Merging or Computing Saturated Cost Partitionings?

A Merge Strategy for the Merge-and-Shrink Framework

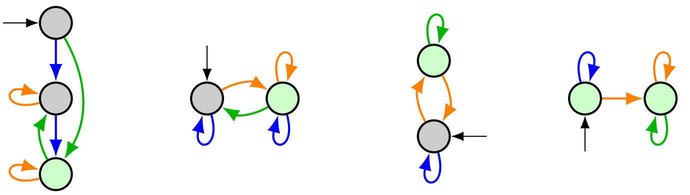
Silvan Sievers, Thomas Keller, Gabriele Röger

ICAPS 2024

University of Basel

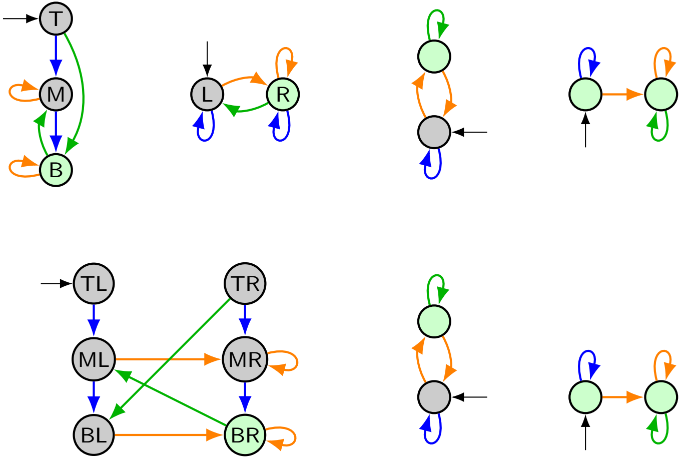
Merge-and-Shrink Abstractions: Idea

Start from atomic factors (projections to single state variables)



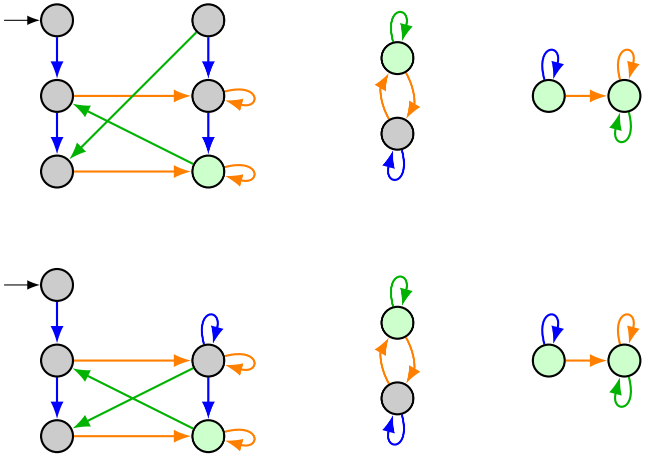
Merge-and-Shrink Abstractions: Idea

Merge: replace two factors with their product



Merge-and-Shrink Abstractions: Idea

Shrink: replace a factor by an abstraction of it



Merge-and-Shrink Algorithm

Dräger et al. STTT 2006, Helmert et al. JACM 2014, Sievers & Helmert JAIR 2021

Input: FTS F
Output: Heuristic for F

function $\mathrm{M\&S}(F)$
    $F' \leftarrow F$
    while not $\textsc{Terminate}(F')$ **do**
        $i, j \leftarrow \textsc{MergeStrategy}(F')$
        $\textsc{LabelReductionStrategy}(F')$
        $\textsc{ShrinkStrategy}(F', i, j)$
        $k \leftarrow \textsc{Merge}(F', i, j)$
        $\textsc{PruneStrategy}(F', k)$
    return $\textsc{ComputeHeuristic}(F')$

- Factor heuristic: abstraction heuristic from single factor (= abstract transition system)
- Run until there is only a single factor and use its factor heuristic, or
- terminate early and use the maximum of the factor heuristics.

Merge-and-Shrink with Saturated Cost Partitioning

Sievers et al. IJCAI 2020

- Saturated Cost Partitioning
  (Seipp et al. JAIR 2020)
  Admissible combination of heuristics.
- Typically better than maximum.
- Depends on the order in which the heuristics are considered.

Merge-and-Shrink with Saturated Cost Partitioning

Sievers et al. IJCAI 2020

- Saturated Cost Partitioning
  (Seipp et al. JAIR 2020)
  Admissible combination of heuristics.
- Typically better than maximum.
- Depends on the order in which the heuristics are considered.

Input: FTS F
Output: Heuristic for F

function $\textsc{M\&SwithSCP}(F)$
    $F' \leftarrow F$, $H \leftarrow \emptyset$
    while not $\textsc{Terminate}(F')$ **do**
        $i, j \leftarrow \textsc{MergeStrategy}(F')$
        $\textsc{LabelReductionStrategy}(F')$
        $\omega \leftarrow \textsc{SCPOrderStrategy}(F')$
        $H \leftarrow H \cup \{h^{\mathsf{SCP}}_{\omega}\}$
        $\textsc{ShrinkStrategy}(F', i, j)$
        $k \leftarrow \textsc{Merge}(F', i, j)$
        $\textsc{PruneStrategy}(F', k)$
    return $\textsc{ComputeMaxHeuristic}(H)$

We want to devise a merge strategy that works well in M&S with cost partitioning.

Evaluating Merge Candidates

For evaluating a pair of factors, we locally assess the value of merging them and of using them as individual heuristics:

$$
\begin{aligned}
h^{init}_{prod} &= h^{\mathcal{T}^{\otimes}}_{F}(s_0) \\
h^{init}_{\text{mFactor}} &= \max(h^{\mathcal{T}^i}_{F}(s_0), h^{\mathcal{T}^j}_{F}(s_0)) \\
h^{init}_{\text{mSCP}} &= \max(h^{\text{SCP}}_{\langle \mathcal{T}^i_F, \mathcal{T}^j_F \rangle}(s_0), h^{\text{SCP}}_{\langle \mathcal{T}^j_F, \mathcal{T}^i_F \rangle}(s_0))
\end{aligned}
$$

Evaluating Merge Candidates

For evaluating a pair of factors, we locally assess the value of merging them and of using them as individual heuristics:

$$
\begin{aligned}
h^{init}_{prod} &= h^{\mathcal{T}^{\otimes}}_{F}(s_0) \\
h^{init}_{\mathrm{mFactor}} &= \max(h^{\mathcal{T}^i}_{F}(s_0), h^{\mathcal{T}^j}_{F}(s_0)) \\
h^{init}_{\mathrm{mSCP}} &= \max(h^{\mathrm{SCP}}_{\langle \mathcal{T}^i_F, \mathcal{T}^j_F \rangle}(s_0), h^{\mathrm{SCP}}_{\langle \mathcal{T}^j_F, \mathcal{T}^i_F \rangle}(s_0))
\end{aligned}
$$

$$
\begin{aligned}
h^{avg}_{prod} &= \textsc{Avg}(h^{\mathcal{T}^{\otimes}}_{F}) \\
h^{avg}_{\mathrm{mFactor}} &= \max(\textsc{Avg}(h^{\mathcal{T}^j}_{F}), \textsc{Avg}(h^{\mathcal{T}^i}_{F})) \\
h^{avg}_{\mathrm{mSCP}} &= \max(\textsc{Avg}(h^{\mathrm{SCP}}_{\langle \mathcal{T}^i_F, \mathcal{T}^j_F \rangle}), \textsc{Avg}(h^{\mathrm{SCP}}_{\langle \mathcal{T}^j_F, \mathcal{T}^i_F \rangle}))
\end{aligned}
$$

Two New Merge Strategies

- maximum factor scoring function (mFactor) prefers candidates whose product heuristic improves most compared to the maximum over the two factor heuristics:

$$\text{Maximize } h_{prod} - h_{\text{mFactor}}$$

  Rationale: greedy decision for the best immediate improvement without looking ahead to the future transformations by M&S.

Two New Merge Strategies

- maximum factor scoring function (mFactor) prefers candidates whose product heuristic improves most compared to the maximum over the two factor heuristics:

$$\text{Maximize } h_{prod} - h_{\text{mFactor}}$$

  Rationale: greedy decision for the best immediate improvement without looking ahead to the future transformations by M&S.

- maximum SCP scoring function (mSCP) adapts the same concept to the integration of cost partitioning into M&S.

$$\text{Maximize } h_{prod} - h_{\text{mSCP}}$$

Comparison

	mFactor		mSCP	
	init	avg	init	avg
$h^{\mathrm{M\&S}}$	889	860	**902**	875
$h^{\mathrm{M\&S}}_{\mathrm{SCP}}$	916	902	**990**	909

Comparison

	mFactor		mSCP	
	init	avg	init	avg
$h^{\mathrm{M\&S}}$	889	860	**902**	875
$h^{\mathrm{M\&S}}_{\mathrm{SCP}}$	916	902	**990**	909

Additional experiments:

- Stopping the M&S algorithm when there is no good merge candidate leads to worse coverage, because continuing merging factors can lead to better factor heuristics in later iterations.
- Adding SCP heuristics for each pair of remaining factors does not pay off.

State-of-the-art Strategies

			SCC		mSCP	
	DFP	sbM	DFP	sbM		SCC
$h^{\text{M\&S}}$	882	920	922	913	902	**926**
$h^{\text{M\&S}}_{\text{SCP}}$	915	965	950	956	990	**1006**

DFP Dräger et al. SPIN 2006, Sievers et al. AAAI 2014

sbM sbMIASM; Fan et al. SoCS 2014, Sievers et al. ICAPS 2016

SCC Sievers et al. ICAPS 2016

Summary

- New merge strategy for M&S with saturated cost partitioning.
- Improves the state of the art of M&S.
- Even better if integrated with the SCC merge strategy.