An Admissible HTN Planning Heuristic

Pascal Bercher and Gregor Behnke and Daniel Höller and Susanne Biundo

Institute of Artificial Intelligence

August, 23th, 2017

ulm university universität uulm

We are concerned with solving hierarchical planning problems.

- How to solve these problems based on well-informed domain-independent heuristics?
- How to provide optimality guarantees?

We are concerned with solving hierarchical planning problems.

- How to solve these problems based on well-informed domain-independent heuristics?
- How to provide optimality guarantees?

We base our formalization upon *hybrid planning*, which fuses

- hierarchical task network (HTN) planning with
- partial order causal link (POCL) planning.

Related Work:

Related Work:

-

Related Work:

-
- HTN planners using control knowledge (e.g., SHOP2).

Related Work:

-
- HTN planners using control knowledge (e.g., SHOP2).
- Hierarchical heuristic search planners for different problem classes (e.g., hybrid, HGN, GTN, HTN+preferences).

A hybrid planning problem is given in terms of:

- a planning domain stating the tasks (primitive and abstract) and methods (how to refine an abstract action?), and
- an initial partial plan, which needs to be refined into a solution.

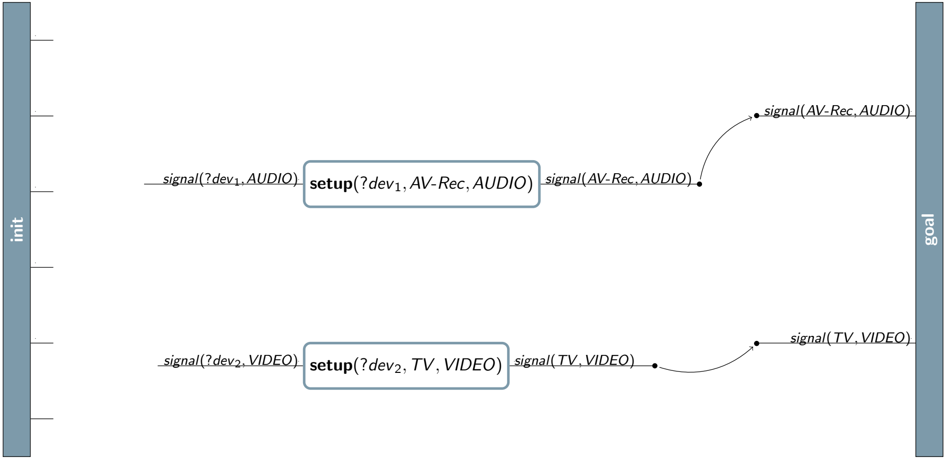
A hybrid planning problem is given in terms of:

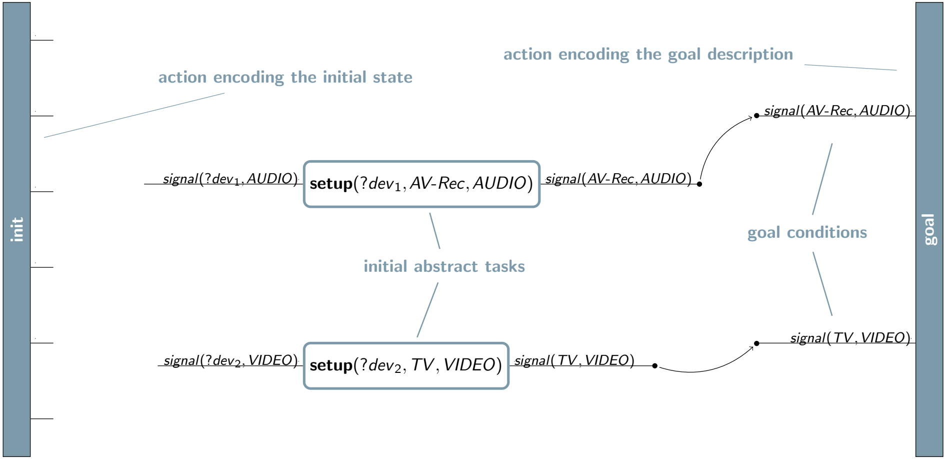
- a planning domain stating the tasks (primitive and abstract) and methods (how to refine an abstract action?), and
- an initial partial plan, which needs to be refined into a solution.

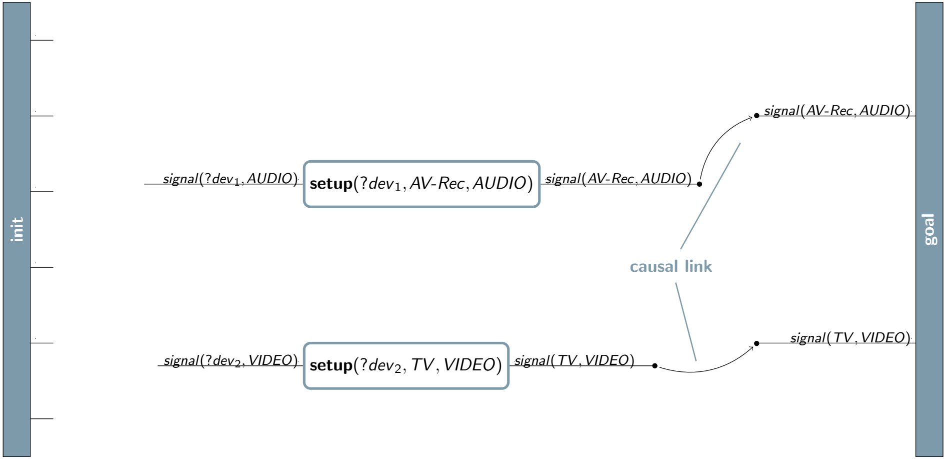
A solution is a partially ordered plan, such that:

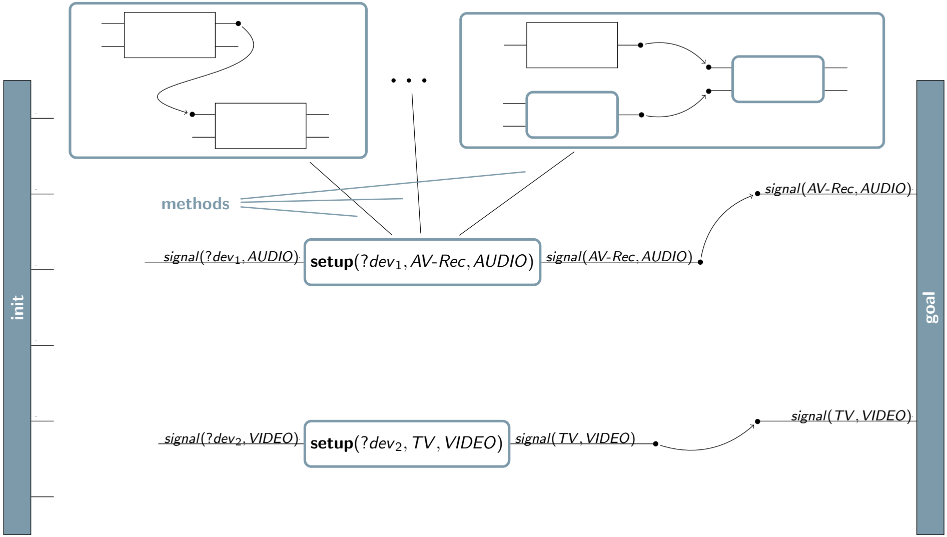
- it is a primitive refinement of the initial partial plan,
- all its linearizations are executable in the initial state, and
- all its linearization satisfy the goal description.

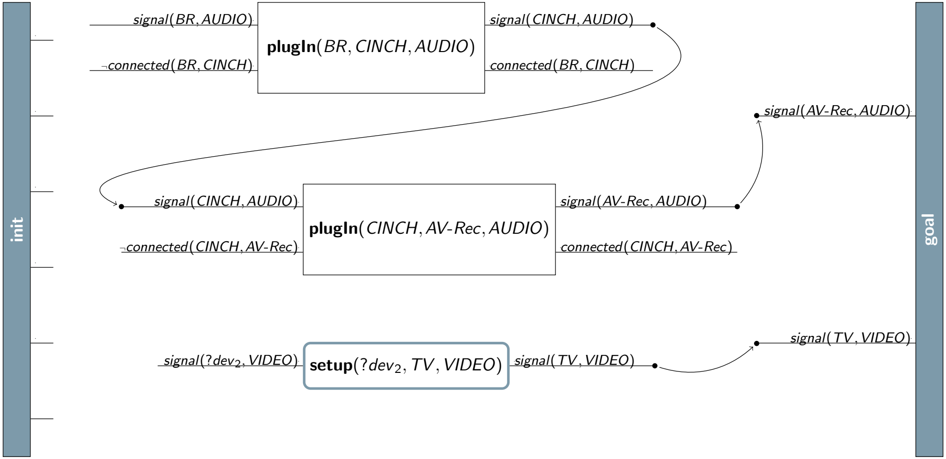
$\rightsquigarrow$ *see example given next.*

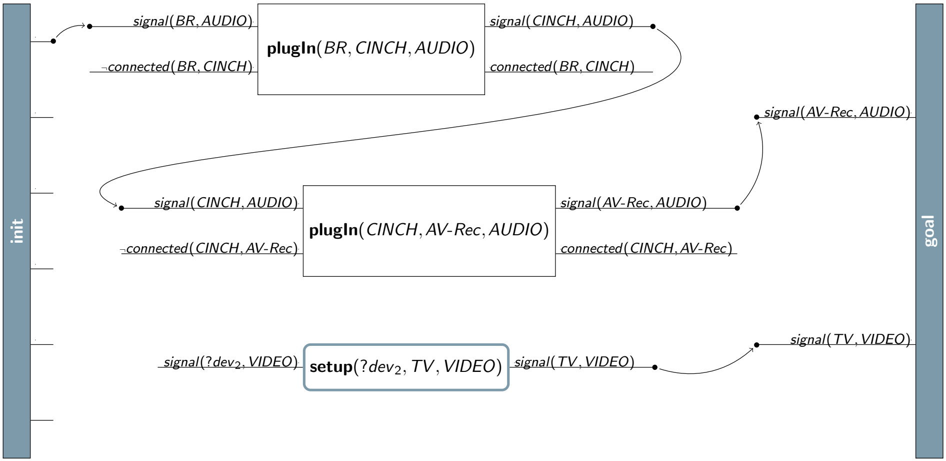

init
goal
signal(?dev$_1$, AUDIO)
setup(?dev$_1$, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
signal(?dev$_2$, VIDEO)
setup(?dev$_2$, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)

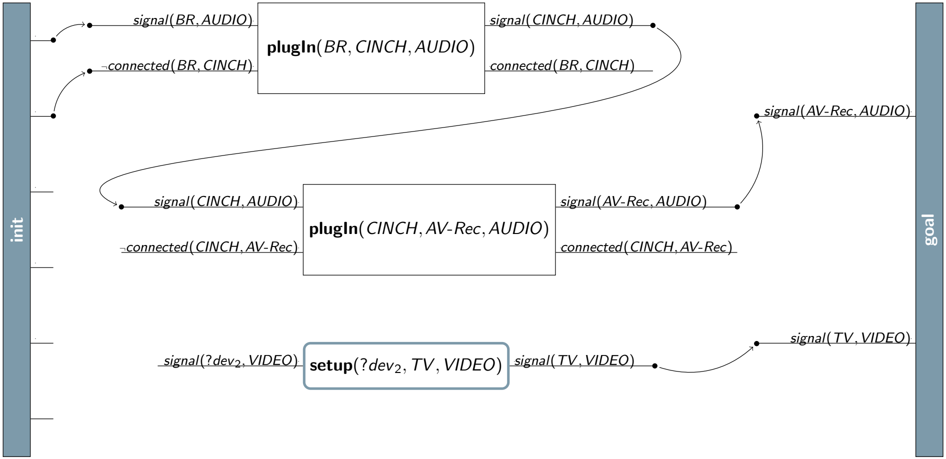

init
goal
action encoding the initial state
action encoding the goal description
initial abstract tasks
goal conditions
signal(?dev1, AUDIO)
setup(?dev1, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)

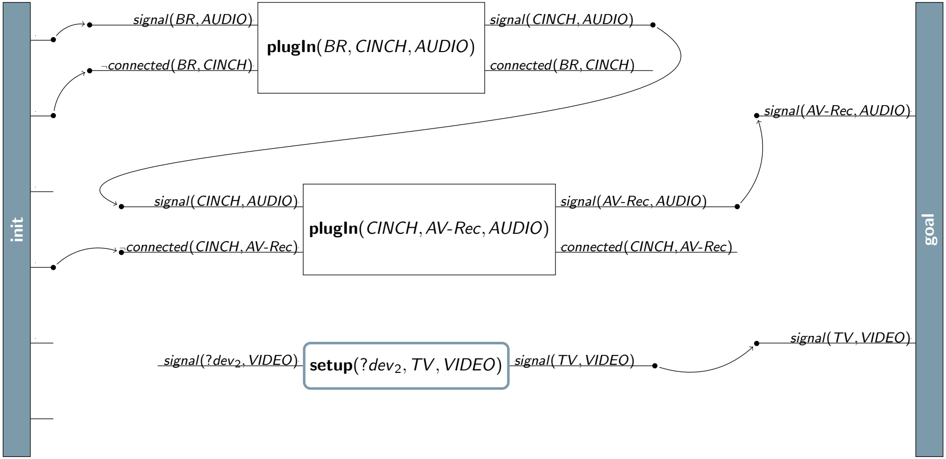

init
goal
signal(?dev₁, AUDIO)
setup(?dev₁, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
signal(?dev₂, VIDEO)
setup(?dev₂, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)
causal link

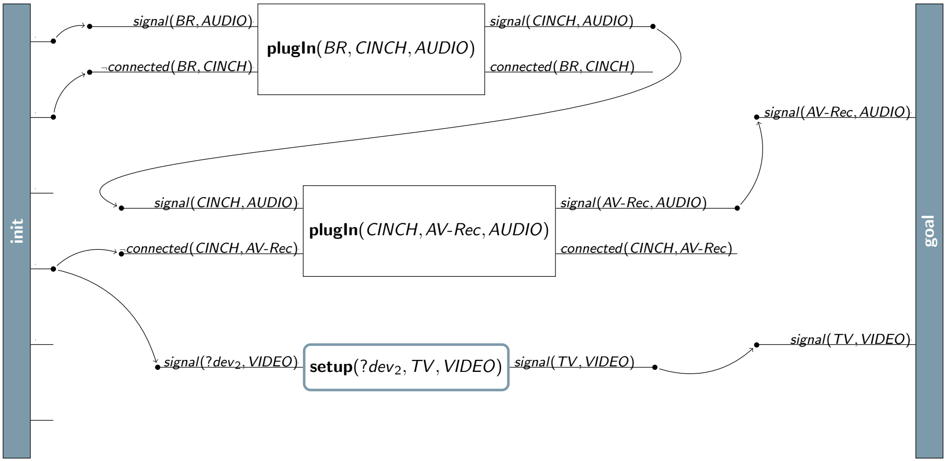

methods
init
goal
signal(?dev1, AUDIO)
setup(?dev1, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)
. . .

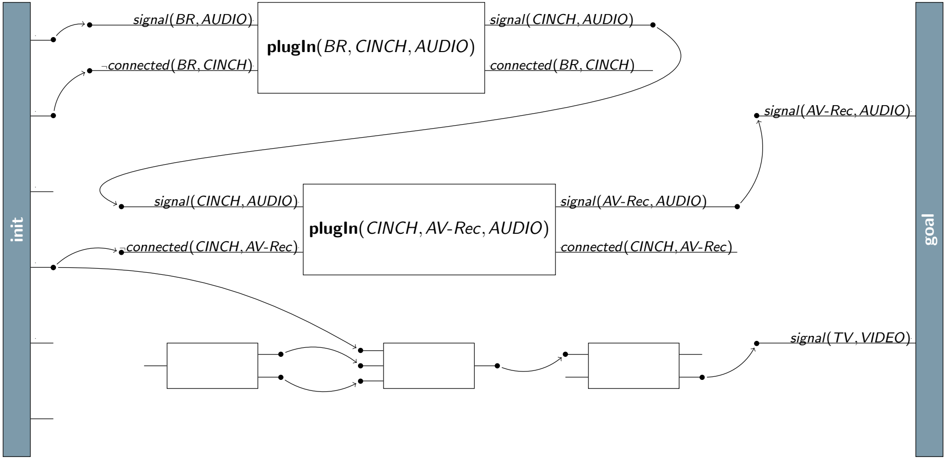

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(CINCH, AUDIO)
¬connected(CINCH, AV-Rec)
plugIn(CINCH, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
connected(CINCH, AV-Rec)
signal(AV-Rec, AUDIO)
signal(?dev₂, VIDEO)
setup(?dev₂, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)
goal

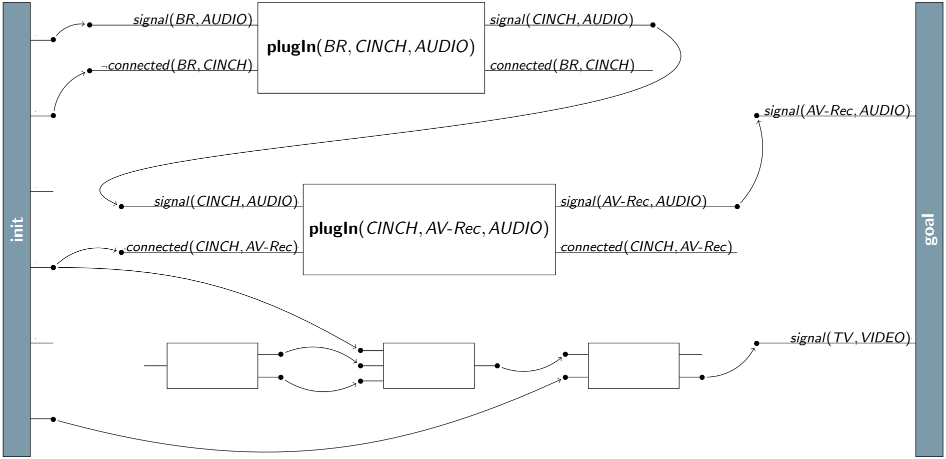

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(AV-Rec, AUDIO)
signal(CINCH, AUDIO)
¬connected(CINCH, AV-Rec)
plugIn(CINCH, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
connected(CINCH, AV-Rec)
signal(TV, VIDEO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
goal

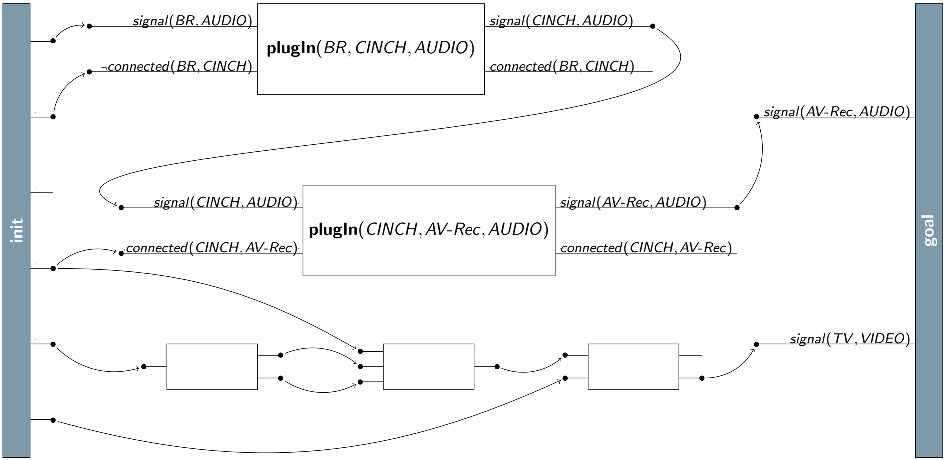

init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(CINCH, AUDIO)
¬connected(CINCH, AV-Rec)
plugIn(CINCH, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
connected(CINCH, AV-Rec)
signal(AV-Rec, AUDIO)
signal(?dev₂, VIDEO)
setup(?dev₂, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)
goal


init
goal
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(CINCH, AUDIO)
¬connected(CINCH, AV-Rec)
plugIn(CINCH, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
connected(CINCH, AV-Rec)
signal(AV-Rec, AUDIO)
signal(?dev2, VIDEO)
setup(?dev2, TV, VIDEO)
signal(TV, VIDEO)
signal(TV, VIDEO)


init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(CINCH, AUDIO)
¬connected(CINCH, AV-Rec)
plugIn(CINCH, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
connected(CINCH, AV-Rec)
signal(?dev₂, VIDEO)
setup(?dev₂, TV, VIDEO)
signal(TV, VIDEO)
signal(AV-Rec, AUDIO)
signal(TV, VIDEO)
goal


init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(CINCH, AUDIO)
¬connected(CINCH, AV-Rec)
plugIn(CINCH, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
connected(CINCH, AV-Rec)
signal(AV-Rec, AUDIO)
signal(TV, VIDEO)
goal


init
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(CINCH, AUDIO)
¬connected(CINCH, AV-Rec)
plugIn(CINCH, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
connected(CINCH, AV-Rec)
signal(AV-Rec, AUDIO)
signal(TV, VIDEO)
goal


init
goal
signal(BR, AUDIO)
¬connected(BR, CINCH)
plugIn(BR, CINCH, AUDIO)
signal(CINCH, AUDIO)
connected(BR, CINCH)
signal(CINCH, AUDIO)
¬connected(CINCH, AV-Rec)
plugIn(CINCH, AV-Rec, AUDIO)
signal(AV-Rec, AUDIO)
connected(CINCH, AV-Rec)
signal(AV-Rec, AUDIO)
signal(TV, VIDEO)

How to come up with heuristic functions?

- Both HTN and hybrid problems are *undecidable*, we hence need severe problem relaxations that them *decidable* and *tractable*.
- $\rightsquigarrow$ We allow task insertion.
- $\rightsquigarrow$ We make the problem acyclic.

How to come up with heuristic functions?

- Both HTN and hybrid problems are *undecidable*, we hence need severe problem relaxations that them *decidable* and *tractable*.
- $\leadsto$ We allow task insertion.
- $\leadsto$ We make the problem acyclic.

How to come up with heuristic functions?

- Both HTN and hybrid problems are *undecidable*, we hence need severe problem relaxations that them *decidable* and *tractable*.
- $\rightsquigarrow$ We allow task insertion.
- $\rightsquigarrow$ We make the problem acyclic.

How does the heuristic work?

Step 1:

- Build a so-called task-decomposition graph (TDG), which represents the hierarchical problem structure.

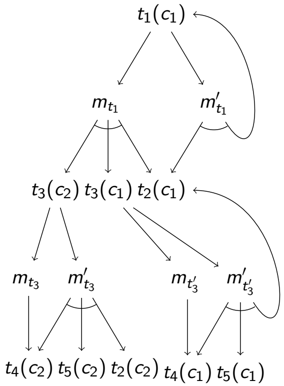
Step 2:

- Calculate heuristic estimates for all tasks in that TDG.

Step 3:

- For a given plan, retrieve the TDG's estimates for all abstract tasks in that plan.

Task decomposition Graph (TDG), example:



A TDG is a ground representation of the task hierarchy.

It is a (possibly cyclic) bipartite graph $\langle V_T, V_M, E_{T\to M}, E_{M\to T}\rangle$ consisting of:

- task vertices V_T (abstract task vertices are OR nodes)
- method vertices V_M (which are AND nods)

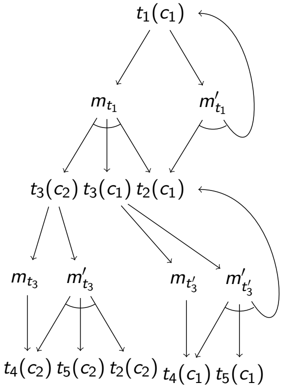
Let $\langle V_T, V_M, E_{T\to M}, E_{M\to T}\rangle$ be a ground TDG. Then, we can calculate a heuristics by exploiting its AND/OR structure:

$$h_T(v_t) := \begin{cases} cost(v_t) & \text{if } v_t \text{ is primitive} \\ \min\limits_{(v_t, v_m)\in E_{T\to M}} h_M(v_m) & \text{else} \end{cases}$$

For a method vertex $v_m = \langle PS, \prec, CL, VC\rangle$, we set:

$$h_M(v_m) := \sum_{(v_m, v_t)\in E_{M\to T}} h_T(v_t)$$

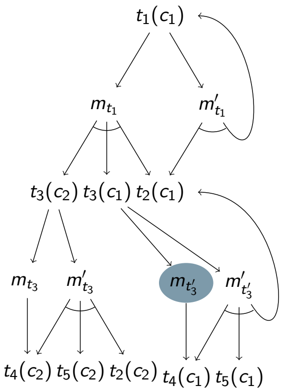
Heuristic computation, example:


$t_1(c_1)$
m_{t_1}
m'_{t_1}
$t_3(c_2)$ $t_3(c_1)$ $t_2(c_1)$
m_{t_3}
m'_{t_3}
$m_{t'_3}$
$m'_{t'_3}$
$t_4(c_2)$ $t_5(c_2)$ $t_2(c_2)$ $t_4(c_1)$ $t_5(c_1)$

$$\begin{aligned} h(m'_{t_3}) &= cost(t_4(c_1)) \\ h(m'_{t'_3}) &= cost(t_4(c_1)) \\ &\quad + cost(t_5(c_1)) \\ &\quad + cost(t_2(c_1)) \\ h(t_3(c_1)) &= min\{h(m'_{t_3}), h(m'_{t'_3})\} \end{aligned}$$

$\ldots$

Heuristic computation, example:



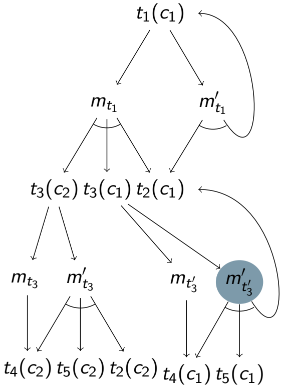
$$h(m'_{t_3}) = cost(t_4(c_1))$$

$$h(m'_{t'_3}) = cost(t_4(c_1)) + cost(t_5(c_1)) + cost(t_2(c_1))$$

$$h(t_3(c_1)) = min\{h(m'_{t_3}), h(m'_{t'_3})\}$$

$$\dots$$

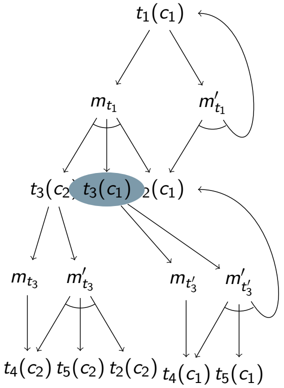
Heuristic computation, example:


$t_1(c_1)$
m_{t_1}
m'_{t_1}
$t_3(c_2)$ $t_3(c_1)$ $t_2(c_1)$
m_{t_3}
m'_{t_3}
$m_{t'_3}$
$m'_{t'_3}$
$t_4(c_2)$ $t_5(c_2)$ $t_2(c_2)$ $t_4(c_1)$ $t_5(c_1)$

$$
\begin{aligned}
h(m'_{t_3}) &= cost(t_4(c_1)) \\
h(m'_{t'_3}) &= cost(t_4(c_1)) \\
&\quad + cost(t_5(c_1)) \\
&\quad + cost(t_2(c_1)) \\
h(t_3(c_1)) &= min\{h(m'_{t_3}), h(m'_{t'_3})\}
\end{aligned}
$$

...

Heuristic computation, example:



$$\begin{aligned} h(m'_{t_3}) &= cost(t_4(c_1)) \\ h(m'_{t'_3}) &= cost(t_4(c_1)) \\ &\quad + cost(t_5(c_1)) \\ &\quad + cost(t_2(c_1)) \\ h(t_3(c_1)) &= min\{h(m'_{t_3}), h(m'_{t'_3})\} \\ &\quad \ldots \end{aligned}$$

Improving Heuristic estimates:

- So far, the TDG is computed only *once*.
  $\rightsquigarrow$ It is a preprocessing heuristic.
- Re-computing the TDG during planning is expensive, but increases heuristic estimates. (We show in which situations TDG recomputation might improve these estimates to avoid unnecessary recomputations).

Space and time increase and decrease due to TGD recomputation:

Qualitative:

Strategy		space			time			space			time			space			time			space			time		
		<	=	>	<	=	>	<	=	>	<	=	>	<	=	>	<	=	>	<	=	>	<	=	>
		UM-Translog						SmartPhone						Satellite						Woodworking					
A^*	TDG_m	2	19	0	1	15	5	1	4	0	1	4	0	22	0	0	0	17	5	7	2	0	1	6	2
	TDG_c	2	19	0	0	13	8	1	4	0	0	4	1	18	0	0	0	10	8	6	2	0	4	3	1
A^*_2	TDG_m	2	19	0	3	16	2	1	4	0	0	4	1	22	0	0	0	13	9	4	5	0	1	6	2
	TDG_c	2	19	0	4	11	6	1	4	0	1	4	0	20	0	0	0	11	9	5	5	0	4	3	3

Quantitative:

Best results (for Woodworking):
TDG_m: 83% search space reduction with 45% runtime reduction
TDG_c: 92% search space reduction with 40% runtime reduction

Worst results (for Satellite):
TDG_c: 5.4% search space reduction, but 100% runtime *increase*

	Strategy	UM-Tr. (21 inst.)			SmartPh. (5 inst.)			Satellite (22 inst.)			Woodw. (11 inst.)			Summary (59 inst.)		
		#s	#o	cost	#s	#o	cost	#s	#o	cost	#s	#o	cost	#s	#o	cost
blind	Uniform	21	21	1.00	4	4	1.00	17	17	1.00	8	8	1.00	50	50	**1.00**
	BF	21	21	1.00	4	4	1.00	15	15	1.00	7	7	1.00	47	47	**1.00**
	DF	21	21	1.00	5	1	1.60	19	7	2.09	8	4	1.44	53	33	2.09
systems	UMCP_{BF}	21	21	1.00	4	4	1.00	15	15	1.00	7	7	1.00	47	47	**1.00**
	UMCP_{DF}	21	21	1.00	4	1	1.60	17	6	2.09	6	4	1.29	48	32	2.09
	UMCP_h	21	21	1.00	5	4	1.40	19	11	1.50	7	7	1.00	52	43	1.50
	Compile	18	18	1.00	5	5	1.00	21	18	1.10	5	5	1.00	49	46	1.10
	Compile_{opt}	16	16	1.00	5	5	1.00	9	9	1.00	5	5	1.00	35	35	**1.00**
A^*	ADD	21	21	1.00	4	1	1.20	21	21	1.00	10	9	1.17	56	52	1.20
	ADD-r	21	21	1.00	5	5	1.00	19	18	1.08	9	4	1.25	54	48	1.25
	Relax	21	21	1.00	5	5	1.00	18	18	1.00	10	8	1.17	54	52	1.17
	OC	21	21	1.00	4	4	1.00	21	21	1.00	10	7	1.17	56	53	1.17
	TDG_m/-rec	21	21	1.00	5	5	1.00	22	21	1.31	9	9	1.00	**57**	56	1.31
	TDG_c/-rec	21	21	1.00	5	5	1.00	18	18	1.00	8	8	1.00	52	52	**1.00**
Greedy-A^*	ADD	21	21	1.00	4	0	1.20	21	20	1.09	10	9	1.17	56	50	1.20
	ADD-r	21	21	1.00	5	5	1.00	20	17	1.10	10	4	1.25	56	47	1.25
	Relax	21	21	1.00	5	5	1.00	18	15	1.10	10	4	1.25	54	45	1.25
	OC	21	21	1.00	4	4	1.00	22	21	1.09	10	7	1.22	**57**	53	1.22
	TDG_m/-rec	21	21	1.00	5	5	1.00	22	17	1.31	9	8	1.08	**57**	51	1.31
	TDG_c	21	21	1.00	5	5	1.00	20	20	1.00	10	10	1.00	56	56	**1.00**
	TDG_c-rec	21	21	1.00	5	5	1.00	20	20	1.00	11	11	1.00	**57**	**57**	**1.00**

Summary:

- Introduced the first admissible heuristic for standard HTN and hybrid planning.
- The proposed heuristic(s) perform best both in terms of coverage and plan quality.

Also in the paper and poster:

- A variant of the heuristic tailored to hybrid planning systems.
- Investigation of TDG recomputation to improve heuristic accuracy.