Planificación

- **Planificación con búsqueda en el espacio de estados (State-space planning)**
  - Algoritmos heurísticos como A* mejoran la eficiencia (IDA ,..)
  - Hemos visto representación de estados y operadores sin una sintaxis fija
  - Planificación como búsqueda "pura" y heurísticas **dependientes del dominio** (A*) tiene limitaciones
    - Necesita trabajar con un conjunto finito de estados
    - Estados con mucha información → problema del marco
    - Sistemas completamente observables → no trabaja en sistemas de informacion parcial
    - Dinamismo, tiempo real, replanificación (eficiente)
    - Sistemas deterministas.
- Otras opciones de algoritmos de planificación → representación del conocimiento
  - **Planificación en el espacio de planes (Plan-space planning ó PSP)**
  - Planificación jerárquica (Hierarchical Task Network Planning ó HTN)
  - Planificación basada en casos
  - Algoritmos específicos de planificación (también basados en búsqueda)
    - POP, GRAPHPLAN --> Se analizan los enlaces y las dependencias de orden entre acciones y literales / priorizar acciones y recursos críticos
  - Planificación dinámica: supervisión, revisión y replanificación a partir del estado actual.
  - Planificación estocástica: incertidumbre en las acciones

Planificación

- Hay dominios donde los estados tienen muchísima información y las acciones cambian muy poco del estado → videojuegos / *problema del marco*
  - Factor de ramificación enorme y muchas acciones irrelevantes
  - Puede haber múltiples objetivos (dependientes o independientes entre sí)
  - Puede haber (o no) dependencia entre acciones → orden parcial
  - La necesidad de una acción puede detectarse sin necesidad de que se hayan decidido las acciones previas (compromiso mínimo)
  - Estados y acciones con estructura (representación en lógica)
  - No sólo secuencias de acciones / orden parcial / recursos compartidos
- **Lenguaje basado en lógica** para describir estados/operadores/dominios
  - PDDL = Planning Domain Definition Language
  - Algoritmos de planning combinan **búsqueda y lógica**
    - PSP (plan space planning) **búsqueda en el espacio de planes parciales**
    - GRAPHPLAN
      - búsqueda heurística similar a A* con **heurísticas independientes del dominio** usando propiedades del lenguaje de representación (grafo de planificación)

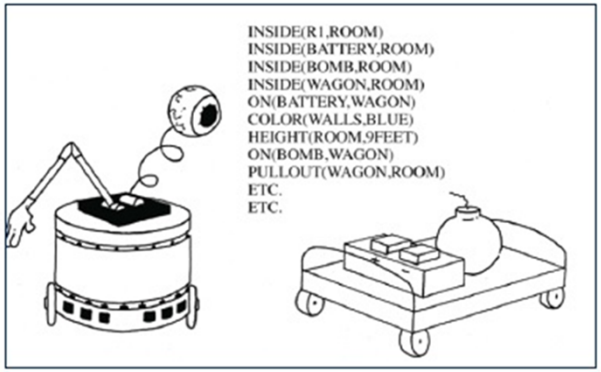


FIGURA 1. EL PROBLEMA DE MARCO SEGÚN DENNETT

Acción (volar(p,desde,hasta),
PRECOND:En(p,desde) ^ avión(p) ^
aeropuerto(desde) ^ aeropuerto(hasta)
EFECTO: ¬ En(p,desde) ^ En(p,hasta)

Problema del marco

- Estados con mucha información: sólo se representan predicados con los hechos ciertos
  - Decidir cómo representar qué parte cambia y qué parte permanece cuando ejecutamos una acción
- **Hipótesis del mundo cerrado**: todo lo que no se sabe es falso
  - permite a un agente asumir de forma “segura” que un hecho es falso si no puede inferir que es verdadero.
  - asume que toda la información positiva relevante ha sido especificada; cualquier otro hecho no especificado se asume por defecto que es falso
- **Hipótesis del mundo abierto**: lo que no se sabe, no se sabe (puede ser cierto o falso, no hago inferencias)

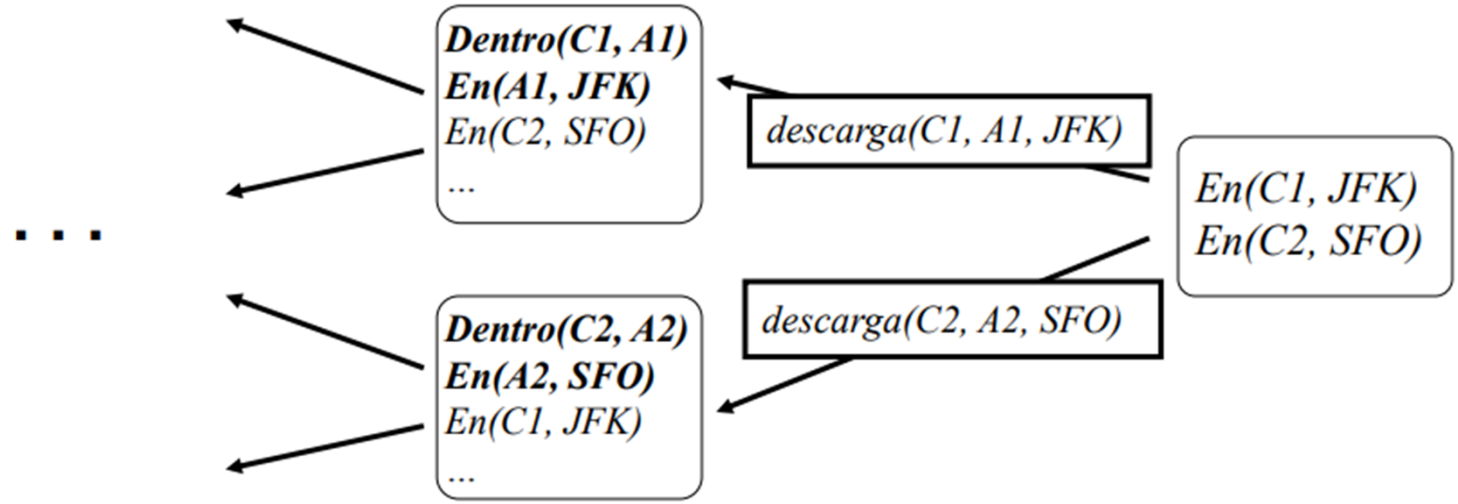
La competición internacional de planificación

- Una forma de hacer que la tecnología de planificación avance más rápido
- Primera edición en 1998
- Creadora del lenguaje PDDL

- Efectivamente ha hecho evolucionar el área de investigación
  - Explosión de nuevos métodos
  - Hibridación de métodos
  - Optimización de métodos
- Todos los resultados están disponibles en la web
  `http://ipc.icaps-conference.org/`
- El código de muchos de los planificadores esta disponible

Planificación

- Búsqueda hacia atrás (empezando por el objetivo) → **menor factor de ramificación**

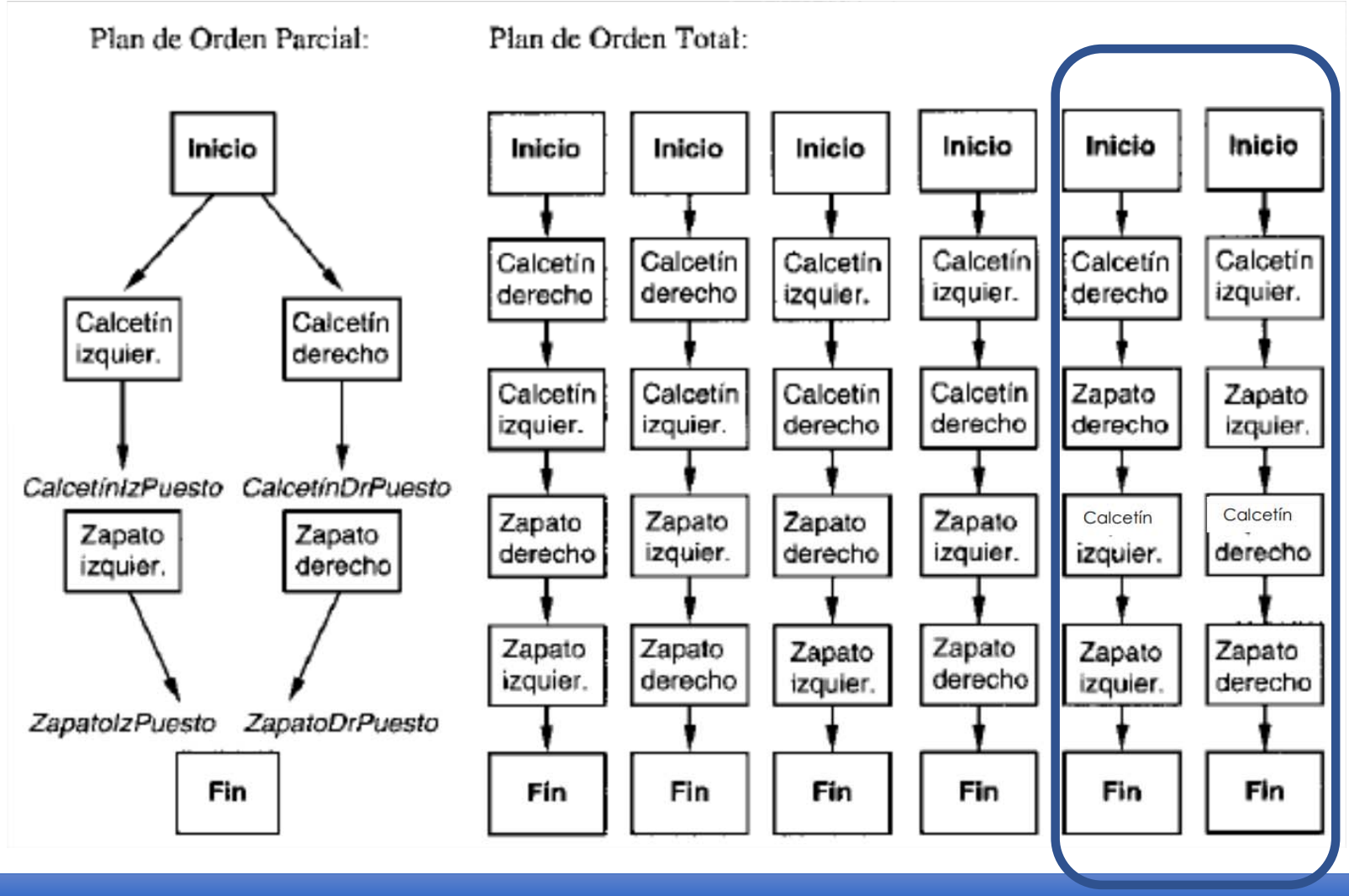


- Suponer independencia de objetivos y resolverlos por separado
  - Dividir el problema en subproblemas que se resuelven independientemente y combinamos sus soluciones. Sobre(A,B), sobre(B,C), sobre(D,mesa)
  - Combinar las soluciones de los subobjetivos en vez de la solución de la conjunción de objetivos simultaneamente.
  - Puede haber interacciones negativas entre los subplanes para cada subobjetivo.
    - Una acción de un subplan elimina un objetivo conseguido por otro subplan.
    - Habría que añadir acciones adicionales al combinar las soluciones.

Planificación como búsqueda

- Un enfoque que trabaje en varios sub-objetivos independientemente, que los solucione con varios subplanes y combine el conjunto de sub-planes utilizados, flexibiliza el orden en el que se construye el plan.
- No está forzado a trabajar en orden.
- Solución de orden total → varias linealizaciones válidas de una solución de orden parcial

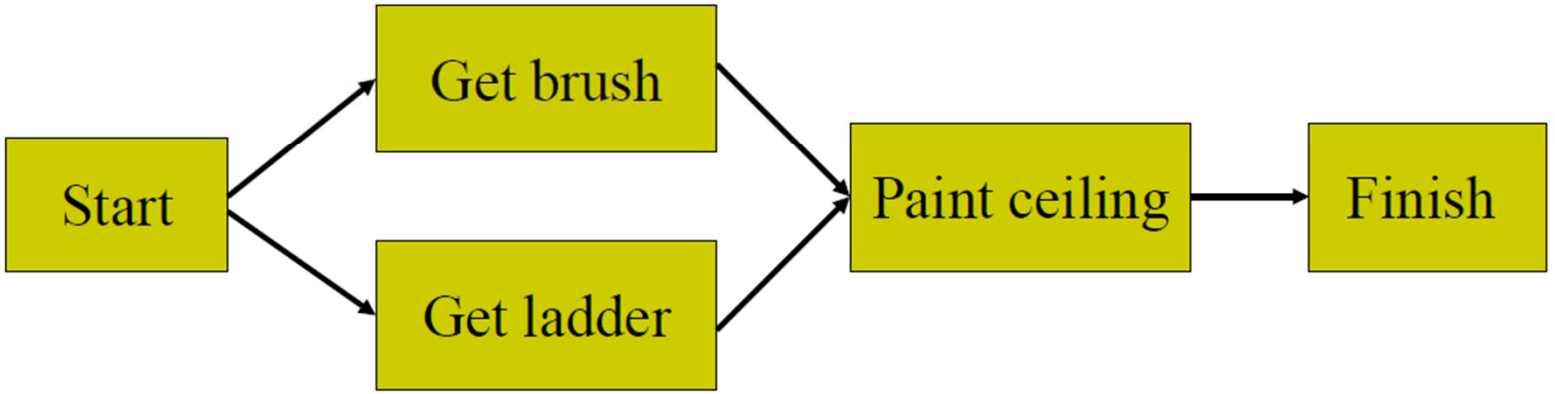
Secuenciación de un plan parcial: colocar las acciones una detrás de otra, sin contradecir ninguna restricción de orden que se deduzca del plan parcial
Punto clave: cualquier secuenciación de un plan parcial solución supone una solución al problema original


Plan de Orden Parcial:
Inicio
Calcetín izquier.
Calcetín derecho
CalcetínIzPuesto
CalcetínDrPuesto
Zapato izquier.
Zapato derecho
ZapatoIzPuesto
ZapatoDrPuesto
Fin
Plan de Orden Total:
Inicio
Calcetín derecho
Calcetín izquier.
Zapato derecho
Zapato izquier.
Fin
Inicio
Calcetín derecho
Calcetín izquier.
Zapato izquier.
Zapato derecho
Fin
Inicio
Calcetín izquier.
Calcetín derecho
Zapato derecho
Zapato izquier.
Fin
Inicio
Calcetín izquier.
Calcetín derecho
Zapato izquier.
Zapato derecho
Fin
Inicio
Calcetín derecho
Calcetín izquier.
Zapato derecho
Zapato izquier.
Fin
Inicio
Calcetín izquier.
Calcetín derecho
Zapato izquier.
Zapato derecho
Fin

Planificación como búsqueda en el espacio de planes parciales

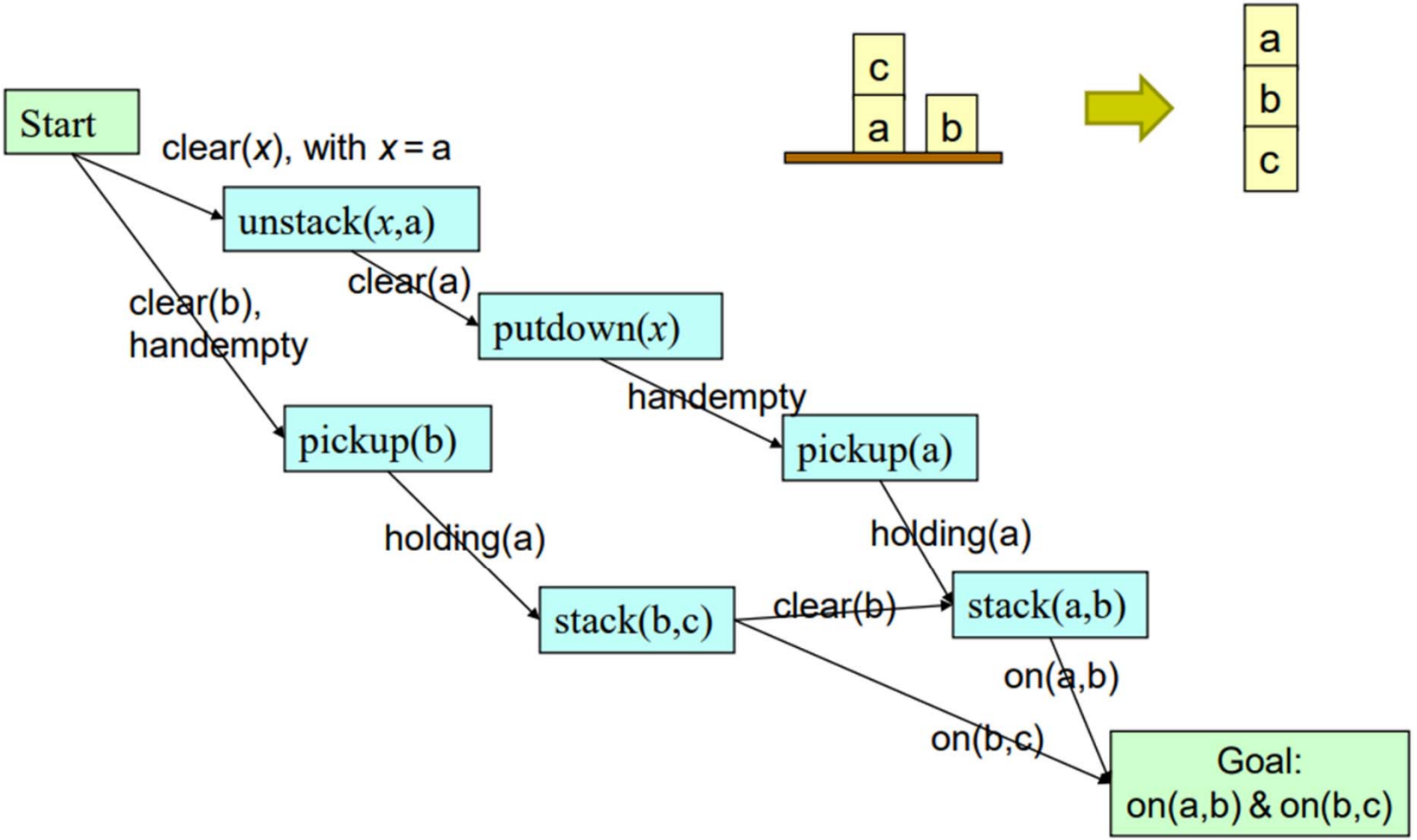
- **Búsqueda en espacio de estados**
  - Nodos: estado del mundo
  - Acciones sobre los estados
- Planificación como **búsqueda en el espacio de planes parciales**
  - Algoritmos de **búsqueda local** que transforman un **plan parcialmente especificado**
  - Estado inicial es un **plan parcialmente especificado**
    - Sólo incluye el estado inicial y final del mundo
  - Acciones: **operaciones de refinamiento del plan**
    - Añadir/eliminar acción, añadir causal links,…
  - La solución es un **plan (de orden parcial (POP))**
    - No secuencias de acciones
    - Plan = grafo / dependencias / temporalidad..

- Plan parcialmente ordenado (Partial-order plan)
  - Compuesto por un conjunto de acciones ordenadas parcialmente
  - Existen restricciones de secuencia en estas acciones
  - Un algoritmo de generación de planes se puede usar para transformar un plan parcialmente ordenado en un plan totalmente ordenado



[Ejemplo de Han Yu (University of Central Florida)]

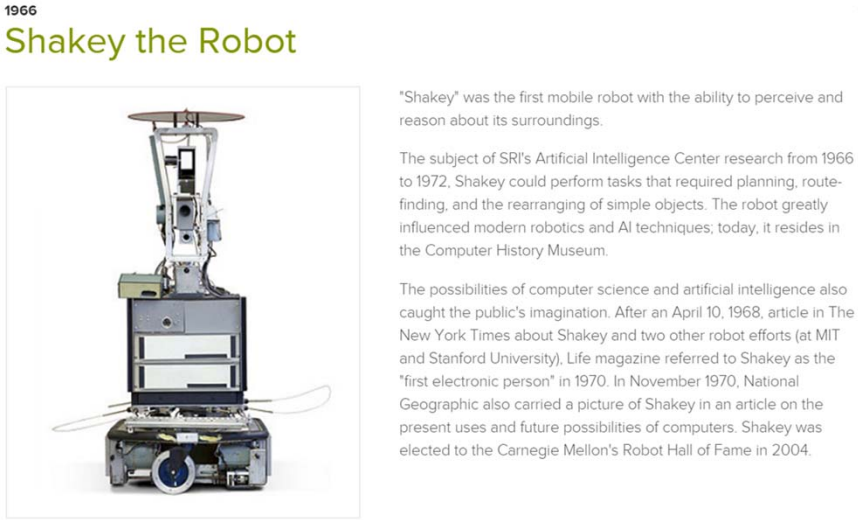
Ejemplo plan



Lenguajes de Planificación

- **STRIPS** (**St**anford **R**esearch **I**nstitute **P**roblem **S**olver) es un generador de planes automatizado y el lenguaje formal de las entradas de este generador de planes.
  - Uso de predicados (lógica de primer orden), solo literales positivos, objetivos limitados a conjunción de literales simples.
  - Hipótesis de mundo cerrado.
- Otros lenguajes de planificación más expresivos: ADL, PDDL
- PDDL estándar de facto → lenguaje para la competición de planificadores

"STanford Research Institute Problem Solver" fue el planificador utilizado en “Shakey” uno de los primeros robots con IA.
Existe un brazo mecánico, con el que se intenta resolver problemas en el dominio del mundo de los bloques



Ejemplo

- Hay un mono en el laboratorio, y quiere bananas, hay tres ubicaciones en el laboratorio: A, B y C. El mono está en la ubicación A. Hay una caja en la ubicación C. Hay bananas en la ubicación B, pero cuelgan del techo. El mono necesita la caja para alcanzar todas las bananas.

Representación en STRIPS

inicial

MonoEn(C)

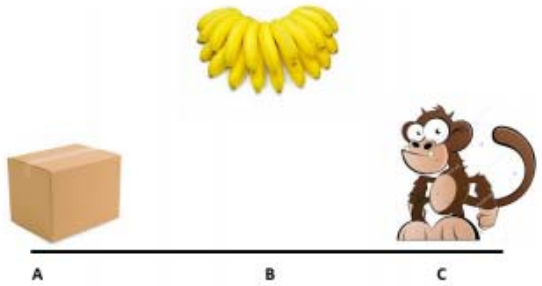
NivelMono(abajo)

CajaEn(A)

BananasEn(B)

objetivo

MonoTiene(Bananas)

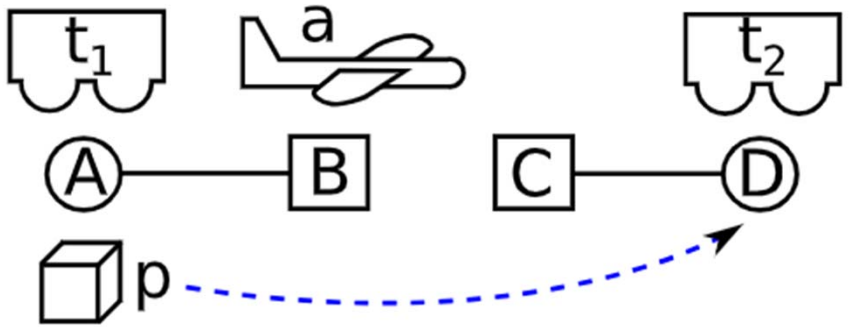


Las acciones que se pueden realizar son las siguientes: IR de una localización a otra; DESPLAZAR un objeto de una localización a otra; SUBIRSE a un objeto; y AGARRAR un objeto (se puede agarrar un objeto si ambos están en la misma ubicación y a la misma altura).

Acciones en STRIPS

	Precondición	Poscondición
Ir(X, Y)	MonoEn(X), NivelMono(abajo)	¬MonoEn(X), MonoEn(Y)
Trepar(X)	MonoEn(X), CajaEn(X), NivelMono(Abajo)	¬NivelMono(Abajo), NivelMono(Arriba)
Bajarse(X)	MonoEn(X), CajaEn(X), NivelMono(Arriba)	¬NivelMono(Arriba), NivelMono(Abajo)
LlevarCaja(X, Y)	MonoEn(X), CajaEn(X), NivelMono(Abajo)	¬CajaEn(X), CajaEn(Y), ¬MonoEn(X), MonoEn(Y)
TomarBananas(X)	MonoEn(X), BananasEn(X), NivelMono(Arriba)	MonoTiene(Bananas)

Razonamiento lógico para resolver el problema



- P propositions:
  - truck-at-A, truck-at-B,
  - plane-at-B, plane-at-C,
  - package-at-A, package-at-B, package-at-C,
  - package-in-t, package-in-a
- $2^9 = 512$ states

Estado inicial

```
(at mydvd amazon)
(at truck amazon)
(at driver home)
(path home amazon)
(link amazon london)
(link london myhouse)
```

Problema de logística
Entrega de DVDs

Conjunto de proposiciones que nos dicen el estado del mundo antes de planificar

Abstracción de la representación
Puede ser una carretera, muchas,..

Estado objetivo

```
(at mydvd myhouse)
```

Un objetivo es un estado parcialmente especificado, representado como una secuencia de literales positivos y simples.

Conjunto de proposiciones que nos describen
el estado del mundo en el que quiero estar después de planificar

Hipótesis de mundo cerrado

No hace falta especificarlo entero cualquier estado que cumpla esto me vale.

Acciones

```
(load mydvd truck depot)
(walk driver home amazon)
(board-truck driver truck amazon)
(drive-truck driver amazon london)
```

Reglas de lo que puede pasar
Estas acciones se incluyen en lo que
Se llama un **DOMINIO**.

Cada dominio tiene un conjunto de acciones

Las acciones tienen parámetros de los objetos involucrados.

```
(:action load
  (:parameters(?d - dvd ?t - truck ?dep - depot)
  (:precondition (and (at ?t ?dep) (at ?d ?dep))
  (:effect (and (not (at ?d ?dep))
                (in ?d ?t))
  )
```

Cada acción tiene que estar bien especificada: cuando podemos hacerla (precondiciones) y qué pasa cuando la aplicamos (efectos)

Realmente esto no es nada nuevo salvo la sintaxis lógica para definir operadores
En los estados hay constantes y en las acciones hay variables y hay que ligarlas.

Fichero de descripción del dominio

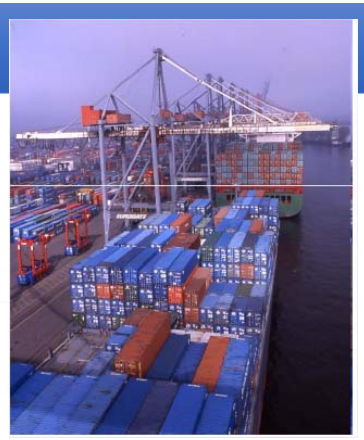
```
(define (domain driverlog)
  (:requirements :strips :typing)
  (:types location locatable - object
          driver truck obj - locatable

  )
  (:predicates
     (at ?obj - locatable ?loc - location)
     (in ?obj1 - obj ?obj - truck)
     (driving ?d - driver ?v - truck)
     (link ?x ?y - location) (path ?x ?y - location)
     (empty ?v - truck)
  )
  (:action LOAD-TRUCK
  :parameters
   (?obj - obj
    ?truck - truck
    ?loc - location)
  :precondition
   (and (at ?truck ?loc) (at ?obj ?loc))
  :effect
   (and (not (at ?obj ?loc)) (in ?obj ?truck)))
```

Fichero de descripción del problema

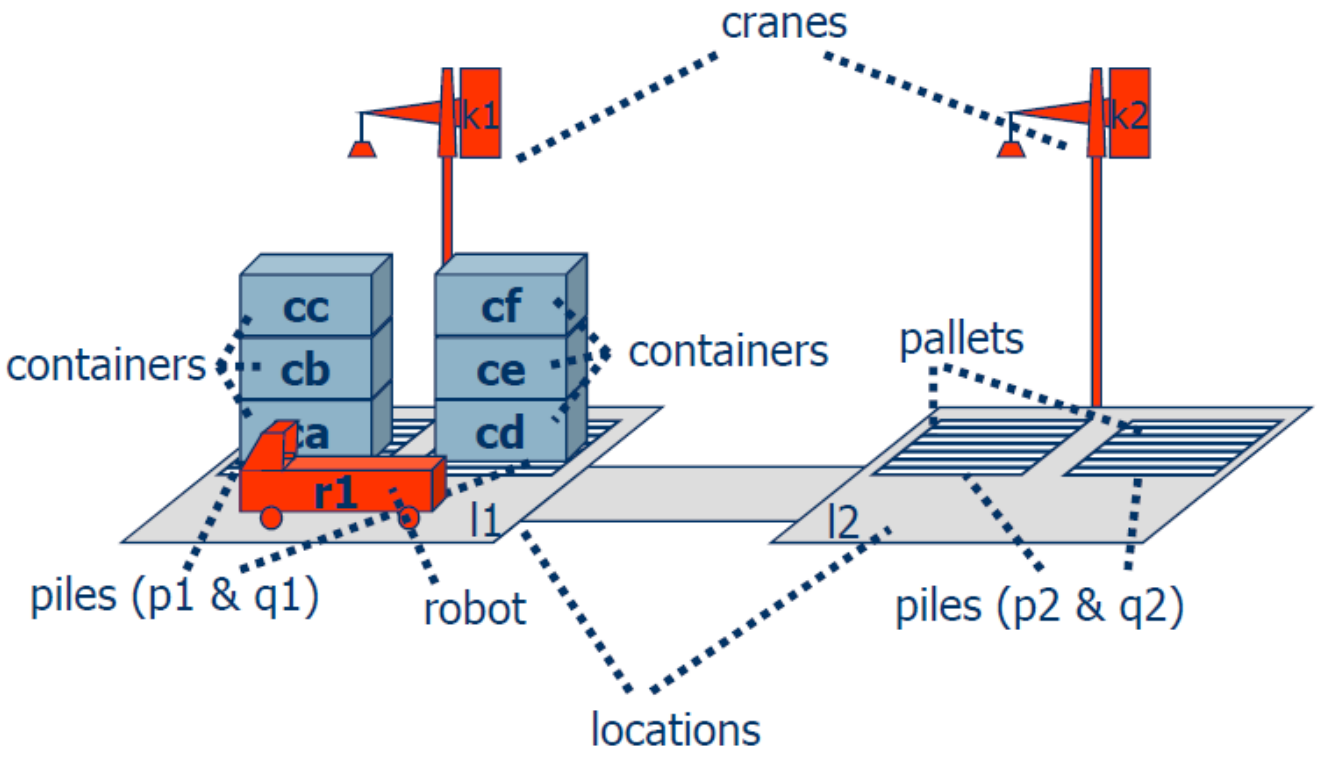
```
(define (problem DLOG-2-2-2)
    (:domain driverlog)
    (:objects
      driver1 - driver
      truck1 - truck
      package1 - obj
      s0 - location
      s1 - location ...)
    (:init
    (at driver1 s12)
    (at truck1 s0)
    (empty truck1)
    (at package1 s0)
    (path s1 p1-0)
    (path p1-0 s1)
    ...
    (link s0 s1)
    (link s1 s0)
    ... )
    (:goal (and (at driver1 s1)
    (at truck1 s1)
    (at package1 s0)
    )))
```

Ejemplo



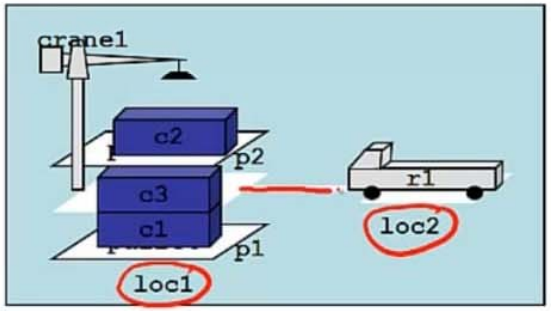
DWR [Dock-Worker Robots]

Estados



DWR Example: STRIPS States

state = {
adjacent(loc1,loc2), adjacent(loc2, loc1),
attached(p1,loc1), attached(p2,loc1),
belong(crane1,loc1),
occupied(loc2),
empty(crane1),
at(r1,loc2),
unloaded(r1),
in(c1,p1),in(c3,p1),
on(c3,c1), on(c1,pallet),
top(c3,p1),
in(c2,p2),
on(c2,pallet),
top(c2,p2)}



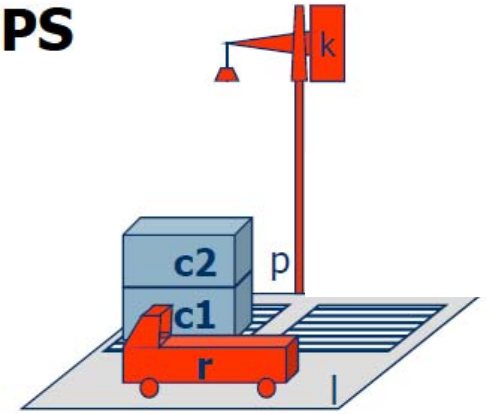
http://projects.laas.fr/planning/aptp/index.html

Representación de acciones en STRIPS

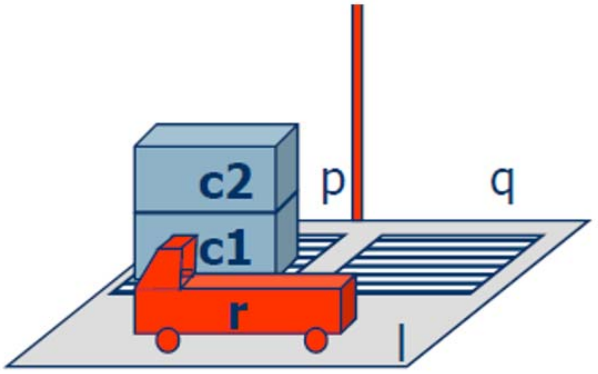
move(r,l,m)
- pre: adjacent(l,m), at(r,l)
- add: at(r,m)
- del: at(r,l)

load(k,l,c,r)
- pre: at(k,l), holding(k,c), at(r,l), unloaded(r)
- add: empty(k), loaded(r,c)
- del: holding(k,c), unloaded(r)

put(k,l,c,d,p)
- pre: at(k,l), at(p,l), holding(k,c), top(d,p)
- add: empty(k), in(c,p), top(c,p), on(c,d)
- del: holding(k,c), top(d,p)



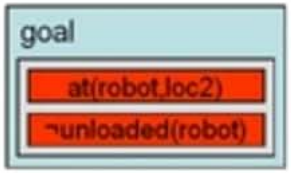
move(r,l,m)
- pre: adjacent(l,m), at(r,l)
- eff: at(r,m), ¬at(r,l)



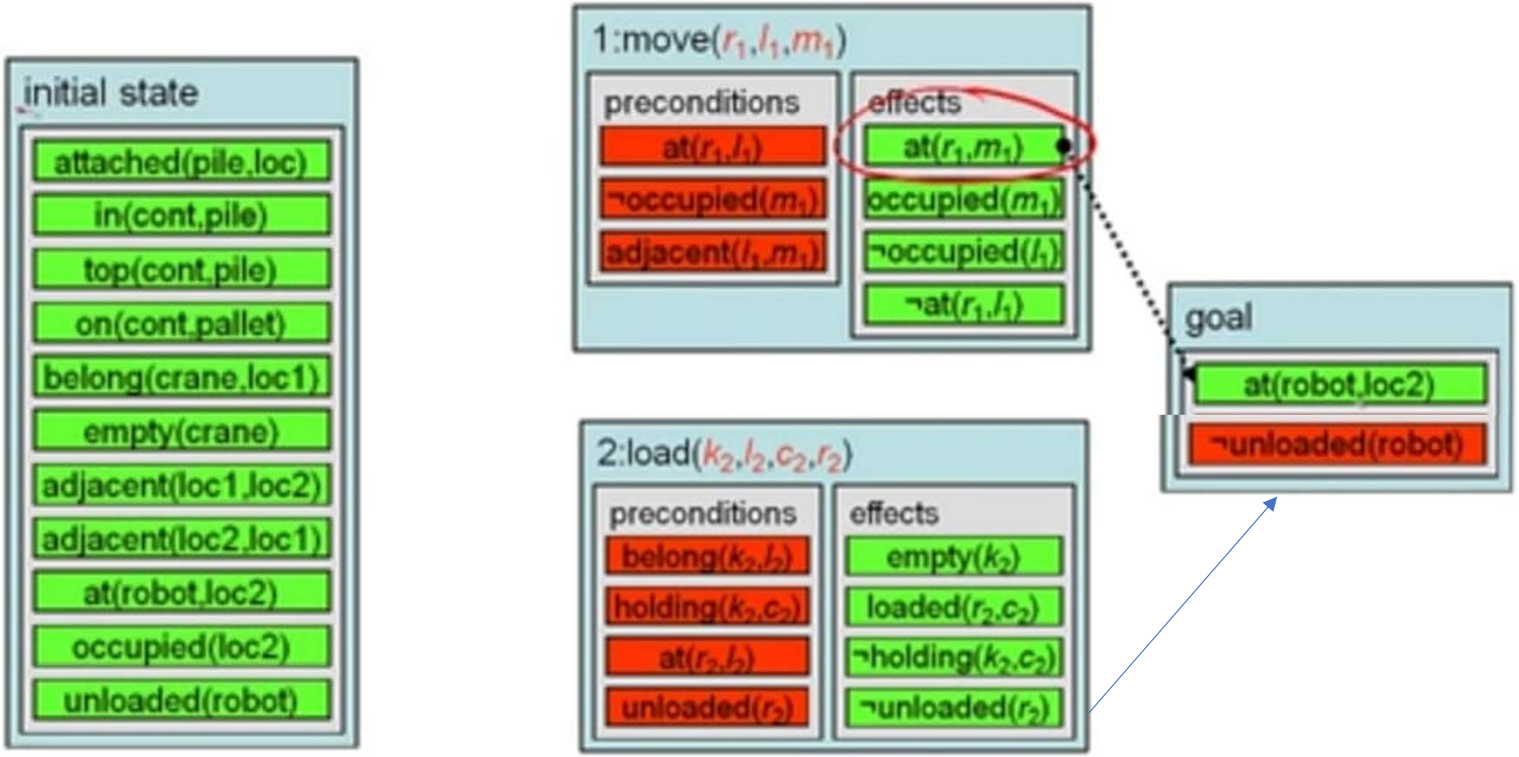
Planificación como búsqueda en el espacio de planes parciales

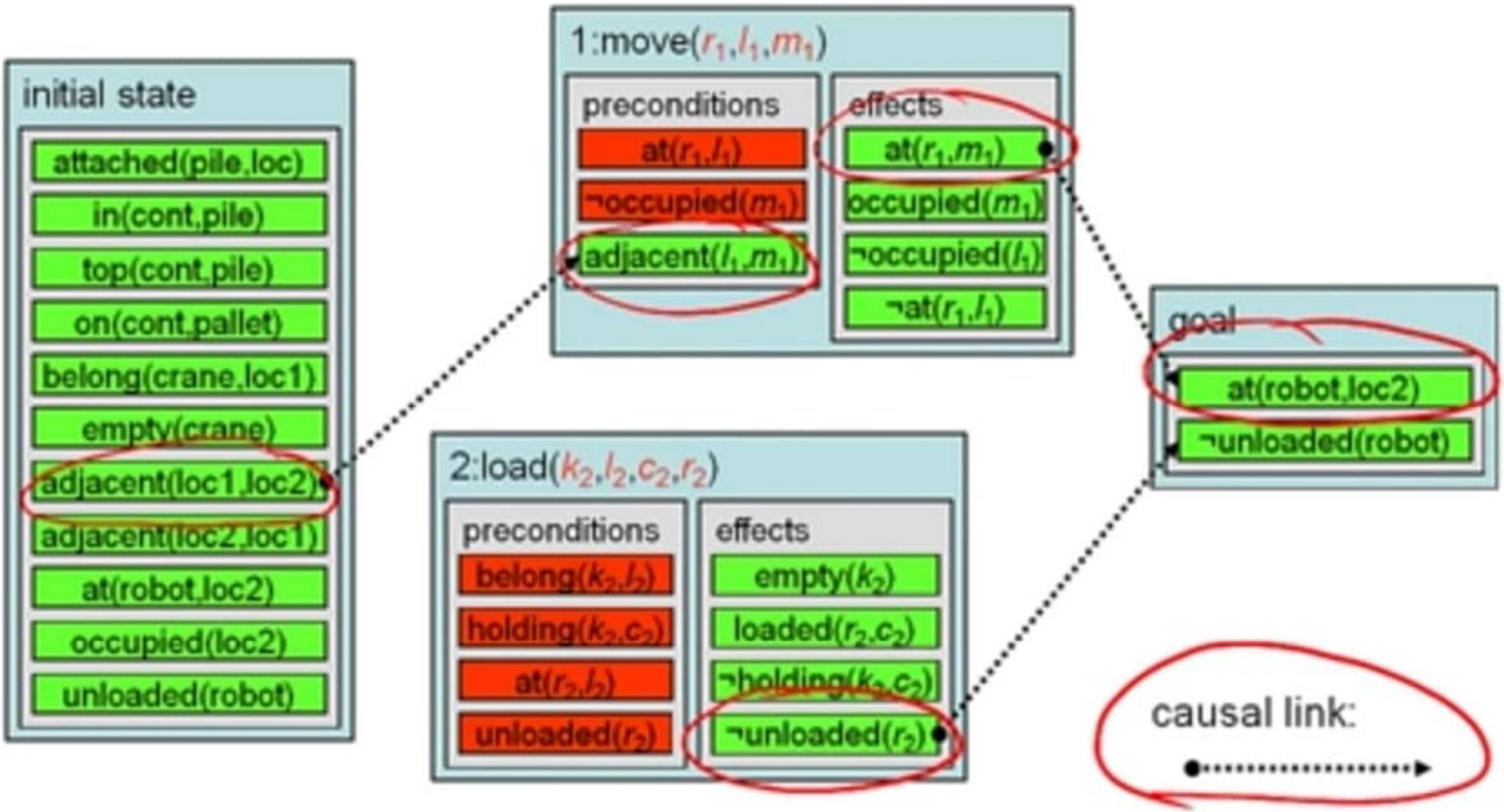
- Planificación en el espacio de planes (Plan-space)
- Búsqueda local: plan parcial que incluye:
  - Estados inicial, final, intermedios
  - un conjunto de operadores parcialmente instanciados
  - un conjunto de restricciones sobre los operadores
- Proceso:
  - Descomponer conjuntos de objetivos en objetivos individuales
  - Planificar para cada uno de ellos por separado
  - Se van detectando y resolviendo los 'fallos' que hacen que aun no sea un plan, imponiendo más y más restricciones hasta que se tiene un plan parcialmente ordenado.
- Una extensión de planificación temporal en el espacio de planes se ha usado recientemente en los Mars Rovers de NASA.

- Estado **inicial** (plan parcial que incluye el estado inicial y el objetivo)
  - Operadores: transforman el plan parcial → Añadir una acción (move, load, ..)

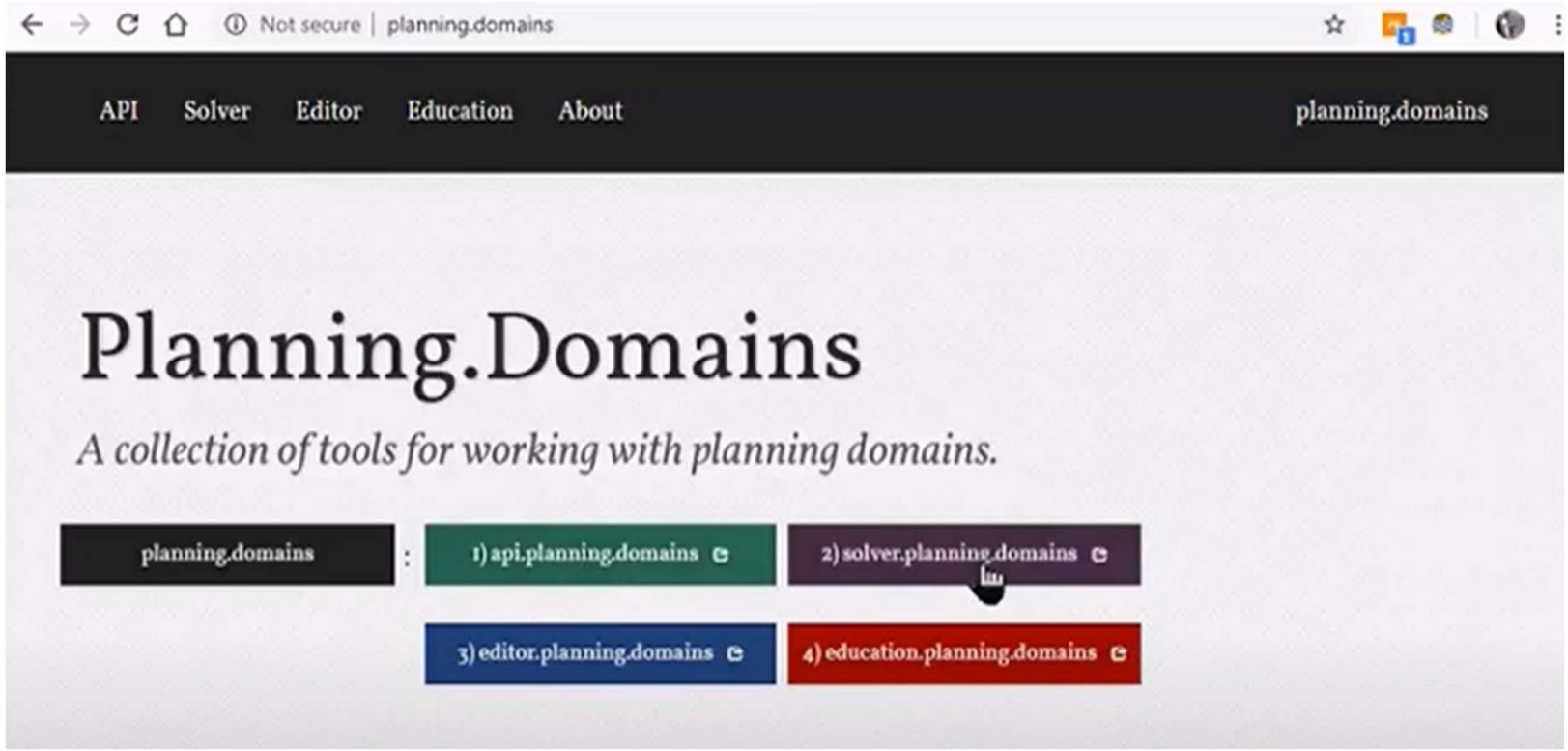


- Estado **inicial** (plan parcial que incluye el estado inicial y el objetivo)
  - Operadores: transforman el plan parcial → Añadir una acción (move, load, ..)




initial state
attached(pile,loc)
in(cont,pile)
top(cont,pile)
on(cont,pallet)
belong(crane,loc1)
empty(crane)
adjacent(loc1,loc2)
adjacent(loc2,loc1)
at(robot,loc2)
occupied(loc2)
unloaded(robot)
1:move(r_1,l_1,m_1)
preconditions
at(r_1,l_1)
¬occupied(m_1)
adjacent(l_1,m_1)
effects
at(r_1,m_1)
occupied(m_1)
¬occupied(l_1)
¬at(r_1,l_1)
goal
at(robot,loc2)
¬unloaded(robot)
2:load(k_2,l_2,c_2,r_2)
preconditions
belong(k_2,l_2)
holding(k_2,c_2)
at(r_2,l_2)
unloaded(r_2)
effects
empty(k_2)
loaded(r_2,c_2)
¬holding(k_2,c_2)
¬unloaded(r_2)
causal link:

http://editor.planning.domains/



https://www.youtube.com/watch?v=XW0z8Oik6G8