Finding and Exploiting LTL Trajectory Constraints in Heuristic Search

Salomé Simon   Gabriele Röger

University of Basel, Switzerland

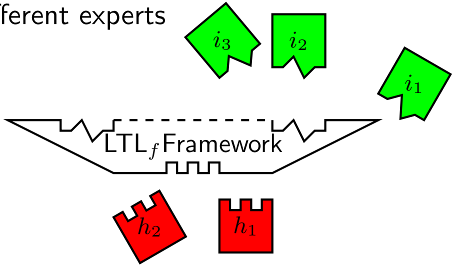
SoCS 2015

Motivation

Goal

framework for describing information about the search space

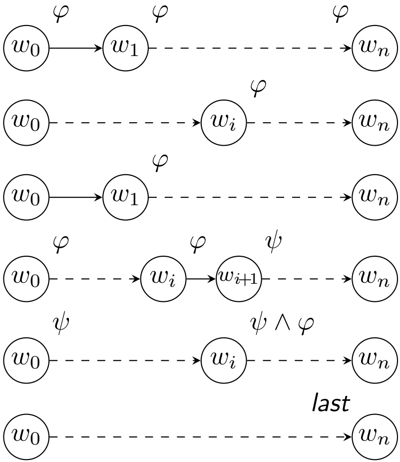
- combining information from different sources
  $\rightsquigarrow$ creating stronger heuristics
- decoupling the derivation and exploitation of information
  $\rightsquigarrow$ split work between different experts



Linear Temporal Logic on Finite Traces (LTL$_f$)

- evaluated over a linear sequence of worlds (= variable assignments)
- extends propositional logic with:

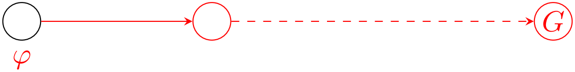
$\Box\varphi$	Always
$\Diamond\varphi$	Eventually
$\bigcirc\varphi$	Next
$\varphi\mathcal{U}\psi$	Until
$\varphi\mathcal{R}\psi$	Release
last	Last world



LTL$_f$ Formulas in the Search Space

variable	$\leftrightarrow$	state variable or action
world	$\leftrightarrow$	node in search space (with incoming action)
world sequence	$\leftrightarrow$	path to a goal node

LTL$_f$ formulas associated to nodes
$\rightarrow$ express conditions all optimal paths to a goal need to fulfill



Feasibility for Nodes

Definition (Feasibility for nodes)

An LTL$_f$ formula φ is feasible for n if for all paths ρ such that

- ρ is applicable in n,
- the application of ρ leads to a goal state, and
- $g(n) + c(\rho) = h^*$

it holds that $\boldsymbol{w}_{\boldsymbol{\rho}}^{\boldsymbol{s}} \models \varphi$.

(where $\boldsymbol{w}_{\boldsymbol{\rho}}^{\boldsymbol{s}} = \langle \{a_1\} \cup s[a_1], \{a_2\} \cup s[\langle a_1, a_2 \rangle], \ldots, \{a_n\} \cup s[\rho], s[\rho] \rangle$)

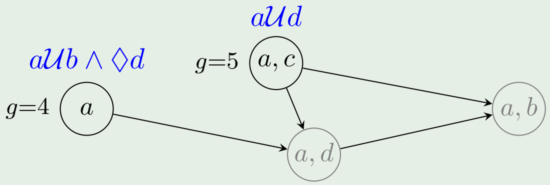
Adding and Propagating Information during the Search

1. **new information during the search**
   directly added to the corresponding node with conjunction

Adding and Propagating Information during the Search

1. **new information during the search**
   directly added to the corresponding node with conjunction

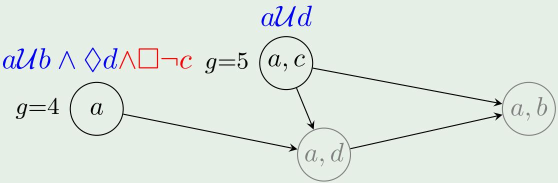
Example



Adding and Propagating Information during the Search

1. **new information during the search**
   directly added to the corresponding node with conjunction

Example



Adding and Propagating Information during the Search

2. **formulas can be propagated with progression to successor nodes**

Theorem

Let φ be a feasible formula for a node n, and let n' be the successor node reached from n with action a. Then progress$(\varphi, \{a\} \cup s(n'))$ is feasible for n'.

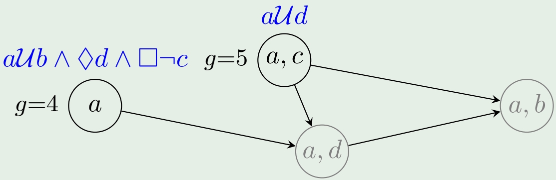
Adding and Propagating Information during the Search

2. **formulas can be propagated with progression to successor nodes**

Theorem

Let φ be a feasible formula for a node n, and let n' be the successor node reached from n with action a. Then progress$(\varphi, \{a\} \cup s(n'))$ is feasible for n'.

Example



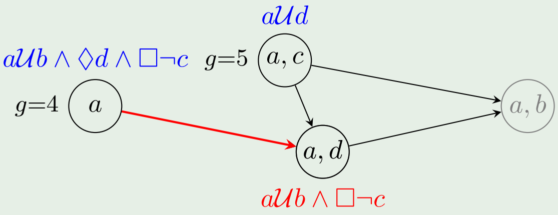
Adding and Propagating Information during the Search

2. **formulas can be propagated with progression to successor nodes**

Theorem

Let φ be a feasible formula for a node n, and let n' be the successor node reached from n with action a. Then progress$(\varphi, \{a\} \cup s(n'))$ is feasible for n'.

Example



Adding and Propagating Information during the Search

3. **duplicate elimination: conjunction of formulas of "cheapest" nodes is feasible**

Theorem

Let n and n' be two search nodes such that $g(n) = g(n')$ and $s(n) = s(n')$. Let further φ_n and $\varphi_{n'}$ be feasible for the respective node. Then $\varphi_n \wedge \varphi_{n'}$ is feasible for both n and n'.

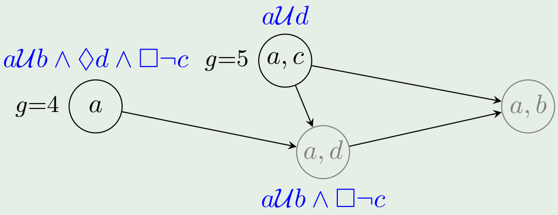
Adding and Propagating Information during the Search

3. **duplicate elimination: conjunction of formulas of “cheapest” nodes is feasible**

Theorem

Let n and n' be two search nodes such that $g(n) = g(n')$ and $s(n) = s(n')$. Let further φ_n and $\varphi_{n'}$ be feasible for the respective node. Then $\varphi_n \land \varphi_{n'}$ is feasible for both n and n'.

Example



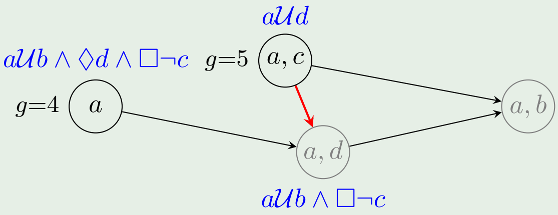
Adding and Propagating Information during the Search

3. **duplicate elimination: conjunction of formulas of "cheapest" nodes is feasible**

Theorem

Let n and n' be two search nodes such that $g(n) = g(n')$ and $s(n) = s(n')$. Let further φ_n and $\varphi_{n'}$ be feasible for the respective node. Then $\varphi_n \wedge \varphi_{n'}$ is feasible for both n and n'.

Example



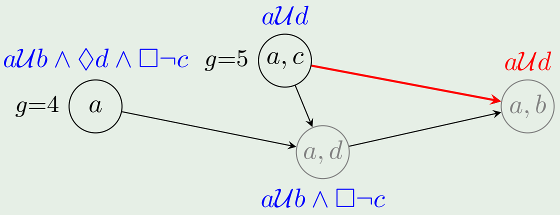
Adding and Propagating Information during the Search

3. **duplicate elimination: conjunction of formulas of "cheapest" nodes is feasible**

Theorem

Let n and n' be two search nodes such that $g(n) = g(n')$ and $s(n) = s(n')$. Let further φ_n and $\varphi_{n'}$ be feasible for the respective node. Then $\varphi_n \wedge \varphi_{n'}$ is feasible for both n and n'.

Example



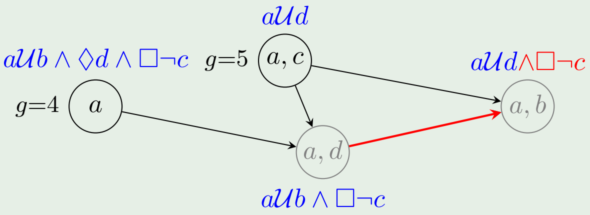
Adding and Propagating Information during the Search

3. **duplicate elimination: conjunction of formulas of "cheapest" nodes is feasible**

Theorem

Let n and n' be two search nodes such that $g(n) = g(n')$ and $s(n) = s(n')$. Let further φ_n and $\varphi_{n'}$ be feasible for the respective node. Then $\varphi_n \wedge \varphi_{n'}$ is feasible for both n and n'.

Example



Encoding Information in LTL_f Formulas

Possible sources of information:

- domain-specific knowledge
- temporally extended goals
- here: information used in specialized heuristics
  - Landmarks and their orderings (Hoffmann et al. 2004, Richter et al. 2008)
  - Unjustified Action Applications (Karpas and Domshlak 2012)

Landmarks

Fact Landmark: A fact that must be true at some point in every solution (Hoffmann et al. 2004)
$\rightarrow$ In LTL$_f$: $\Diamond l$

Further information about landmarks:

- **First achievers**: $l \vee \bigvee_{a \in FA_l} \Diamond a$
- **Required again**: $(\Diamond l)\mathcal{U} l'$
- **Goal**: $\bigwedge_{g \in G} \left((\Diamond g)\mathcal{U} \bigwedge_{g' \in G} g'\right)$

Unjustified Action Applications

If an action is applied, its effects must be of some use (Karpas and Domshlak 2012)

1. one of its effects is necessary for applying another action
2. one of its effects is a goal variable (that is not made false again)

$$\varphi_a = \bigvee_{e \in add(a) \setminus G} \Big((e \wedge \bigwedge_{\substack{a' \in A \text{ with} \\ e \in add(a')}} \neg a') \mathcal{U} \bigvee_{\substack{a' \in A \text{ with} \\ e \in pre(a')}} a' \Big) \vee \bigvee_{e \in add(a) \cap G} \Big((e \wedge \bigwedge_{\substack{a' \in A \text{ with} \\ e \in add(a')}} \neg a') \mathcal{U} \big(\mathit{last} \vee \bigvee_{\substack{a' \in A \text{ with} \\ e \in pre(a')}} a' \big) \Big)$$

Heuristics

- Very rich temporal information possible
  $\rightarrow$ heuristics can use different levels of relaxation
- Proof of concept heuristic extracts landmarks from node-associated formulas
  $\rightarrow$ looses temporal information between landmarks

Experiment Setup

Configurations:

1. h_{LA}: standard admissible landmark heuristic (Karpas and Domshlak 2009)
2. $h_{\mathrm{AL}}^{\mathrm{LM}}$: LTL landmark extraction heuristic with sources:
   - Landmarks (First achievers, Required again, Goal)
3. $h_{\mathrm{AL}}^{\mathrm{LM+UAA}}$: LTL landmark extraction heuristic with sources:
   - Landmarks (First achievers, Required again, Goal)
   - Unjustified Action Applications

- all heuristics use BJOLP landmark extraction and optimal cost partitioning
- search algorithm: h_{LA} uses $\mathrm{LM\text{-}A}^*$, the others a slight variant we call $\mathrm{LTL\text{-}A}^*$

Coverage

	h_{LA}	$h_{\mathrm{AL}}^{\mathrm{LM}}$	$h_{\mathrm{AL}}^{\mathrm{LM+UAA}}$
airport (50)	**31**	28	26
elevators-08 (30)	**14**	**14**	13
floortile (20)	2	2	**4**
freecell (80)	**52**	51	50
mprime (35)	19	19	**20**
nomystery (20)	**18**	17	16
openstacks-08 (30)	**14**	12	12
openstacks-11 (20)	**9**	7	7
parcprinter-08 (30)	**15**	14	14
parcprinter-11 (20)	**11**	10	10
pipesworld-tan (50)	9	**10**	**10**
scanalyzer-08 (30)	**10**	9	9
sokoban-08 (30)	**22**	21	**22**
tidybot (20)	**14**	**14**	13
other domains (931)	**483**	**483**	**483**
Sum (1396)	**723**	711	709

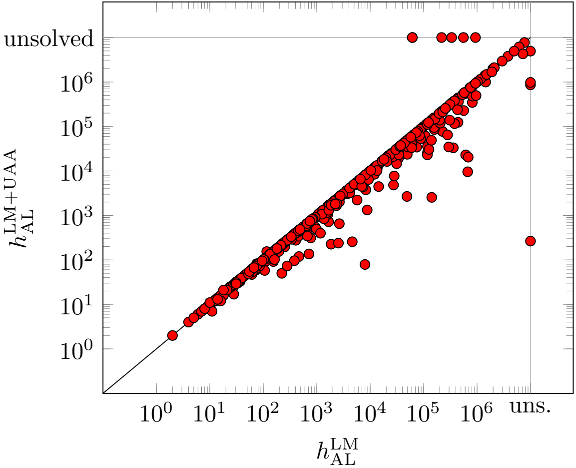
Memory Consumption

h_{LA} looses no task due to memory limit, but $h_{\mathrm{AL}}^{\mathrm{LM}}$ 11 in total

- airport: over 300% of memory usage compared to h_{LA}
- average: 120%
- approx. half the domains $< 100\%$

Impact of Unjustified Action Applications

Comparison of expansions between $h_{\mathrm{AL}}^{\mathrm{LM}}$ and $h_{\mathrm{AL}}^{\mathrm{LM+UAA}}$:



Conclusion

- associate nodes in the search space with LTL$_f$ formulas
  $\rightarrow$ conditions for optimal plan
- separation between finding information and exploiting information
- allows to easily combine information from different sources
- concrete examples in this paper:
  - finding information: landmarks and unjustified action applications
  - exploiting information: extracting landmarks

Future Work

- better informed heuristics (less relaxation)
- encodings for other kinds of information
- strengthening other heuristics with the information of LTL$_f$ trajectory constraints