Planning and Optimization

C16. M&S: Strategies and Label Reduction

Malte Helmert and Gabriele Röger

Universität Basel

November 17, 2016

Motivation

Generic Algorithm Template

Generic Abstraction Computation Algorithm

$abs := \{\mathcal{T}^{\pi_{\{v\}}} \mid v \in V\}$
while *abs* contains more than one abstract transition system:
  select $\mathcal{A}_1$, $\mathcal{A}_2$ from *abs*
  shrink $\mathcal{A}_1$ and/or $\mathcal{A}_2$ until $size(\mathcal{A}_1) \cdot size(\mathcal{A}_2) \leq N$
  $abs := abs \setminus \{\mathcal{A}_1, \mathcal{A}_2\} \cup \{\mathcal{A}_1 \otimes \mathcal{A}_2\}$
return the remaining abstract transition system in *abs*

Remaining questions:

- Which abstractions to select? $\rightsquigarrow$ merging strategy
- How to shrink an abstraction? $\rightsquigarrow$ shrinking strategy

Merging Strategies

Linear Merging Strategies

Linear Merging Strategy

In each iteration after the first, choose the abstraction computed in the previous iteration as $\mathcal{A}_1$.

Rationale: only maintains one "complex" abstraction at a time

$\rightsquigarrow$ Fully defined by an ordering of atomic projections.

Linear Merging Strategies: Choosing the Ordering

Use similar causal graph criteria as for growing patterns.

Example: Strategy of h_{HHH}

h_{HHH}: Ordering of atomic projections

- Start with a goal variable.
- Add variables that appear in preconditions of operators affecting previous variables.
- If that is not possible, add a goal variable.

Rationale: increases h quickly

Non-linear Merging Strategies

- Non-linear merging strategies only recently gained more interest in the planning community.
- One reason: Better label reduction techniques (later in this chapter) enabled a more efficient computation.
- Examples:
  - DFP: preferrably merge transition systems that must synchronize on labels that occur close to a goal state.
  - UMC and MIASM: Build clusters of variables with strong interactions and first merge variables within each cluster.
- Each merge-and-shrink heuristic computed with a non-linear merging strategy can also be computed with a linear merging strategy.
- However, linear merging can require a super-polynomial blow-up of the final representation size.

Shrinking Strategies

Shrinking strategies

How to shrink an abstraction?

We cover two common approaches:

- f-preserving shrinking
- bisimulation-based shrinking

f-preserving Shrinking Strategy

f-preserving Shrinking Strategy

Repeatedly combine abstract states with
identical abstract goal distances (h values) and
identical abstract initial state distances (g values).

Rationale: preserves heuristic value and overall graph shape

Tie-breaking Criterion

Prefer combining states where $g + h$ is high.
In case of ties, combine states where h is high.

Rationale: states with high $g + h$ values are less likely to be explored by A*, so inaccuracies there matter less

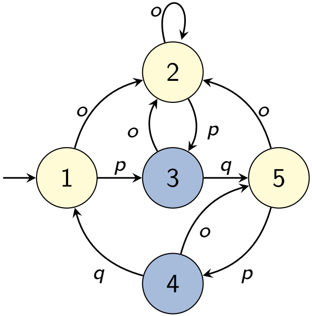
Bisimulation

Definition (Bisimulation)

Let $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$ be a transition system. An equivalence relation $\sim$ on S is a bisimulation for $\mathcal{T}$ if for every $\langle s, \ell, s' \rangle \in T$ and every $t \sim s$ there is a transition $\langle t, \ell, t' \rangle \in T$ with $t' \sim s'$.

A bisimulation $\sim$ is goal-respecting if $s \sim t$ implies that either $s, t \in S_\star$ or $s, t \notin S_\star$.

Bisimulation: Example



$\sim$ with equivalence classes $\{\{1, 2, 5\}, \{3, 4\}\}$ is a goal-respecting bisimulation.

Bisimulations as Abstractions

Theorem (Bisimulations as Abstractions)

Let $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$ be a transition system and $\sim$ be a bisimulation for $\mathcal{T}$. Then $\alpha_\sim : S \to \{[s]_\sim \mid s \in S\}$ with $\alpha_\sim(s) = [s]_\sim$ is an abstraction of $\mathcal{T}$.

Note: $[s]_\sim$ denotes the equivalence class of s.

Note: Surjectivity follows from the definition of the codomain as the image of $\alpha_\sim$.

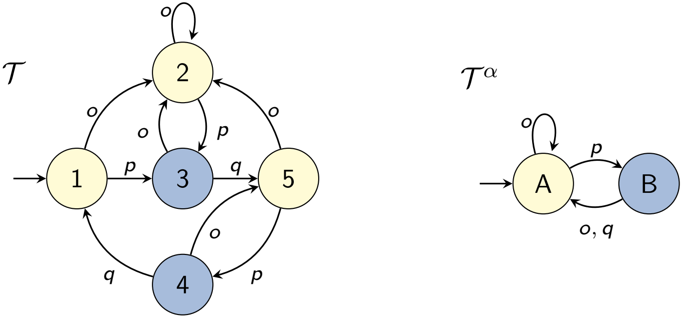
Abstractions as Bisimulations

Definition (Abstraction as Bisimulation)

Let $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$ be a transition system and $\alpha : S \to S'$ be an abstraction of $\mathcal{T}$. The abstraction induces the equivalence relation $\sim_\alpha$ as $s \sim_\alpha t$ iff $\alpha(s) = \alpha(t)$.
We say that α is a (goal-respecting) bisimulation for $\mathcal{T}$ if $\sim_\alpha$ is a (goal-respecting) bisimulation for $\mathcal{T}$.

Abstraction as Bisimulations: Example

Abstraction α with
$\alpha(1) = \alpha(2) = \alpha(5) = A$ and $\alpha(3) = \alpha(4) = B$
is a goal-respecting bisimulation for $\mathcal{T}$.



Goal-respecting Bisimulations are Exact (1)

Theorem

Let X be a collection of transition systems. Let α be an abstraction for $\mathcal{T}_i \in X$. If α is a goal-respecting bisimulation then the transformation from X to $X' := (X \setminus \{\mathcal{T}_i\}) \cup \{\mathcal{T}_i^\alpha\}$ is exact.

Proof.

Let $\mathcal{T}_X = \mathcal{T}_1 \otimes \cdots \otimes \mathcal{T}_n = \langle S, L, c, T, s_0, S_\star \rangle$ and w.l.o.g. $\mathcal{T}_{X'} = \mathcal{T}_1 \otimes \cdots \otimes \mathcal{T}_{i-1} \otimes \mathcal{T}_i^\alpha \otimes \mathcal{T}_{i+1} \otimes \cdots \otimes \mathcal{T}_n = \langle S', L', c', T', s_0', S_\star' \rangle$.
Consider $\sigma(\langle s_1, \ldots, s_n \rangle) = \langle s_1, \ldots, s_{i-1}, \alpha(s_i), s_{i+1}, \ldots, s_n \rangle$ for the mapping of states and $\tau = \mathrm{id}$ for the mapping of labels.

1. Mappings σ and τ satisfy the requirements of safe transformations because α is an abstraction and we have chosen the mapping functions as before.

...

Goal-respecting Bisimulations are Exact (2)

Proof (continued).

2. If $\langle s', \ell, t' \rangle \in T'$ with $s' = \langle s'_1, \dots, s'_n \rangle$ and $t' = \langle t'_1, \dots, t'_n \rangle$, then for $j \neq i$ transition system $\mathcal{T}_j$ has transition $\langle s'_j, \ell, t'_j \rangle$ (*) and $\mathcal{T}_i^\alpha$ has transition $\langle s'_i, \ell, t'_i \rangle$. This implies that $\mathcal{T}_i$ has a transition $\langle s''_i, \ell, t''_i \rangle$ for some $s''_i \in \alpha^{-1}(s'_i)$ and $t''_i \in \alpha^{-1}(t'_i)$. As α is a bisimulation, there must be such a transition for *all* such s''_i and t''_i (**).
   Each $s \in \sigma^{-1}(s')$ has the form $s = \langle s_1, \dots, s_n \rangle$ with $s_j = s'_j$ for $j \neq i$ and $s_i \in \alpha^{-1}(s'_i)$. Analogously for each $t = \langle t_1, \dots, t_n \rangle \in \sigma^{-1}(t')$. From (*) and (**) follows that $\mathcal{T}_j$ has a transition $\langle s_j, \ell, t_j \rangle$ for all $j \in \{1, \dots, n\}$, so for each such s and t, T contains the transition $\langle s, \ell, t \rangle$.

...

Goal-respecting Bisimulations are Exact (3)

Proof (continued).

3. For $s'_\star = \langle s'_1, \dots, s'_n \rangle \in S'_\star$, each s'_j with $j \neq i$ must be a goal state of $\mathcal{T}_j$ (*) and s'_i must be a goal state of $\mathcal{T}_i^\alpha$. The latter implies that at least on $s''_i \in \alpha^{-1}(s'_i)$ is a goal state of $\mathcal{T}_i$. As α is goal-respecting, all states from $\alpha^{-1}(s'_i)$ are goal states of $\mathcal{T}_i$ (**).
   Consider $s_\star = \langle s_1, \dots, s_n \rangle \in \sigma^{-1}(s'_\star)$. By the definition of σ, $s_j = s'_j$ for $j \neq i$ and $s_i \in \alpha^{-1}(s'_i)$. From (*) and (**), each s_j ($j \in \{1, \dots, n\}$) is a goal state of $\mathcal{T}_j$ and, hence, $s_\star$ a goal state of $\mathcal{T}_X$.
4. As $\tau = \text{id}$ and the transformation does not change the label cost function, $c(\ell) = c'(\tau(\ell))$ for all $\ell \in L$.

□

Bisimulations: Discussion

- As all bisimulations preserve all relevant information, we are interested in the coarsest such abstraction (to shrink as much as possible).
- There is always a unique coarsest bisimulation for $\mathcal{T}$ and it can be computed efficiently (from the explicit representation).
- In some cases, computing the bisimulation is still too expensive or it cannot sufficiently shrink a transition system.

Greedy Bisimulations

Definition (Greedy Bisimulation)

Let $\mathcal{T} = \langle S, L, c, T, s_0, S_\star \rangle$ be a transition system. An equivalence relation $\sim$ on S is a greedy bisimulation for $\mathcal{T}$ if it is a bisimulation for the system $\langle S, L, c, T^G, s_0, S_\star \rangle$, where
$T^G = \{\langle s, \ell, t \rangle \mid \langle s, \ell, t \rangle \in T, h^*(s) = h^*(t) + c(\ell)\}$.

Greedy bisimulation only considers transitions that are used in an optimal solution of some state of $\mathcal{T}$.

Greedy Bisimulation is h-preserving

Theorem

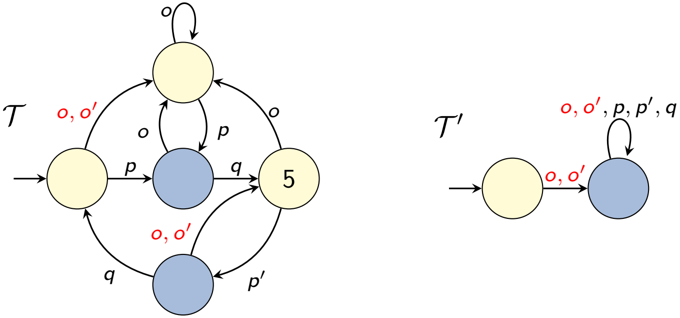
Let $\mathcal{T}$ be a transition system and let α be an abstraction of $\mathcal{T}$. If $\sim_\alpha$ is a goal-respecting greedy bisimulation for $\mathcal{T}$ then $h^*_{\mathcal{T}^\alpha} = h^*_{\mathcal{T}}$.

(Proof omitted.)

Note: This does not mean that replacing $\mathcal{T}$ with $\mathcal{T}^\alpha$ in a collection of transition systems is a safe transformation! Abstraction α preserves solution costs "locally" but not "globally".

Label Reduction

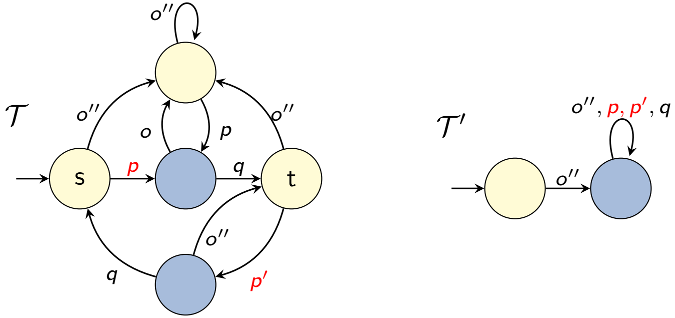
Label Reduction: Motivation (1)



Whenever there is a transition with label o' there is also a transition with label o. If o' is not cheaper than o, we can always use the transition with o.

Idea: Replace o and o' with label o'' with cost of o

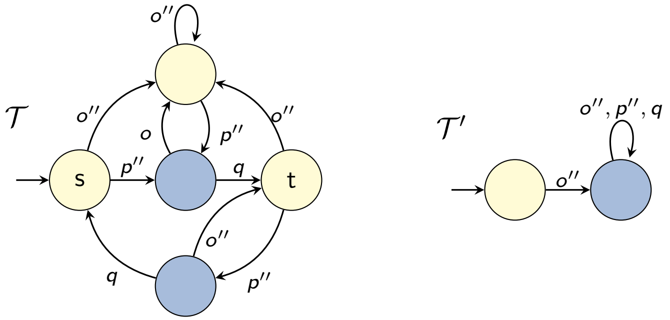
Label Reduction: Motivation (2)



States s and t are not bisimilar due to labels p and p'. In $\mathcal{T}'$ they label the same (parallel) transitions. If p and p' have the same cost, in such a situation there is no need for distinguishing them.

Idea: Replace p and p' with label p'' with same cost.

Label Reduction: Motivation (3)



Label reductions reduce the time and memory requirement for merge and shrink steps and enable coarser bisimulation abstractions.

When is label reduction a safe transformation?

Label Reduction: Definition

Definition (Label Reduction)

Let X be a collection of transition systems with label set L and label cost function c. A label reduction $\langle\tau, c'\rangle$ for X is given by a function $\tau : L \to L'$, where L' is an arbitrary set of labels, and a label cost function c' on L' such that for all $\ell \in L$, $c'(\tau(\ell)) \leq c(\ell)$.

For $\mathcal{T} = \langle S, L, c, T, s_0, S_\star\rangle \in X$ the label-reduced transition system is $\mathcal{T}^{\langle\tau,c'\rangle} = \langle S, L', c', \{\langle s, \tau(\ell), t\rangle \mid \langle s, \ell, t\rangle \in T\}, s_0, S_\star\rangle$.

The label-reduced collection is $X^{\langle\tau,c'\rangle} = \{\mathcal{T}^{\langle\tau,c'\rangle} \mid \mathcal{T} \in X\}$.

$L' \cap L \neq \emptyset$ and $L' = L$ are allowed.

Label Reduction is Safe (1)

Theorem (Label Reduction is Safe)

Let X be a collection of transition systems and $\langle \tau, c' \rangle$ be a label-reduction for X. The transformation from X to $X^{\langle \tau, c' \rangle}$ is safe.

Proof.

We show that the transformation is safe, using $\sigma = \mathrm{id}$ for the mapping of states and τ for the mapping of labels.

The label set of $\mathcal{T}_{X^{\langle \tau, c' \rangle}}$ corresponds to the image of τ by the definition of $X^{\langle \tau, c' \rangle}$ and $\mathcal{T}_{X^{\langle \tau, c' \rangle}}$.

The label cost function of $\mathcal{T}_{X^{\langle \tau, c' \rangle}}$ is c' and has the required property by the definition of label reduction. ...

Label Reduction is Safe (2)

Theorem (Label Reduction is Safe)

Let X be a collection of transition systems and $\langle \tau, c' \rangle$ be a label-reduction for X. The transformation from X to $X^{\langle \tau, c' \rangle}$ is safe.

Proof (continued).

By the definition of synchronized products, $\mathcal{T}_X$ has a transition $\langle \langle s_1, \dots, s_{|X|} \rangle, \ell, \langle t_1, \dots, t_{|X|} \rangle \rangle$ if for all i, $\mathcal{T}_i \in X$ has a transition $\langle s_i, \ell, t_i \rangle$. By the definition of label-reduced transition systems, this implies that $\mathcal{T}^{\langle \tau, c' \rangle}$ has a corresponding transition $\langle s_i, \tau(\ell), t_i \rangle$, so $\mathcal{T}_{X^{\langle \tau, c' \rangle}}$ has a transition $\langle s, \tau(\ell), t \rangle = \langle \sigma(s), \tau(\ell), \sigma(t) \rangle$ (definition of synchronized products).

For each goal state $s_\star$ of $\mathcal{T}_X$, state $\sigma(s_\star) = s_\star$ is a goal state of $\mathcal{T}_{X^{\langle \tau, c' \rangle}}$ because the transformation replaces each transition system with a system that has the same goal states. □

More Terminology

Let X be a collection of transition systems with labels L. Let $\ell, \ell' \in L$ be labels and let $\mathcal{T} \in X$.

- Label ℓ is alive in X if all $\mathcal{T}' \in X$ have some transition labelled with ℓ. Otherwise, ℓ is dead.
- Label ℓ locally subsumes label ℓ' in $\mathcal{T}$ if for all transitions $\langle s, \ell', t \rangle$ of $\mathcal{T}$ there is also a transition $\langle s, \ell, t \rangle$ in $\mathcal{T}$.
- ℓ globally subsumes ℓ' if it locally subsumes ℓ' in all $\mathcal{T}' \in X$.
- ℓ and ℓ' are locally equivalent in $\mathcal{T}$ if they label the same transitions in $\mathcal{T}$, i.e. ℓ locally subsumes ℓ' in $\mathcal{T}$ and vice versa.
- ℓ and ℓ' are $\mathcal{T}$-combinable if they are locally equivalent in all transition systems $\mathcal{T}' \in X \setminus \{\mathcal{T}\}$.

Exact Label Reduction

Theorem (Criteria for Exact Label Reduction)

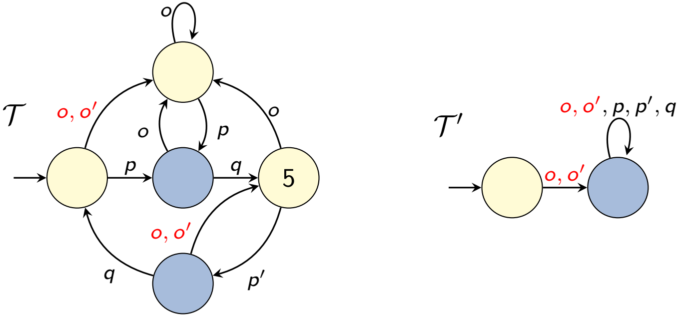
Let X be a collection of transition systems with cost function c and label set L that contains no dead labels.

Let $\langle \tau, c' \rangle$ be a label-reduction for X such that τ combines labels ℓ_1 and ℓ_2 and leaves other labels unchanged. The transformation from X to $X^{\langle \tau, c' \rangle}$ is exact iff $c(\ell_1) = c(\ell_2)$, $c'(\tau(\ell)) = c(\ell)$ for all $\ell \in L$, and

- *ℓ_1 globally subsumes ℓ_2, or*
- *ℓ_2 globally subsumes ℓ_1, or*
- *ℓ_1 and ℓ_2 are $\mathcal{T}$-combinable for some $\mathcal{T} \in X$.*

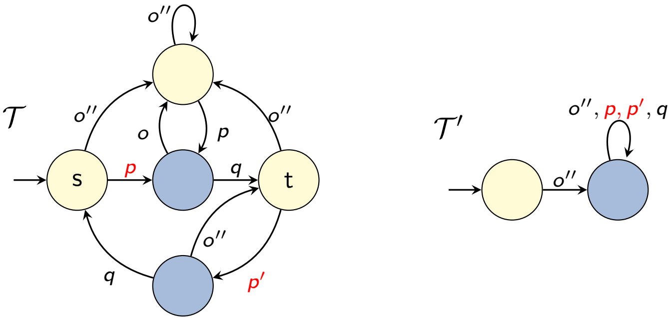
(Proof omitted.)

Back to Example (1)



Label o' globally subsumes label o.

Back to Example (2)



Labels p and p' are $\mathcal{T}$-combinable.

Computation of Exact Label Reduction (1)

- For given labels ℓ_1, ℓ_2, the criteria can be tested in low-order polynomial time.
- Finding globally subsumed labels involves finding subset relationsships in a set family.
  $\rightsquigarrow$ no linear-time algorithms known
- The following algorithm exploits only $\mathcal{T}$-combinability.

Computation of Exact Label Reduction (2)

eq_i := set of label equivalence classes of $\mathcal{T}_i \in X$

Label-reduction based on $\mathcal{T}_i$-combinability

$eq := \{L\}$
for $j \in \{1, \dots, |X|\} \setminus \{i\}$
    Refine eq with eq_j
// two labels are in the same set of eq
// iff they are locally equivalent in all $\mathcal{T}_j \neq \mathcal{T}_i$.
$\tau = \mathrm{id}$
for $B \in eq$
    $samecost := \{[\ell]_{\sim_c} \mid \ell \in B, \ell' \sim_c \ell'' \text{ iff } c(\ell') = c(\ell'')\}$
    for $L' \in samecost$
        ℓ_{new} := new label
        $c'(\ell_{\text{new}})$:= cost of labels in L'
        for $\ell \in L'$
            $\tau(\ell) = \ell_{\text{new}}$

Application in Merge-and-Shrink Algorithm

Generic Abstraction Computation Algorithm with Label Reduction

$abs := \{\mathcal{T}^{\pi_{\{v\}}} \mid v \in V\}$
while *abs* contains more than one abstract transition system:
    select $\mathcal{T}_1$, $\mathcal{T}_2$ from *abs*
    possibly label-reduce all $\mathcal{T} \in abs$
        (e.g. based on $\mathcal{T}_1$- and/or $\mathcal{T}_2$-combinability).
    shrink $\mathcal{T}_1$ and/or $\mathcal{T}_2$ until $size(\mathcal{T}_1) \cdot size(\mathcal{T}_2) \leq N$
    possibly label-reduce all $\mathcal{T} \in abs$
    $abs := abs \setminus \{\mathcal{T}_1, \mathcal{T}_2\} \cup \{\mathcal{T}_1 \otimes \mathcal{T}_2\}$
return the remaining abstract transition system in *abs*

Summary

Summary

- There is a wide range of merging and shrinking strategies. We only covered some important ones.
- Bisimulation is an exact shrinking method.
- Label reduction is crucial for the performance of the merge-and-shrink algorithm, especially when using bisimilarity for shrinking.

Literature

Literature (1)

References on merge-and-shrink abstractions:

- Klaus Dräger, Bernd Finkbeiner and Andreas Podelski.
  Directed Model Checking with Distance-Preserving Abstractions.
  Proc. SPIN 2006, pp. 19–34, 2006.
  Introduces merge-and-shrink abstractions (for model-checking) and DFP merging strategy.

- Malte Helmert, Patrik Haslum and Jörg Hoffmann.
  Flexible Abstraction Heuristics for Optimal Sequential Planning.
  Proc. ICAPS 2007, pp. 176–183, 2007.
  Introduces merge-and-shrink abstractions for planning.

Literature (2)

Raz Nissim, Jörg Hoffmann and Malte Helmert.
Computing Perfect Heuristics in Polynomial Time: On Bisimulation and Merge-and-Shrink Abstractions in Optimal Planning.
Proc. IJCAI 2011, pp. 1983–1990, 2011.
Introduces bisimulation-based shrinking.

Malte Helmert, Patrik Haslum, Jörg Hoffmann and Raz Nissim.
Merge-and-Shrink Abstraction: A Method for Generating Lower Bounds in Factored State Spaces.
Journal of the ACM 61 (3), pp. 16:1–63, 2014.
Detailed journal version of the previous two publications.

Literature (3)

Silvan Sievers, Martin Wehrle and Malte Helmert.
Generalized Label Reduction for Merge-and-Shrink Heuristics.
Proc. AAAI 2014, pp. 2358–2366, 2014.
Introduces label reduction as covered in these slides
(there has been a more complicated version before).

Gaojian Fan, Martin Müller and Robert Holte.
Non-linear merging strategies for merge-and-shrink based on variable interactions.
Proc. AAAI 2014, pp. 2358–2366, 2014.
Introduces UMC and MIASM merging strategies