Improving Hierarchical Planning Performance by the Use of Landmarks

Mohamed Elkawkagy, Pascal Bercher, Bernd Schattenberg, and Susanne Biundo

Institute of Artificial Intelligence, Ulm University, Germany

AAAI-12: Twenty-Sixth Conference on Artificial Intelligence, 22-26 July 2012, Toronto, Canada

Abstract

We present novel domain-independent planning strategies based on hierarchical landmarks.

- We ran our evaluations on four distinguished benchmark domains. These domains are divided into two categories:
  - Domains with a deep expansion hierarchy such as Um-Translog and SmartPhone, and
  - Domains with shallow expansion hierarchy such as Satellite and WoodWorking.
- Our empirical evaluation shows that our landmark strategies outperform established search strategies.

Planning Framework

The introduced domain-independent planning strategies are used by our **Hierarchical Task Network (HTN) Planning** formalism.

Planning Problem

An HTN planning problem is a 3-tuple $\Pi = \langle D, S_{init}, P_{init}\rangle$

- $D = \langle T, M\rangle$ is a domain model, where T and M denote finite sets of tasks *(abstract and primitive)* and methods.
- S_{init} is an initial state.
- P_{init} is an initial plan. A plan $P = \langle S, C\rangle$ consists of a set S of plan steps and a set C of constraints such as ordering constraints and causal link constraints.

Note that the hierarchy abstraction is achieved through the methods M.
A method is a pair $\langle t(\tau), P\rangle$

- $t(\tau)$ is the abstract task, and
- P is the plan to achieve the task $t(\tau)$.

Solution Plan

A plan $P = \langle S, C\rangle$ is a solution to Π iff:

- P is a successor of the initial plan P_{init} in the induced search space.
- P contains only primitive plan steps, is executable in S_{init} and has consistent constraint sets.

Algorithm 1: *Standard Refinement Algorithm*

Input : The sequence $\texttt{Fringe} = \langle P_{\text{init}}\rangle$.
Output : A solution or $\texttt{Fail}$.

while $\texttt{Fringe} = \langle P_1 \ldots P_n\rangle \neq \varepsilon$ **do**
  $F \leftarrow f^{\text{FlawDet}}(P_1)$
  if $F = \emptyset$ **then return** P_1
  $\langle \texttt{m}_1 \ldots \texttt{m}_k\rangle \leftarrow f^{\text{ModOrd}}\left(\bigcup_{\texttt{f}\in F} f^{\text{ModGen}}(\texttt{f})\right)$
  $\texttt{succ} \leftarrow \langle \texttt{app}(\texttt{m}_1, P_1) \ldots \texttt{app}(\texttt{m}_k, P_1)\rangle$
  $\texttt{Fringe} \leftarrow f^{\text{PlanOrd}}(\texttt{succ} \circ \langle P_2 \ldots P_n\rangle)$
return $\texttt{fail}$

In our algorithm, the search strategy is a combination of the plan modification and plan ordering functions.
For example, in order to perform a depth first search, the plan ordering is the identity function ($f^{\text{PlanOrd}}(\bar{P}) = \bar{P}$ for any sequence of P).

- The plan ordering function f^{PlanOrd} orders the updated search-space.
- The modification ordering function f^{ModOrd} determines which branch of the search space to visit first.

Landmarks

In HTN Planning, landmarks are tasks that occur in the plan sequence leading from a problem's initial plan P_{init} to any solution.
The information about landmarks is stored in a so-called Landmark Table.
Each landmark table entry is a 3-tuple

$$LT = \langle t(\tau),\ M(t(\tau)),\ O(t(\tau))\rangle$$

- $t(\tau)$ is an abstract task,
- $M(t(\tau))$ are its mandatory tasks (tasks, which occur in all methods of $t(\tau)$), and
- $O(t(\tau))$ are the optional tasks (for each method, there is a set containing the remaining tasks).

Landmark extraction is done using a so-called task decomposition graph (TDG) of Π.
A TDG is a relaxed representation of how the initial plan P_{init} of a planning problem Π can be decomposed (cf. Figure 1).

Example

Let $\Pi = \langle D, S_{init}, P_{init}\rangle$ an HTN planning problem with $D = \langle \{t_1(\tau_1), \ldots, t_5(\tau_5)\}, \{m_a, m'_a, m_b, m'_b\}\rangle$, $P_{\text{init}} = \langle\{l_1{:}t_1(\tau_1)\}, \{\tau_1{=}c_1\}\rangle$, and constants c_1 and c_2, where:

$$m_a := \langle t_1(\tau_1), \langle\{l_1{:}t_3(\tau_1), l_2{:}t_3(\tau_2), l_3{:}t_2(\tau_1)\}, \{\tau_1{\neq}\tau_2\}\rangle\rangle$$
$$m'_a := \langle t_1(\tau_1), \langle\{l_4{:}t_2(\tau_1), l_5{:}t_1(\tau_1)\}, \emptyset\rangle\rangle$$
$$m_b := \langle t_3(\tau_1), \langle\{l_6{:}t_4(\tau_1), l_7{:}t_5(\tau_1)\}, \emptyset\rangle\rangle$$
$$m'_b := \langle t_3(\tau_1), \langle\{l_8{:}t_4(\tau_1)\}, \emptyset\rangle\rangle$$

The TDG for Π is given in Figure 1.

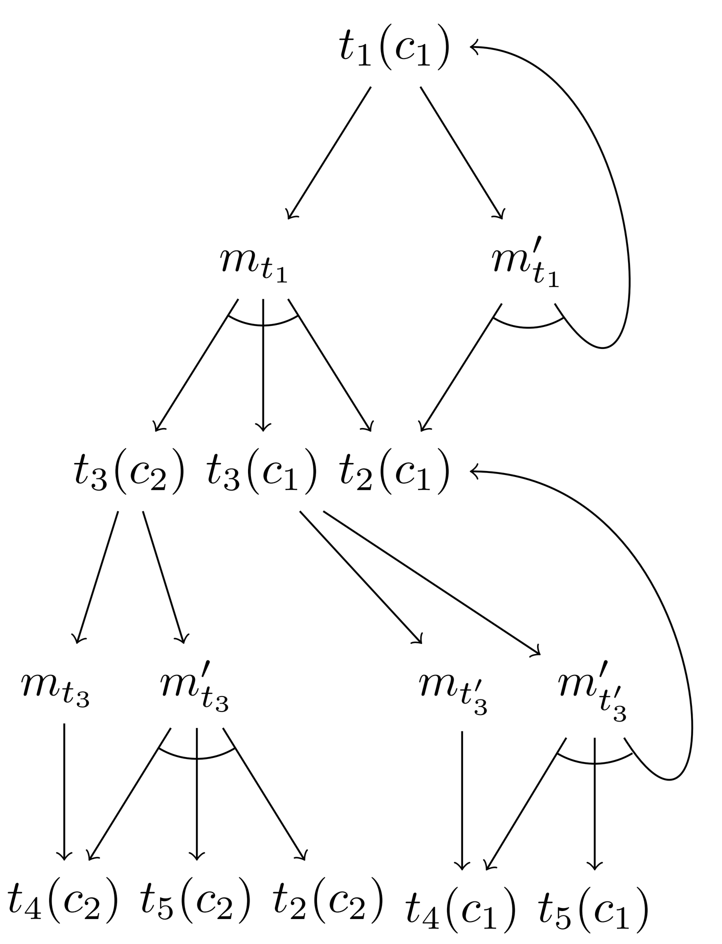

Figure 1: *The TDG for the planning problem* Π.

The method vertices are given as follows:
$m_{t_1} = \langle t_1(c_1), m_a|_{\tau_1=c_1,\tau_2=c_2}\rangle$, $m'_{t_1} = \langle t_1(c_1), m'_a|_{\tau_1=c_1}\rangle$,
$m_{t_3} = \langle t_3(c_2), m_b|_{\tau_1=c_2}\rangle$, $m'_{t_3} = \langle t_3(c_2), m'_b|_{\tau_1=c_2}\rangle$,
$m_{t'_3} = \langle t_3(c_1), m_b|_{\tau_1=c_1}\rangle$, $m'_{t'_3} = \langle t_3(c_1), m'_b|_{\tau_1=c_2}\rangle$

The according landmark table is given as follows:

Abs. Task	Mandatory	Optional
$t_1(c_1)$	$\{t_2(c_1)\}$	$\{\{t_3(c_2), t_3(c_1)\}, \{t_1(c_1)\}\}$
$t_3(c_2)$	$\{t_4(c_2)\}$	$\{\emptyset, \{t_5(c_2), t_2(c_2)\}\}$
$t_3(c_1)$	$\{t_4(c_1)\}$	$\{\emptyset, \{t_5(c_1), t_2(c_1)\}\}$

Landmark-Aware Strategies

Our strategies solely operate on the optional tasks.

Definition 1 (Landmark Cardinality)**.** *Given a landmark table LT, we define the* landmark cardinality *of a set of tasks $o = \{t_1(\overline{\tau}_1), \ldots, t_n(\overline{\tau}_n)\}$ to be*

$$|o|_{LT} := |\{t(\overline{\tau}) \in o \mid \langle t(\overline{\tau}),\ M(t(\overline{\tau})),\ O(t(\overline{\tau}))\rangle \in LT\}|$$

Definition 2 (Closure of the Optional Set)**.** *The* closure of the optional set *for a given ground task $t(\overline{\tau})$ and a landmark table LT is the smallest set $O^*(t(\overline{\tau}))$, such that $O^*(t(\overline{\tau})) := \emptyset$ for primitive $t(\overline{\tau})$ and, otherwise:*

$$O^*(t(\overline{\tau})) := O(t(\overline{\tau})) \cup \bigcup_{o\in O(t(\overline{\tau}))} \left(\bigcup_{t'(\overline{\tau}')\in o} O^*(t'(\overline{\tau}'))\right)$$

with $\langle t(\overline{\tau}),\ M(t(\overline{\tau})),\ O(t(\overline{\tau}))\rangle \in LT$

Definition 3 (Landmark Strategies)**.** *Let $P = \langle S, C\rangle$ be a plan and $t_i(\overline{\tau}_i)$ and $t_j(\overline{\tau}_j)$ be ground instances of two abstract tasks in S that are referenced by two abstract task flaws $\texttt{f}_i$ and $\texttt{f}_j$, respectively, that are found in P.*
Let a given landmark table LT contain the corresponding entries

$\langle t_i(\overline{\tau}_i),\ M(t_i(\overline{\tau}_i)),\ O(t_i(\overline{\tau}_i))\rangle$ *and*
$\langle t_j(\overline{\tau}_j),\ M(t_j(\overline{\tau}_j)),\ O(t_j(\overline{\tau}_j))\rangle$

Then, the given modification ordering function orders a plan modification $\texttt{m}_i$ before $\texttt{m}_j$ if and only if $\texttt{m}_i$ addresses $\texttt{f}_i$, $\texttt{m}_j$ addresses $\texttt{f}_j$, and one of the four criteria hold:

$$\text{lm}_1 : \sum_{o\in O(t_i(\overline{\tau}_i))} |o|_{LT} < \sum_{o\in O(t_j(\overline{\tau}_j))} |o|_{LT}$$

$$\text{lm}_1^* : \sum_{o\in O^*(t_i(\overline{\tau}_i))} |o|_{LT} < \sum_{o\in O^*(t_j(\overline{\tau}_j))} |o|_{LT}$$

$$\text{lm}_2 : \sum_{o\in O(t_i(\overline{\tau}_i))} |o| < \sum_{o\in O(t_j(\overline{\tau}_j))} |o|$$

$$\text{lm}_2^* : \sum_{o\in O^*(t_i(\overline{\tau}_i))} |o| < \sum_{o\in O^*(t_j(\overline{\tau}_j))} |o|$$

Example

Let a plan P contain two abstract tasks $t_1(c_1)$ and $t_3(c_2)$.
Let the landmark table contain:
$\langle t_1(c_1)\ ,\ \{t_2(c_1)\}\ ,\ \{\{t_3(c_2), t_3(c_1)\}\ , \{t_1(c_1)\}\}\rangle$
$\langle t_3(c_1)\ ,\ \{t_4(c_1)\}\ ,\ \{\emptyset, \{t_5(c_1), t_2(c_1)\}\}\rangle$
$\langle t_3(c_2)\ ,\ \{t_4(c_2)\}\ ,\ \{\emptyset, \{t_5(c_2), t_2(c_2)\}\}\rangle$

$$\text{lm}_1 : \sum_{o\in O(t_3(c_2))} |o|_{LT} < \sum_{o\in O(t_1(c_1))} |o|_{LT} \iff 0 < 2+1$$

$$\text{lm}_1^* : \sum_{o\in O^*(t_3(c_2))} |o|_{LT} < \sum_{o\in O^*(t_1(c_1))} |o|_{LT} \iff 0 < 3+0+1$$

Evaluation

- Our novel landmark strategies are compared with standard HTN strategies.
- Our refinement algorithm (cf. Algorithm 1) can simulate behavior of any system when using the according modification f^{ModOrd} and plan ordering f^{PlanOrd} functions.

UM-Translog Domain.

Mod. ordering function f^{ModOrd}	#1 org	#1 red	#2 org	#2 red	#3 org	#3 red
UMCP	952	244	994	229	215	127
ems	2056	1048	2199	1806	876	235
SHOP	1735	353	1911	274	911	190
lm_1	243	180	447	184	190	122
lm_1^*	1772	212	370	205	1002	140
lm_2	3311	255	1670	248	925	151
lm_2^*	846	226	991	238	1755	122
lcf	1878	225	3020	209	267	322
da-HotSpot	2414	1958	–	2030	578	352
du-HotSpot	1319	775	987	1090	391	258
HotZone	473	196	498	224	171	137

WoodWorking domain.

Mod. ordering function f^{ModOrd}	#1 org	#1 red	#2 org	#2 red	#3 org	#3 red
UMCP	228	133	259	125	892	218
ems	415	298	–	2457	–	512
SHOP	–	–	–	–	–	3578
lm_1	96	55	171	159	564	197
lm_1^*	82	50	614	98	2109	1245
lm_2	881	433	–	362	–	–
lm_2^*	1359	403	–	367	–	893
lcf	2067	350	–	–	–	–
da-HotSpot	113	85	355	110	–	–
du-HotSpot	–	–	–	–	–	–
HotZone	–	–	–	418	–	–

SmartPhone domain.

Mod. ordering function f^{ModOrd}	#1 org	#1 red	#2 org	#2 red	#3 org	#3 red
UMCP	80	30	256	115	–	–
ems	107	52	235	148	–	–
SHOP	95	73	–	–	–	–
lm_1	50	30	134	53	–	465
lm_1^*	65	50	392	173	–	–
lm_2	60	50	181	53	–	680
lm_2^*	98	76	1632	327	–	697
lcf	63	40	–	159	8455	6827
da-HotSpot	45	43	–	203	1747	1041
du-HotSpot	52	46	638	166	–	3421
HotZone	65	33	490	212	–	–

Satellite domain.

Mod. ordering function f^{ModOrd}	#1 org	#1 red	#2 org	#2 red	#3 org	#3 red
UMCP	91	91	51	41	2035	1336
ems	74	60	62	53	2608	2856
SHOP	66	67	113	111	270	264
lm_1	89	80	209	208	767	652
lm_1^*	86	85	54	43	1024	969
lm_2	132	86	151	140	–	5804
lm_2^*	102	80	191	99	–	–
lcf	95	93	154	77	1551	1338
da-HotSpot	69	67	85	78	2136	1131
du-HotSpot	107	49	270	150	–	–
HotZone	76	64	142	62	–	4764