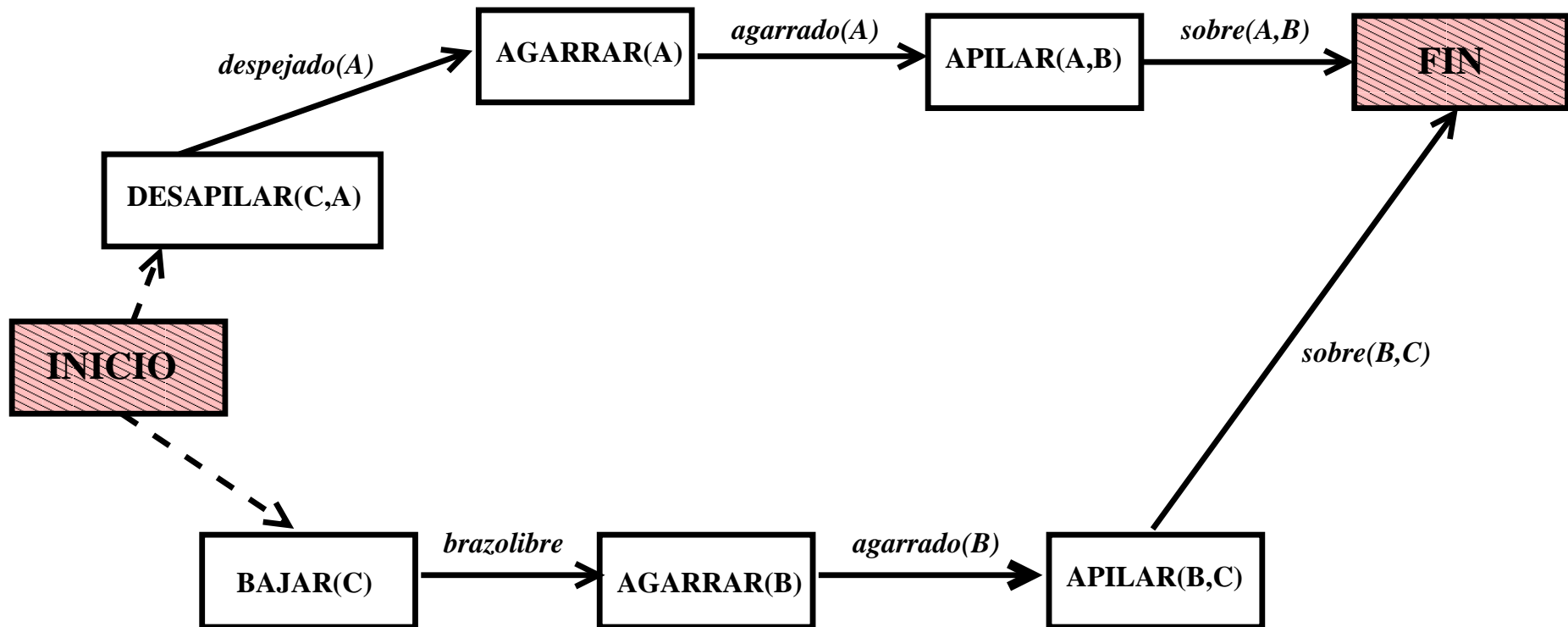
Anomalia de Sussman en POP

- Resolver la precondition $despejado(A)$ de $AGARRAR(A)$ mediante la acción nueva $desapilar(C,A)$:



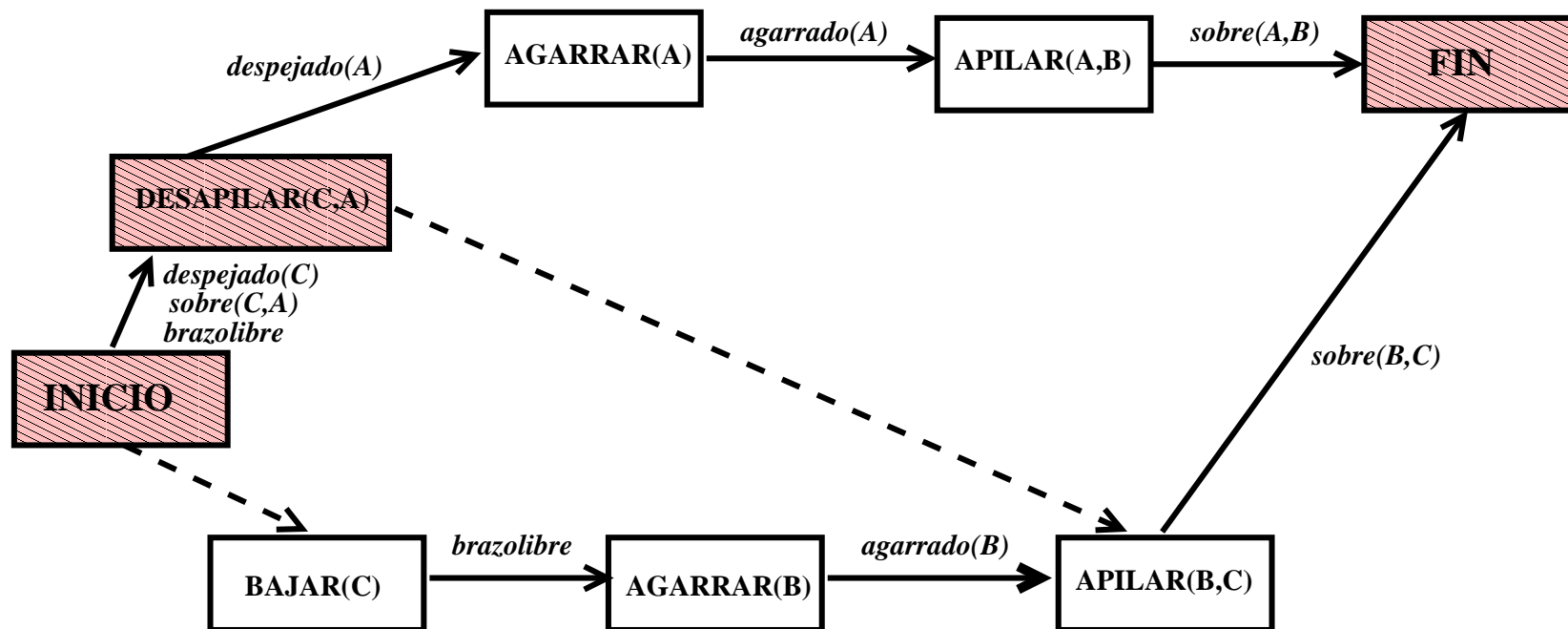
Anomalia de Sussman en POP

- Resolver la precondición *brazolibre* de AGARRAR(B) mediante la acción nueva bajar(C) (existen más opciones):



Anomalia de Sussman en POP

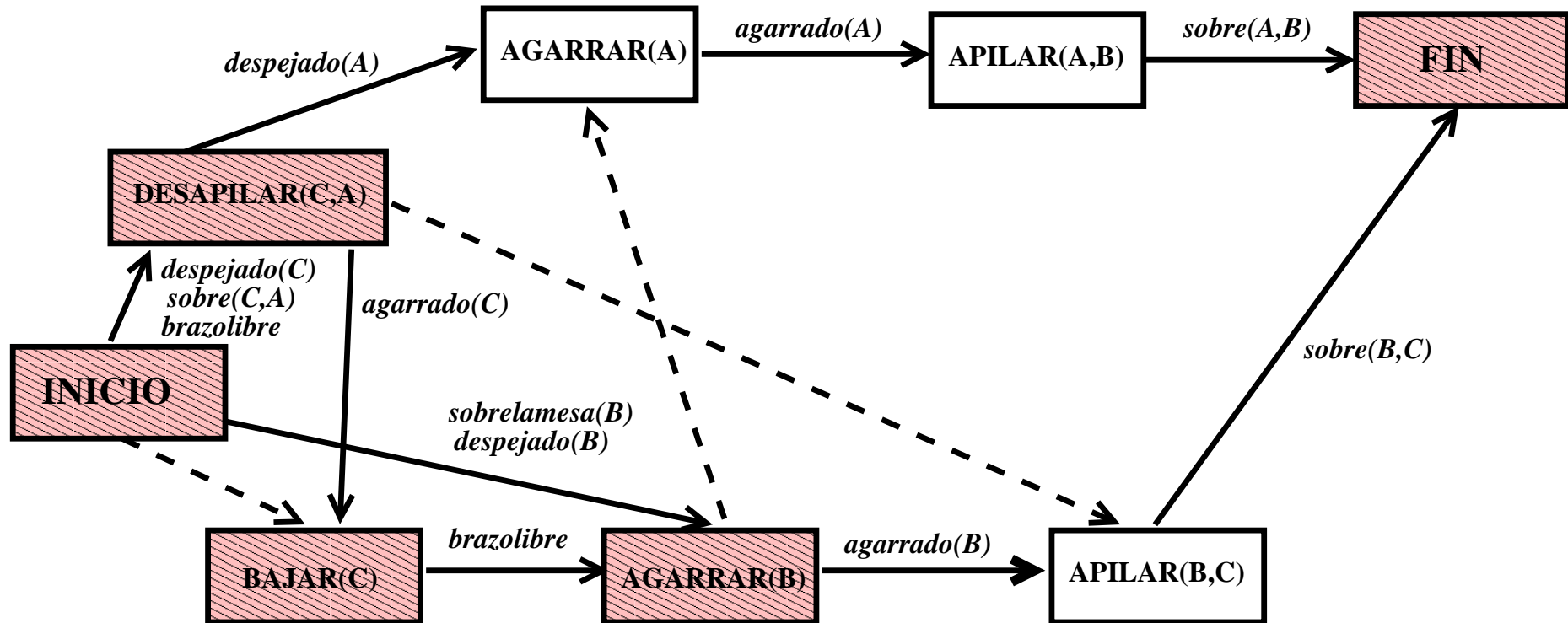
- Usar la acción INICIO para, mediante establecimiento simple, resolver todas las precondiciones de DESAPILAR(C,A) (existen otras opciones):
 - El enlace causal entre INICIO y DESAPILAR(C,A) que asegura *despejado(C)*, se ve amenazado por APILAR(B,C): resolver mediante promoción (única opción), colocando la restricción de que APILAR(B,C) debe ir tras DESAPILAR(C,A)



Anomalía de Sussman en POP

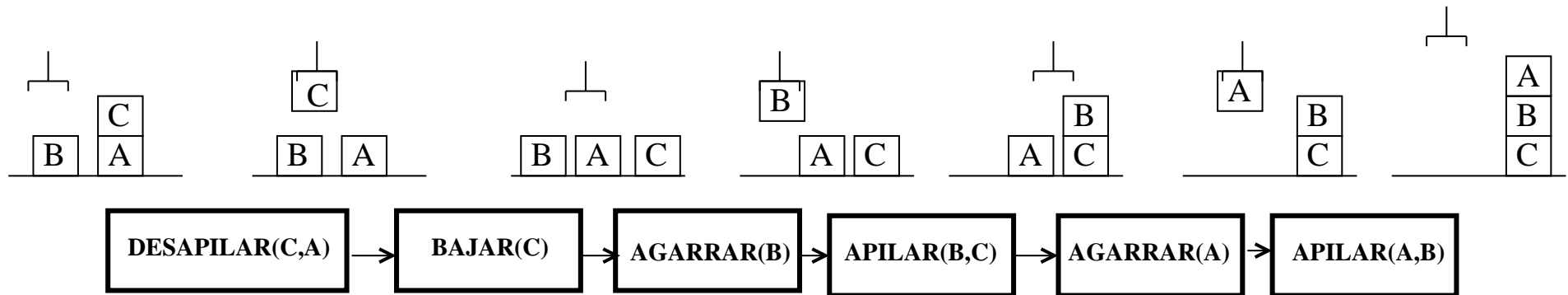
- Usar la acción INICIO para, mediante establecimiento simple, resolver las restantes precondiciones de AGARRAR(B); igualmente mediante establecimiento simple, usar DESAPILAR(C,A) para resolver la precondición *agarrado(C)* de BAJAR(C)
 - El enlace causal entre BAJAR(C) y AGARRAR(B) que asegura *brazolibre*, se ve amenazado por AGARRAR(A): resolver mediante promoción, colocando la restricción de que AGARRAR(A) debe ir tras AGARRAR(B)
 - De esta manera, también se resuelve la amenaza de APILAR(A,B) al enlace causal entre INICIO y AGARRAR(B), ya que la anterior restricción implica que APILAR(A,B) ocurre después de AGARRAR(B)

Anomalía de Sussman en POP



Anomalía de Sussman en POP

- El plan parcial resultante es una solución: no tiene conflictos, no tiene ciclos y no hay precondiciones abiertas
- Secuenciación del plan:



- Nótese que se intercalan los pasos necesarios para solucionar los dos objetivos originales

Algoritmo POP (pseudocódigo)

FUNCION POP(ESTADO-INICIAL,OBJETIVOS)

1. Hacer PLAN-ACTUAL igual PLAN-INICIAL(ESTADO-INICIAL,OBJETIVOS)
2. Repetir hasta que PLAN-ACTUAL sea una solución:
 - 2.1 Sea S un paso de PLAN-ACTUAL y C una de sus precondiciones abiertas
 - 2.2 *Escoger*, mediante establecimiento simple o como acción nueva, un paso T que tenga a C entre sus efectos;
si no existe tal paso o ya se han intentado todas las alternativas, devolver FALLO
 - 2.3 Modificar PLAN-ACTUAL, añadiendo el enlace causal $T \text{ --c--> } S$ y la restricción temporal $T < S$
 - 2.4 Si T es nuevo, modificar también PLAN-ACTUAL para añadir T como nuevo paso y las restricciones temporales $INICIO < T$ y $T < FIN$
 - 2.5 Para cada paso U de PLAN-ACTUAL que amenace a $T \text{ --c--> } S$, *escoger* una de las dos siguientes maneras de resolver la amenaza, siempre que el resultado sea consistente:
añadir a PLAN-ACTUAL la restricción $U < T$ (degradación)
o añadir la restricción $S < U$ (promoción);
si ya se han intentado ambas opciones, devolver FALLO
 - 2.6 Si T es nuevo, para cada enlace causal $Si \text{ --c--> } Sj$ de PLAN-ACTUAL que se vea amenazado por T,
escoger una de las dos siguientes maneras de resolver la amenaza, siempre que el resultado sea consistente:
añadir a PLAN-ACTUAL la restricción $T < Si$ (degradación)
o añadir la restricción $Sj < T$ (promoción);
si ya se han intentado ambas opciones, devolver FALLO
3. Devolver PLAN-ACTUAL

POP como búsqueda

- Cada una de las elecciones se puede reconsiderar, en caso de fallo
 - El pseudocódigo anterior es una versión *no determinista* del algoritmo
- Arbol de búsqueda:
 - Nodos: planes parciales
 - Ramificación por la elección de acciones que resuelven precondiciones abiertas
 - Ramificación por las distintas maneras de resolver las amenazas
- La elección de la precondición a resolver no es una elección a reconsiderar, ya que finalmente deberán estar todas resueltas
 - Aunque el orden en el que se vayan considerando puede influir en la eficiencia de la búsqueda

POP: algunas consideraciones

- Por simplificar, no hemos tenido en cuenta las complicaciones que podrían surgir con las variables de los operadores:
 - Uso de unificación
 - Algunas variables podrían quedar sin instanciar (principio de mínimo compromiso)
 - Restricciones de desigualdad
- Uso de heurísticas en POP
 - Difícil estimar la “cercanía” de un plan parcial a una solución
- Heurística para seleccionar la precondition a resolver: heurística más restrictiva
 - Seleccionar la condición que puede ser resuelta por el *menor número* de acciones

Ampliación del tema

- Existen muchas otras técnicas de planificación:
 - GRAPHPLAN
 - SATPLAN
- Otras cuestiones que aquí no vemos:
 - Planificación con recursos limitados
 - Planificación mediante descomposición jerárquica
 - Planificación en entornos con incertidumbre
 - Ejecución del plan: vigilancia y replanificación
 - Planificación continua
 - Planificación multiagente

Bibliografía

- Russell, S. y Norvig, P. *Artificial Intelligence (A Modern Approach)* (Prentice–Hall, 2000). Second Edition
 - Cap. 11: “Planning”.
- Borrajo, D., Juristo, N., Martínez, V. y Pazos, J. *Inteligencia Artificial: métodos y técnicas*.1993.
- Rich, E. y Knight, K. *Inteligencia Artificial (segunda edición)*.1996.
- Nilsson, N. *Principios de Inteligencia Artificial*. 1987.
- Escolano, F. et al. *Inteligencia Artificial: modelos, técnicas y áreas de aplicación* (Ed. Thomson, 2000)