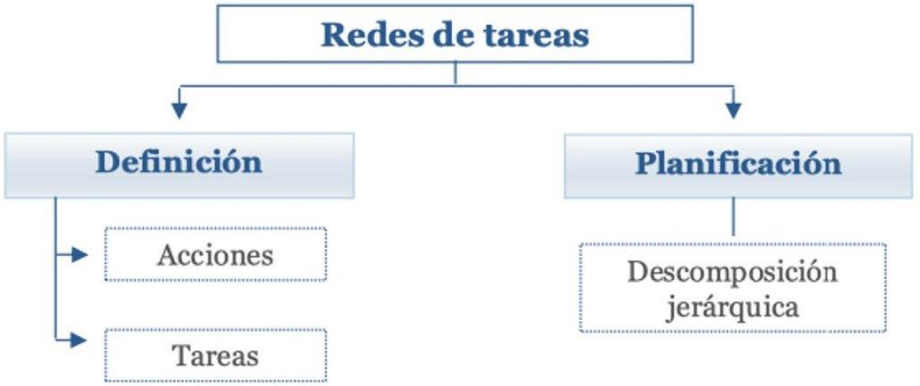
Razonamiento y planificación automática

Nerea Luis Mingueza / Alejandro Cervantes

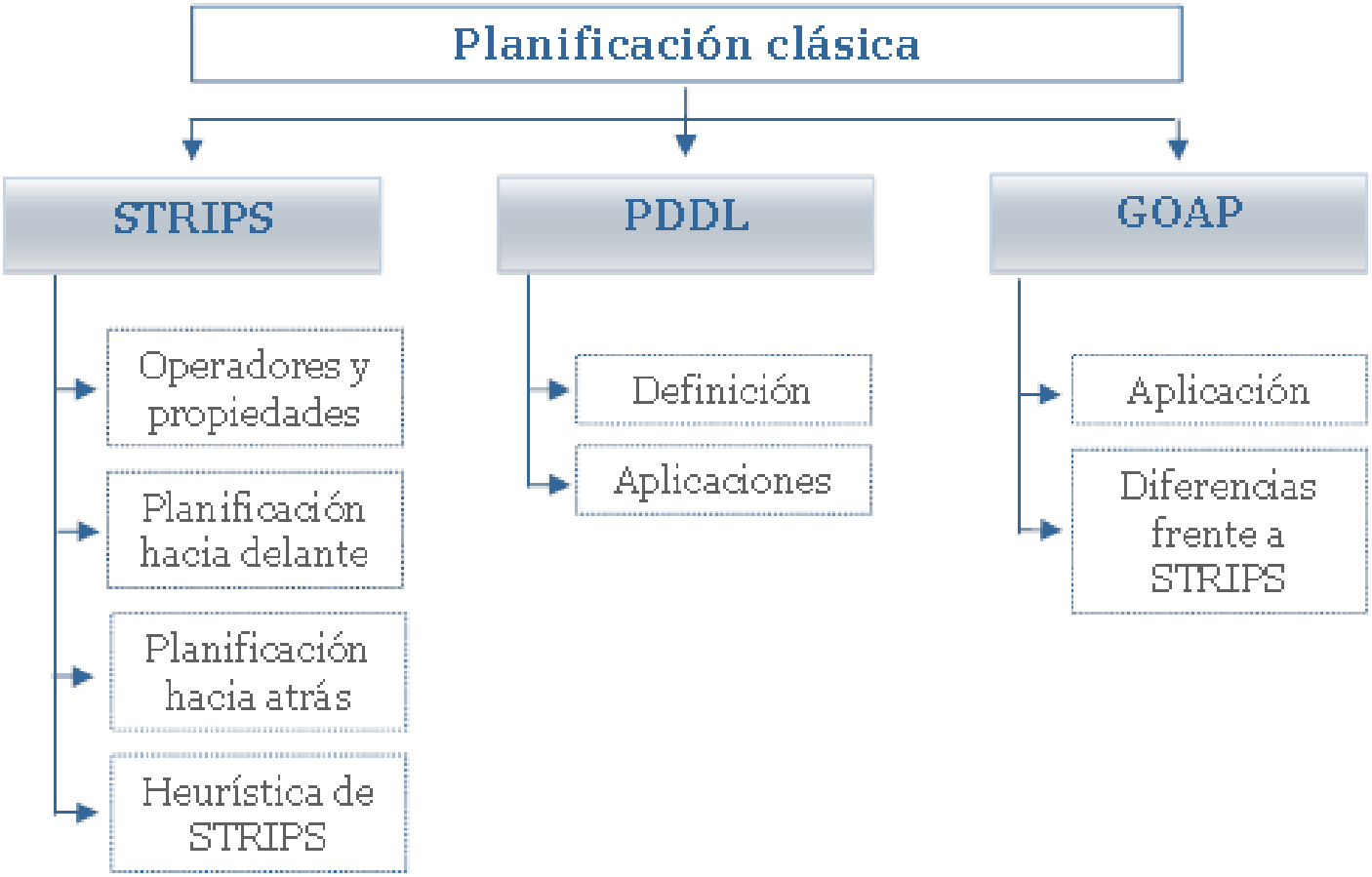
Tema 9: Redes Jerárquicas de Tareas

Índice

- Definición
- Estrategia
- Tareas y Métodos
- Planificación usando HTN



Planificación Clásica (Repaso)


Planificación clásica
STRIPS
PDDL
GOAP
Operadores y propiedades
Planificación hacia delante
Planificación hacia atrás
Heurística de STRIPS
Definición
Aplicaciones
Aplicación
Diferencias frente a STRIPS

Componentes de la planificación

- **Estado actual del entorno:** es una representación estructura que crea el agente al momento de percibir su entorno. Como, por ejemplo, su posición actual, si llueve o no llueve, la posición de una mesa, etc.
- **Meta:** es cualquier condición que un agente quiera satisfacer. Un agente puede tener varias posibles metas, pero en un instante determinado solo una puede estar activa, controlando el comportamiento.
- **Acción:** es un paso simple y atómico dentro de un plan que hace que un agente haga algo (ir a un punto, activar un objeto, etc.)
- **Plan:** secuencia de acciones.
- **Proceso de planificación:** Un agente proporciona a un sistema (planificador) un estado actual del entorno, un conjunto de acciones y una meta que desea satisfacer, y el planificador busca un plan que con la ejecución de sus acciones consiga esta meta.

Problemas de la planificación clásica

- Tiempo
- Escalabilidad
- Complejidad

Redes Jerárquicas de Tareas

Motivación

- En los problemas reales los expertos pueden establecer relaciones entre tareas con dependencias entre sí (redes). Estas tareas a realizar corresponden a los objetivos de planificación clásica.
- Divide y vencerás: partir del concepto de tarea y llegar al objetivo de red instanciada
- En los niveles finales del plan aparecen tareas atómicas que se resuelven mediante secuencias (parcialmente ordenadas) de operadores
- Dos niveles de lenguaje, mayor expresividad
- Inconveniente: crear el dominio es más complejo y requiere más conocimiento de experto

Ejemplo

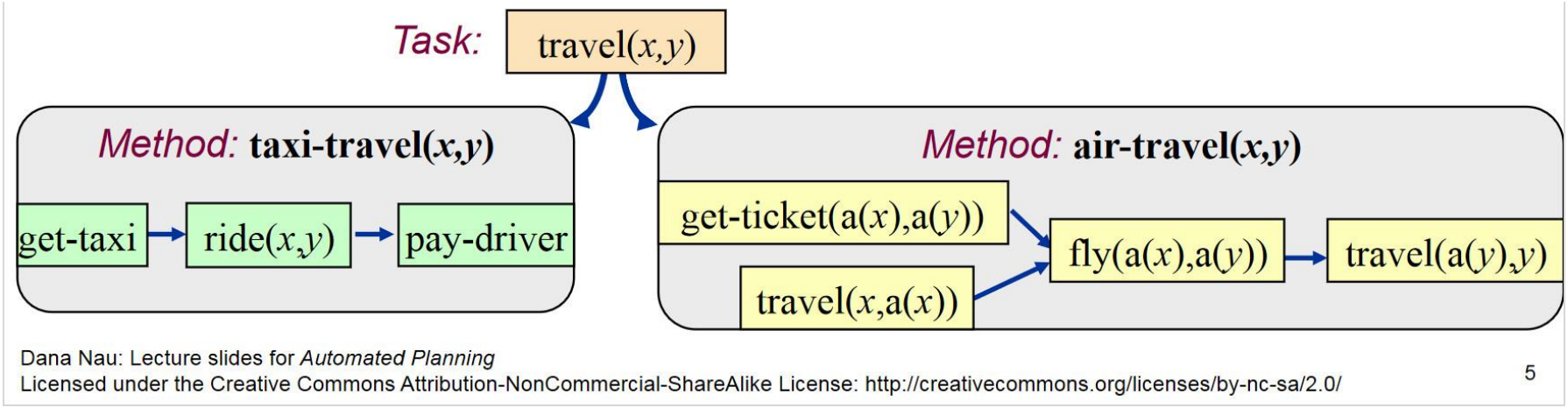
- Supongamos un problema de logística donde un tren puede cargar hasta 2 contenedores simultáneamente y luego llevarlos entre diversas localizaciones. Existirá una acción "`cargar(contenedor)`", otra "`mover(localización)`" y otra "`descargar(contenedor)`".

- Tendremos un objetivo: ambos contenedores en sus destinos: `(and (en contenedor1, destino1) (en contenedor2, destino2))`

- Un planificador clásico dividirá el objetivo en dos: `(en contenedor1, destino1)` y `(en contenedor2, destino2)`. Si no tiene un criterio de optimización, probablemente intente obtener primero el objetivo 1 y luego el objetivo 2: `cargar(1), mover(destino1), descargar(1),mover(origen),cargar(2)`, etc.

- Sin embargo, el conocimiento de estos dominios nos dice que, si el tren pudiera cargar varios contenedores, casi siempre será preferible que el tren vaya lleno.
  Llenar_tren: hasta(lleno): *cargar*(contenedores) **Tarea compuesta**

Descomposición en tareas



Observación: usamos ya siempre grafos de acciones: a & b -> c , etc.

Fuente: https://www.cs.umd.edu/~nau/cmsc722/slides/chapter11.pdf

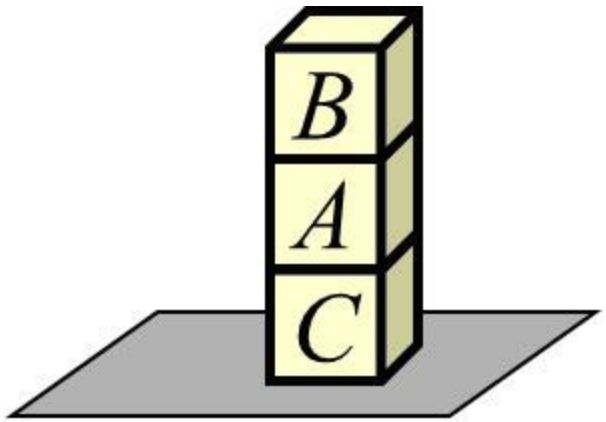
Simple Task Network (STN)

La planificación mediante red de tareas es una técnica de planificación de la IA que rompe con la tradición de la planificación (Ghallab, 2004). La idea básica detrás de esta técnica incluye:

- Una descripción de estado inicial
- Una red de tareas inicial como un objetivo a alcanzar (corresponden a objetivos de planificación clásica)
- Un conocimiento de dominio, que consiste en redes de tareas primitiva y compuestas (abstractas).

Ejemplo de STN

Objetivo: invertir el orden de la pila



Procedimiento recursivo

Tarea: iniciar-inversión(X_{TOP})

Quitar(X_{TOP},Y), Poner-mesa(X_{TOP}), invertir-encima(Y,X_{TOP})

Método 1

Tarea: invertir-encima(Y_{TOP},X_{BASE})

Quitar(Y_{TOP},Z), Poner-sobre(Y_{TOP},X_{BASE}), invertir-encima(Z,Y_{TOP})

Método 2

Tarea: invertir-encima(Y_{TOP},X_{BASE})

Quitar(Y_{TOP},Z), Poner-sobre(Y_{TOP},X_{BASE}), fin-inversion(Z,Y_{TOP})

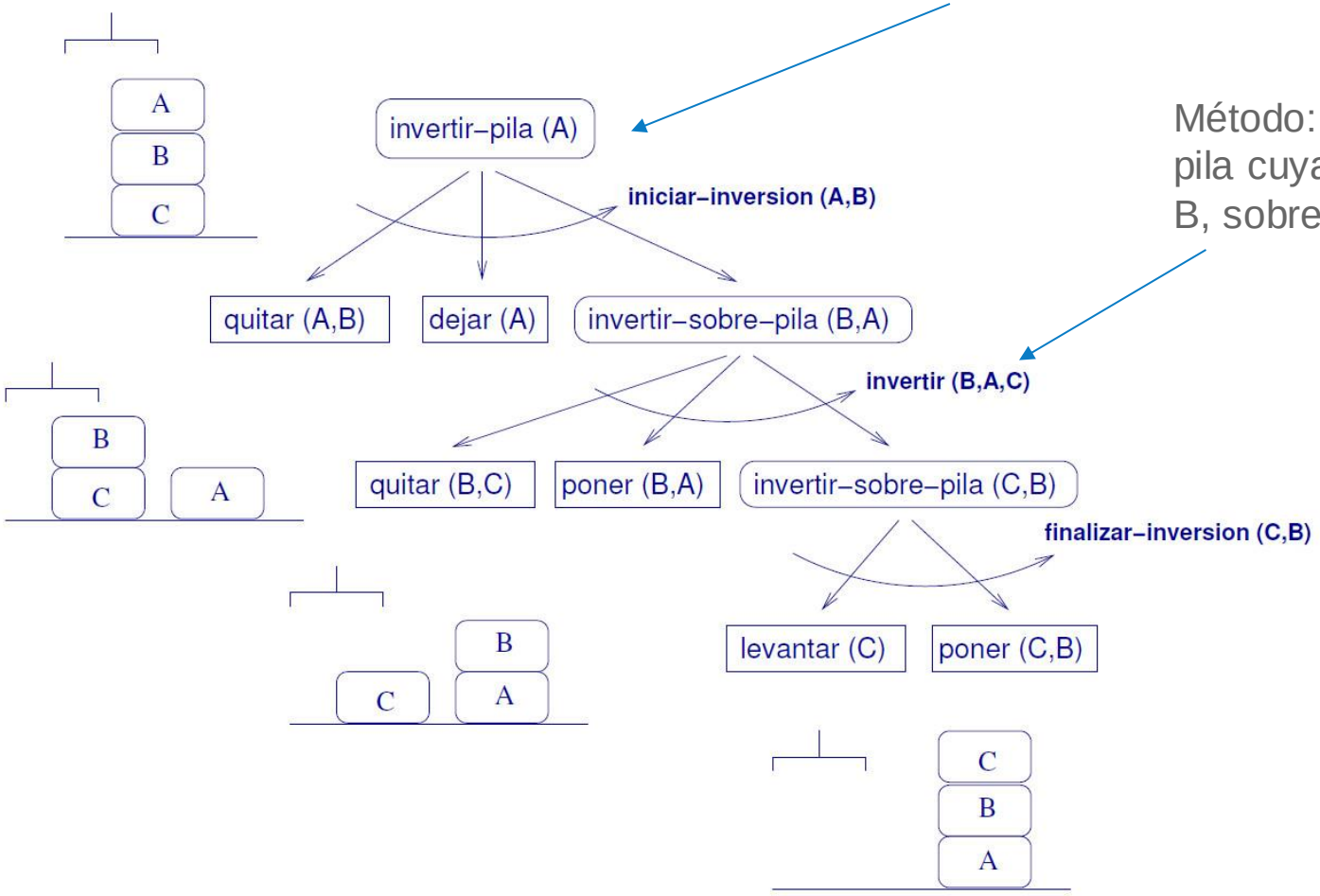
Tarea: fin-inversión(Z_{MESA},X_{BASE})

Sujetar(Z_{MESA}), Poner-sobre(Z_{MESA},X_{BASE})

Ejemplo de STN

Tarea: nvertir la pila cuya cima es A, sobre mesa

Método: Invertir la pila cuya cima es B, sobre A



Definiciones

Tipos de tareas: compuestas o primitivas

Entradas: estado inicial, tareas compuestas, orden en el que se deben conseguir y teoría del dominio (métodos y acciones)

Métodos (methods): están formados por

- Nombre de una tarea compuesta
- precondiciones para poder realizar la descomposición
- conjunto de subtareas en las que se puede descomponer
- restricciones que se deben cumplir en esas subtareas

Acciones: consiguen las tareas primitivas. Semejantes a los operadores de planificación STRIPS.

HTN Hierarchical Task Network

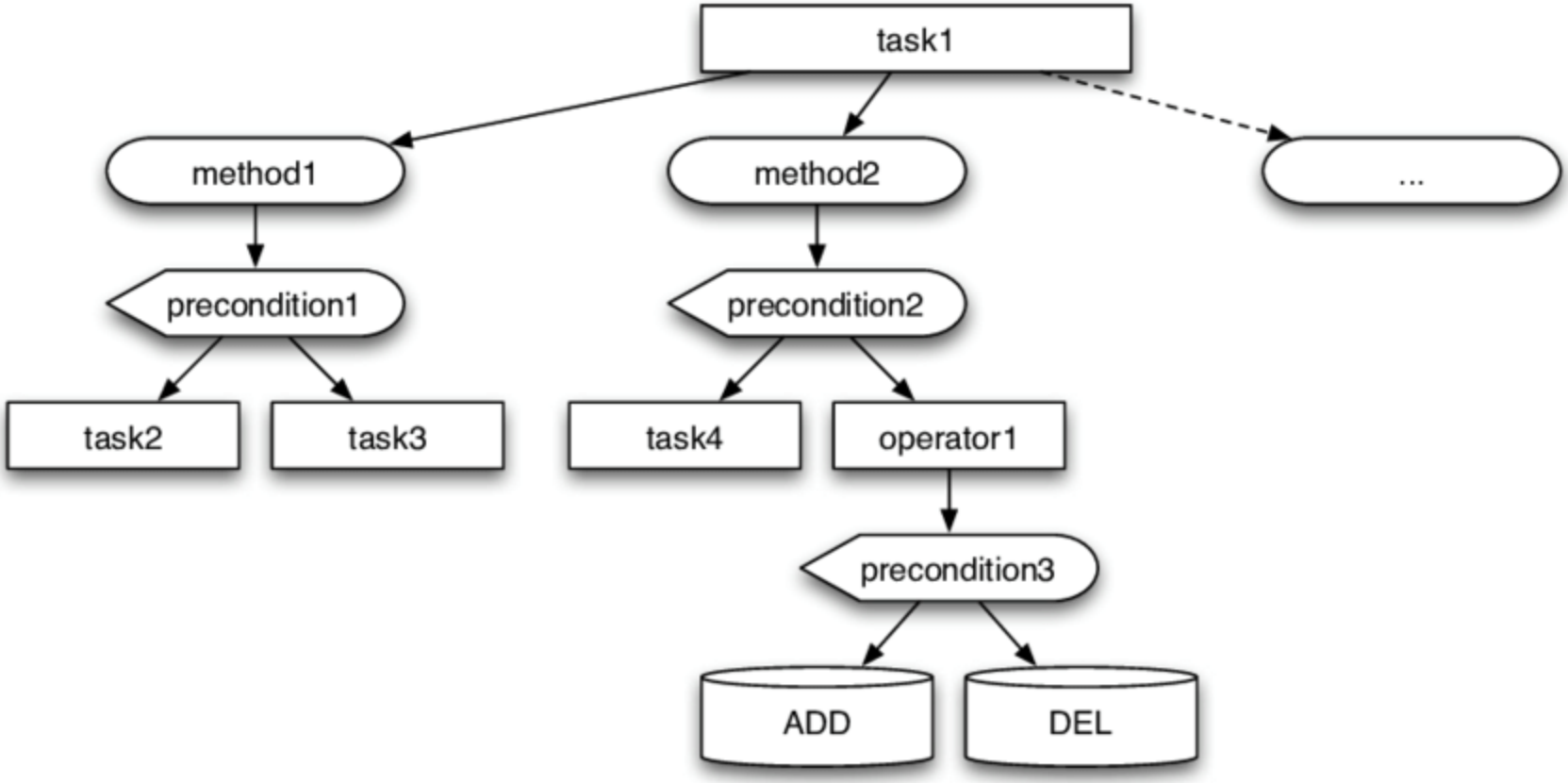
Una HTN se define como w = (U, C) donde C es un conjunto de restricciones:

- *de-precedencia*: u ≺ v
- *antes-de*: `before (U',l)` establece que en cualquier plan π solución de P, l debe ser cierto antes de U'
- *después-de* : `after (U',l)` establece que en cualquier plan π solución de P, l debe ser cierto despues de U'
- *en-medio-de*: `between (U',U'',l)` establece que l debe ser cierto después de U' y mantenerse así al menos hasta el estado anterior a U''

Los métodos HTN se describen entonces como:

```
m = (name(m), task(m), precond(m), network(m))
```

http://ocw.uc3m.es/cursos-archivados/planificacion-automatica/transparencias/conocimiento-control.pdf


task1
method1
method2
...
precondition1
precondition2
task2
task3
task4
operator1
precondition3
ADD
DEL

Ejemplo de definición HTN

método:	iniciar-inversión (b1, b2)
tarea:	invertir-pila (b1)
subtareas:	u_1 =quitar (b1, b2), u_2 = dejar (b1), u_3 = invertir-sobre-pila (b2, b1)
restriccs:	$u_1 \prec u_2, u_2 \prec u_3$, before ($\{u_1\}$, libre (b1)), before ($\{u_1\}$, encima (b1,b2)), before ($\{u_1\}$, brazo-libre), before ($\{u_3\}$, libre (b1))

método:	invertir (b1, b2, b3)
tarea:	invertir-sobre-pila (b1, b2)
subtareas:	u_1 = quitar (b1, b3), u_2 = poner (b1, b2), u_3 = invertir-sobre-pila (b3, b1)
restriccs:	$u_1 \prec u_2, u_2 \prec u_3$, before ($\{u_3\}$, libre (b1)) before ($\{u_1\}$, encima (b1,b3)), before ($\{u_1\}$, brazo-libre), before ($\{u_3\}$, libre (b1))

método:	finalizar-inversión (b1, b2)
tarea:	invertir-sobre-pila (b1, b2)
subtareas:	u_1 = levantar (b1), u_2 = poner (b1, b2)
restriccs:	$u_1 \prec u_2$

Fuente: http://ocw.uc3m.es/cursos-archivados/planificacion-automatica/transparencias/conocimiento-control.pdf

Ejemplo de métodos en HTN

```
(:method (obj-at ?obj ?loc-goal)

same-city-deliver
  ((in-city ?loc-goal ?city-goal) (obj-at ?obj ?loc-now)
   (in-city ?loc-now ?city-goal) (truck ?truck ?city-goal))
  ((:task in-city-delivery ?truck ?obj ?loc-now ?loc-goal))

different-city-deliver
  ((in-city ?loc-goal ?city-goal) (obj-at ?obj ?loc-now)
   (in-city ?loc-now ?city-now) (different ?city-goal ?city-now)
   (truck ?truck-now ?city-now) (truck ?truck-goal ?city-goal)
   (airport ?airport-now) (in-city ?airport-now ?city-now)
   (airport ?airport-goal) (in-city ?airport-goal ?city-goal))

(:ordered
  (:task in-city-delivery ?truck-now ?obj ?loc-now ?airport-now)
  (:task air-deliver-obj ?obj ?airport-now ?airport-goal)
  (:task in-city-delivery ?truck-goal ?obj ?airport-goal ?loc-goal)
))
```

Fuente: http://ocw.uc3m.es/cursos-archivados/planificacion-automatica/transparencias/conocimiento-control.pdf

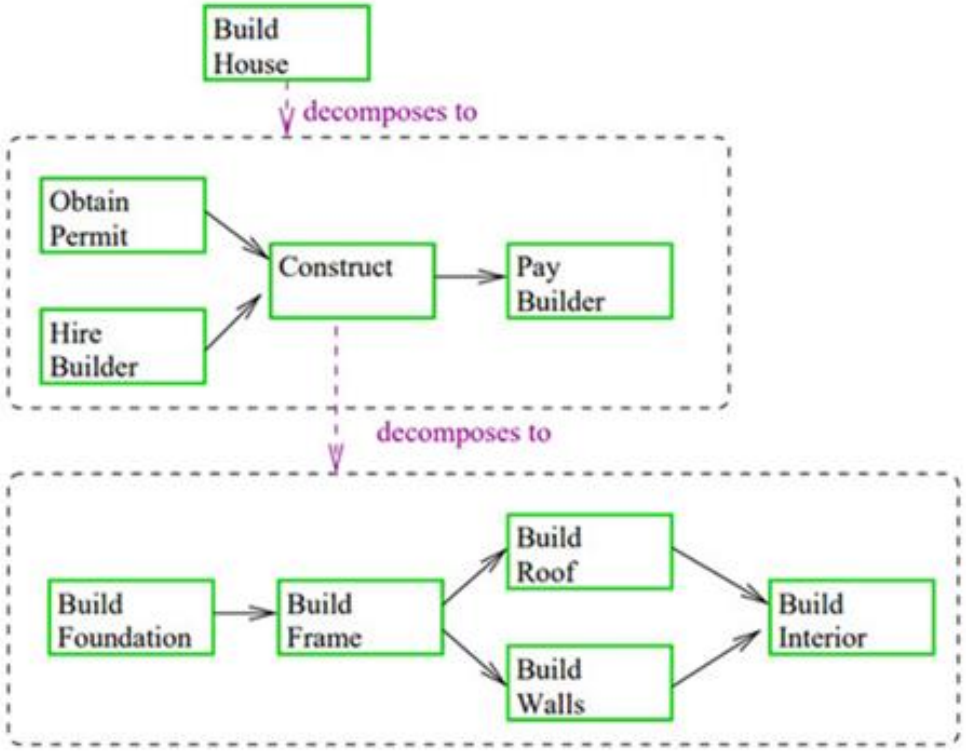
Ejemplos

Ejemplo de definición

Idea principal: muchas tareas en la vida real ya tienen una estructura jerárquica incorporada. Por ejemplo: una tarea computacional, una misión militar o una tarea administrativa.

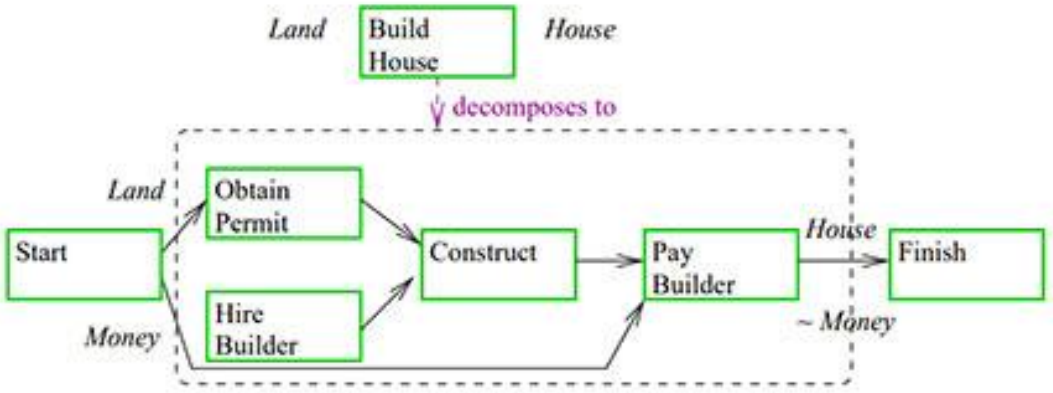
Usar la **jerarquía** incorporada en el dominio ayuda a escapar de la explosión exponencial de las posibles combinaciones que tendrían las acciones atómicas.

Ejemplo de aplicación: la actividad de construir una casa consiste en obtener los permisos necesarios, encontrar un constructor, construir el exterior/interior, etc. En el enfoque de las HTN, se utilizan operadores abstractos al igual que operadores primitivos durante la generación del plan.

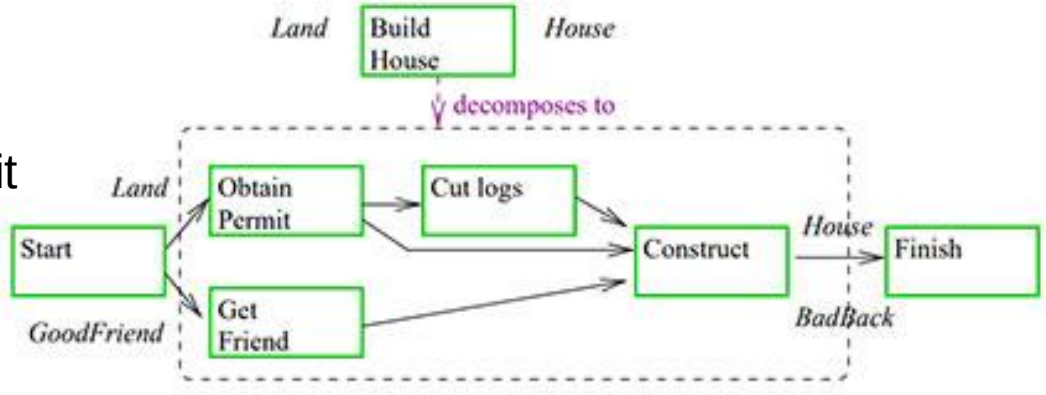


Ejemplo de posibles descomposiciones

P1: Obtain Permit
P2: Hire Builder
P3: Construct
P4: PayBuilder

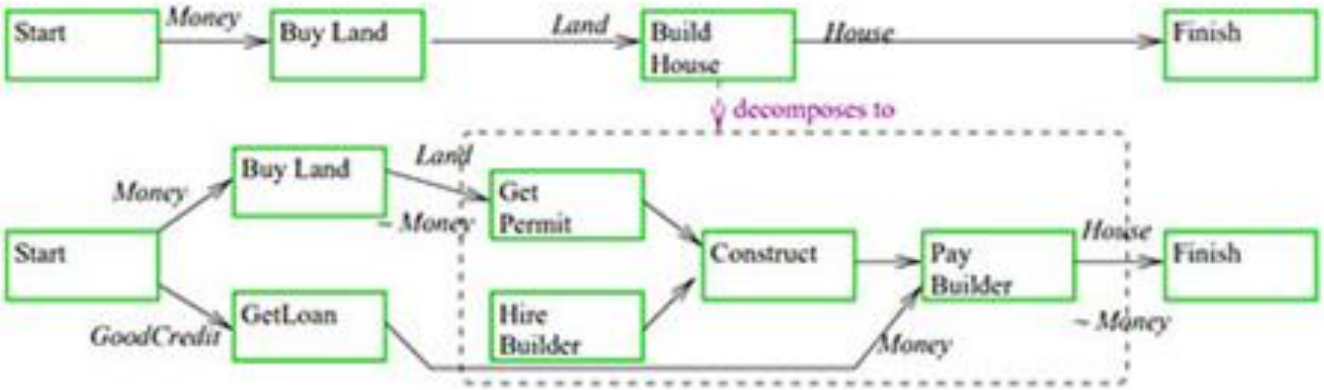


P1: Obtain Permit
P2: Get Friend
P3: Construct



Sobre las flechas, las precondiciones

Ejemplo de descomposiciones alternativas

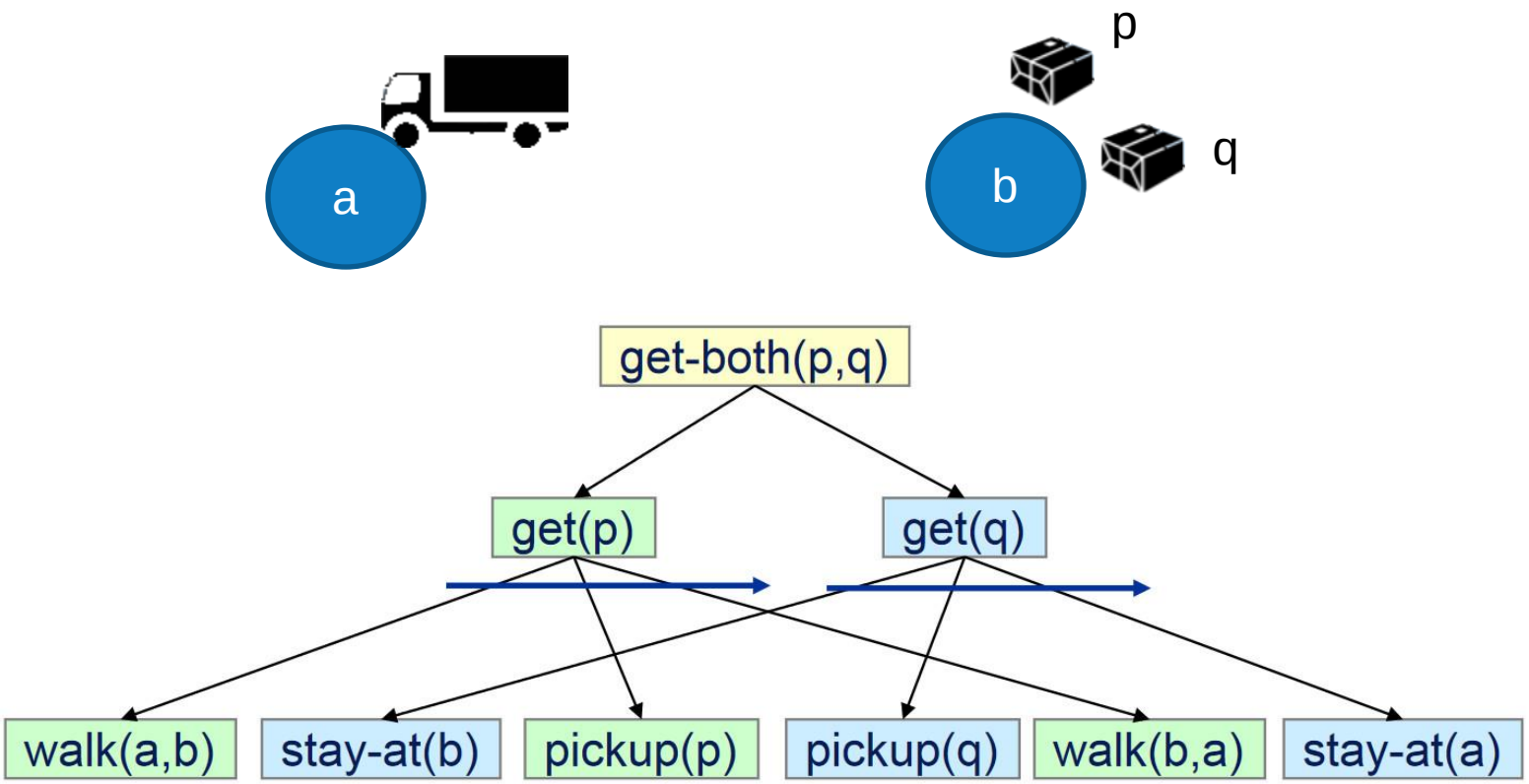


Planificación en HTN

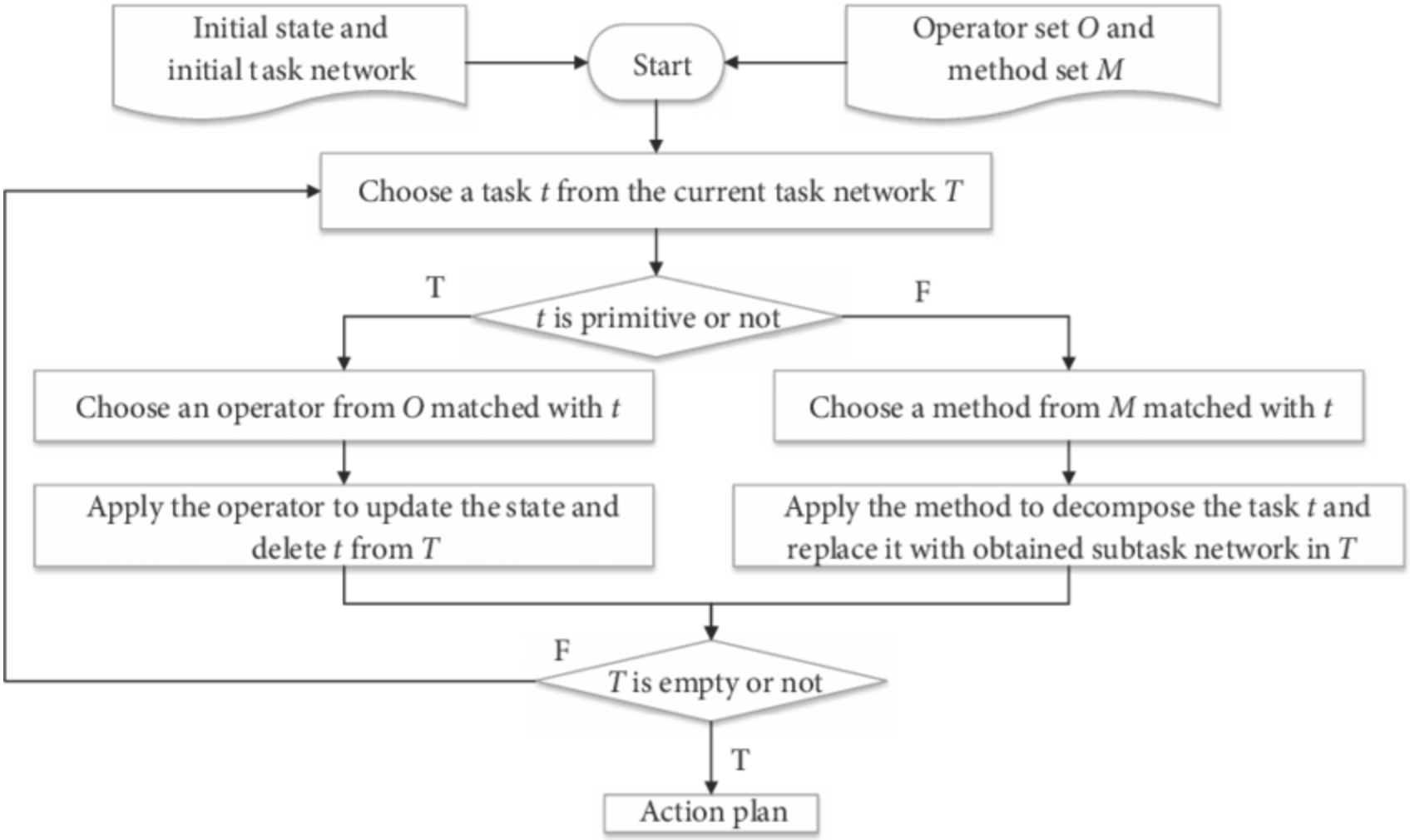
Planificación en HTN

- Una red de tareas representa una jerarquía de tareas, cada una de las cuales puede ejecutarse, si la tarea es primitiva, o ser descompuesta en subtareas refinadas.
- El proceso de planificación procede descomponiendo una red del paso anterior hasta que se descompongan todas las tareas compuestas, es decir, se encuentre una solución.
- La solución es un plan que equivale a un conjunto de tareas primitivas aplicables al estado mundial inicial.
- Se pueden aplicar técnicas de planificación de orden total (TOP) o parcial (POP). La segunda permite solapar subtareas de dos tareas en el tiempo por lo que suele preferirse.

Utilidad de POP en HTN

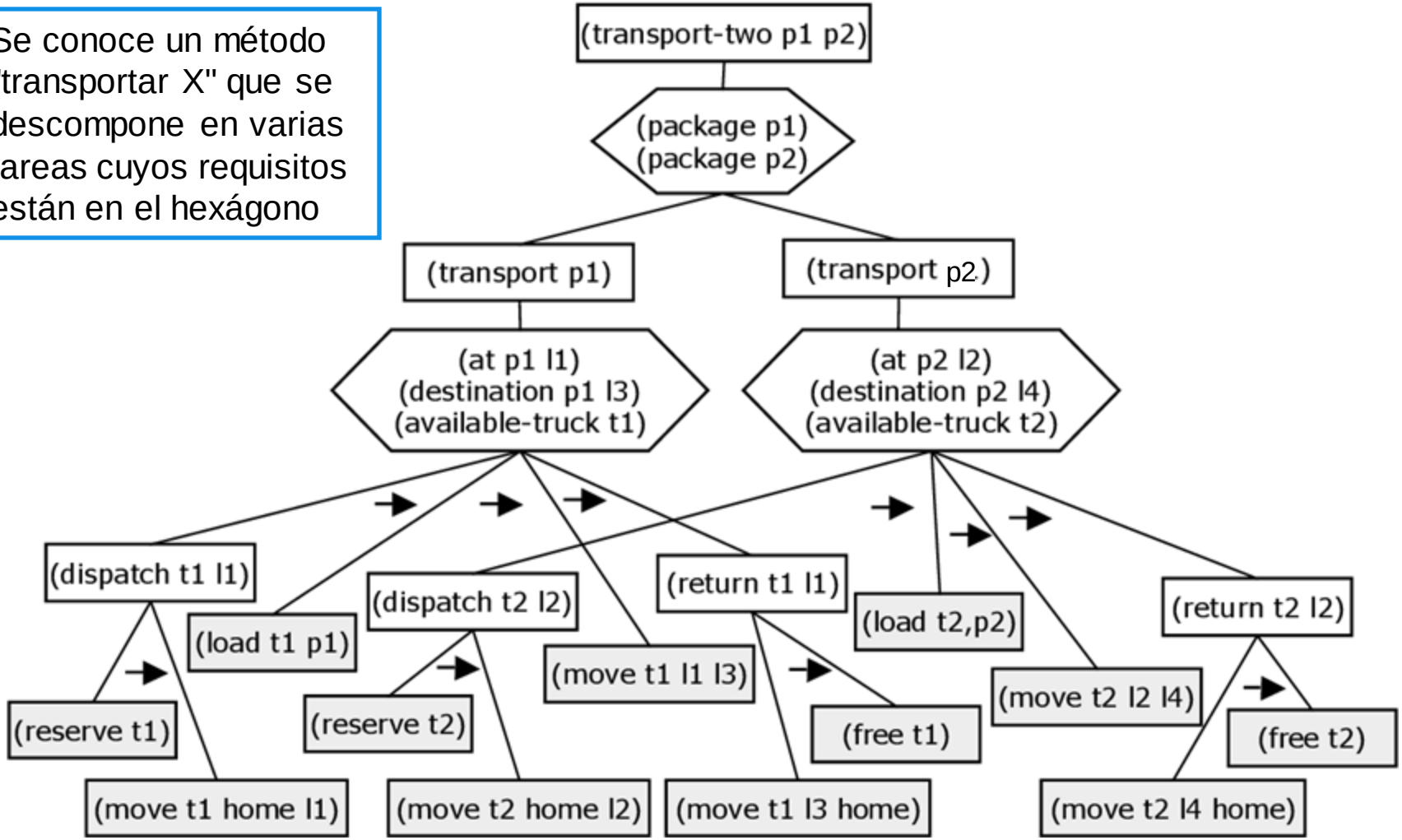


Algoritmo de planificación en HTN



Ejemplo instanciado

Se conoce un método "transportar X" que se descompone en varias tareas cuyos requisitos están en el hexágono



SHOP

La mayor contribución hacia la planificación de HTN ha surgido después de la propuesta del planificador jerárquico simple (SHOP) (Nau D. S., 1999) y sus sucesores.

SHOP es un planificador basado en HTN que muestra un rendimiento eficiente incluso en problemas complejos, pero a costa de proporcionar un conocimiento de dominio bien escrito y posiblemente algorítmico. La disputa sobre si proporcionar mucho conocimiento a un planificador debería considerarse una "trampa" en el mundo de la planificación de la inteligencia artificial sigue vigente (Nau D. S., 1999).

Historia

Los planes se van generando gradualmente de operadores más generales a más concretos:

Estableciendo niveles de abstracción en las precondiciones de los operadores: ABSTRIPS [Sacerdoti, 1974],ALPINE [Knoblock, 1994]

Refinando los operadores sucesivamente:
NOAH [Sacerdoti, 1977], MOLGEN [Stefik, 1981b,Stefik, 1981a]

Preprogramando en qué ese divide cada operador:
O-PLAN [Currie and Tate, 1991], SHOP2 [Nau et al., 2003]

Recursos

- Planificador HTN en Python: PyHOP

  https://bitbucket.org/dananau/pyhop/src/master/

- Estado del arte en HTN

  https://www.sciencedirect.com/science/article/pii/S0004370215000247

- Transparencias adicionales

  https://www.cs.umd.edu/~nau/cmsc722/slides/chapter11.pdf

www.unir.net