Universität Hamburg
DER FORSCHUNG | DER LEHRE | DER BILDUNG

CHAI
Humanities-Centered AI

Universität Münster

Compression vs. Accuracy: Compact Models for Efficiency and Interpretation

IJCAI-ECAI 2026 – Bremen, Germany

Malte Luttermann[1], Jan Speller[2], Marcel Gehrke[1], Tanya Braun[2]

[1]Institute for Humanities-Centered Artificial Intelligence, University of Hamburg, Germany
[2]Data Science Group, University of Münster, Germany

August 15, 2026

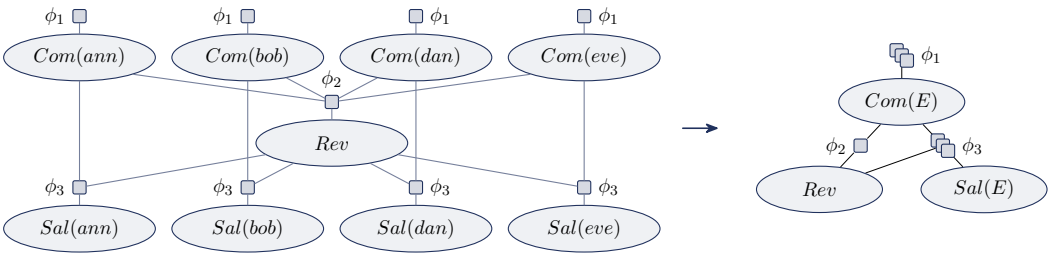
Agenda

1. Introduction and background [Tanya]
   - Motivation, use cases
   - Probabilistic (relational) models
   - Lifted probabilistic inference
2. Compressing models [Malte]
   - The basics of colour passing
   - Recent advances in colour passing
3. Accuracy considerations [Jan]
   - Hierarchical colour passing
   - Geometric interpretation
4. Summary and future directions [Tanya]

Compressing Models – Problem Setup I

Goal: Uncover indistinguishability in a propositional model and construct a lifted model.

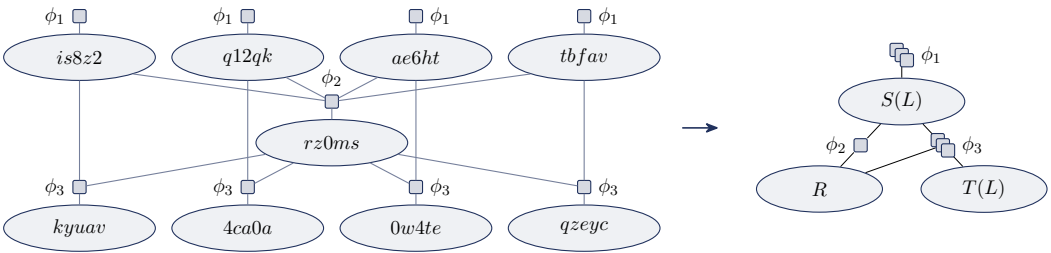
- Input: A factor graph M.
- Output: A parametric factor graph entailing equivalent semantics as M.



Compressing Models – Problem Setup II

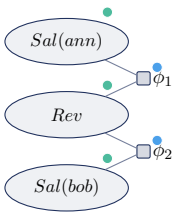
Uncovering indistinguishability is challenging:

- Nodes might carry arbitrary identifiers,
- repeated structures are hidden.



Compressing Models – Colour Passing

- Colour random variables and factors by their local properties,

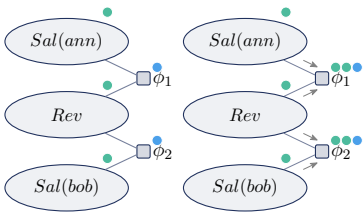


Kristian Kersting, Babak Ahmadi, and Sriraam Natarajan (2009). »Counting Belief Propagation«. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009)*. AUAI Press, pp. 277–284.

Babak Ahmadi, Kristian Kersting, Martin Mladenov, and Sriraam Natarajan (2013). »Exploiting Symmetries for Scaling Loopy Belief Propagation and Relational Training«. *Machine Learning* 92, pp. 91–132.

Compressing Models – Colour Passing

- Colour random variables and factors by their local properties,
- iteratively propagate colours,

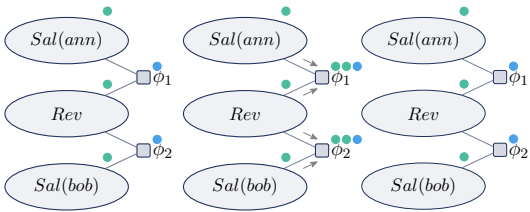


Kristian Kersting, Babak Ahmadi, and Sriraam Natarajan (2009). »Counting Belief Propagation«. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009)*. AUAI Press, pp. 277–284.

Babak Ahmadi, Kristian Kersting, Martin Mladenov, and Sriraam Natarajan (2013). »Exploiting Symmetries for Scaling Loopy Belief Propagation and Relational Training«. *Machine Learning* 92, pp. 91–132.

Compressing Models – Colour Passing

- Colour random variables and factors by their local properties,
- iteratively propagate colours,

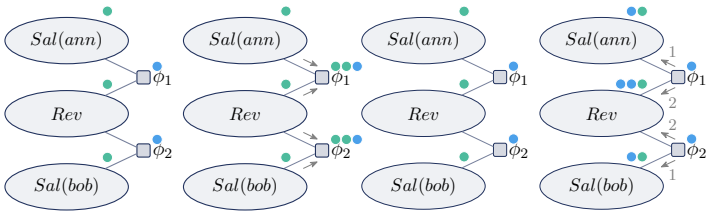


Kristian Kersting, Babak Ahmadi, and Sriraam Natarajan (2009). »Counting Belief Propagation«. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009)*. AUAI Press, pp. 277–284.

Babak Ahmadi, Kristian Kersting, Martin Mladenov, and Sriraam Natarajan (2013). »Exploiting Symmetries for Scaling Loopy Belief Propagation and Relational Training«. *Machine Learning* 92, pp. 91–132.

Compressing Models – Colour Passing

- Colour random variables and factors by their local properties,
- iteratively propagate colours,

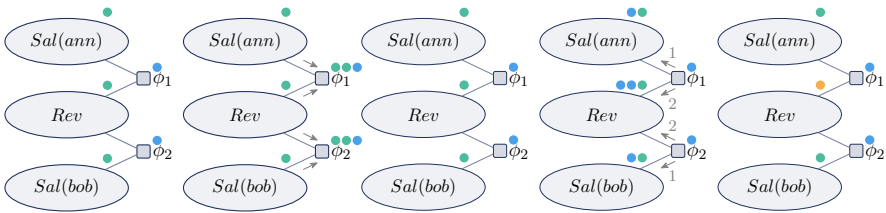


Kristian Kersting, Babak Ahmadi, and Sriraam Natarajan (2009). »Counting Belief Propagation«. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009)*. AUAI Press, pp. 277–284.

Babak Ahmadi, Kristian Kersting, Martin Mladenov, and Sriraam Natarajan (2013). »Exploiting Symmetries for Scaling Loopy Belief Propagation and Relational Training«. *Machine Learning* 92, pp. 91–132.

Compressing Models – Colour Passing

- Colour random variables and factors by their local properties,
- iteratively propagate colours,
- build groups according to colours at fixed point.

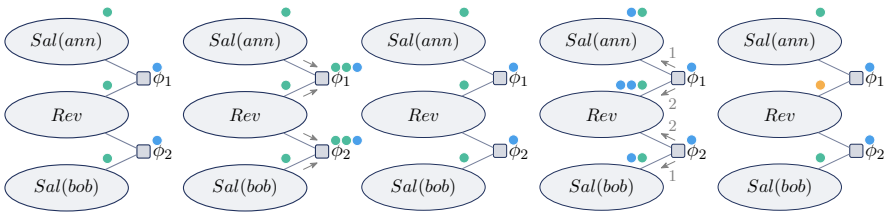


Kristian Kersting, Babak Ahmadi, and Sriraam Natarajan (2009). »Counting Belief Propagation«. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009)*. AUAI Press, pp. 277–284.

Babak Ahmadi, Kristian Kersting, Martin Mladenov, and Sriraam Natarajan (2013). »Exploiting Symmetries for Scaling Loopy Belief Propagation and Relational Training«. *Machine Learning* 92, pp. 91–132.

Details that Need to be Addressed

1. Identification of commutative factors
2. Identification of exchangeable (i.e., equivalent) factors
3. Introduction of logical variables

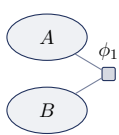


Kristian Kersting, Babak Ahmadi, and Sriraam Natarajan (2009). »Counting Belief Propagation«. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009)*. AUAI Press, pp. 277–284.

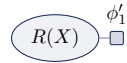
Babak Ahmadi, Kristian Kersting, Martin Mladenov, and Sriraam Natarajan (2013). »Exploiting Symmetries for Scaling Loopy Belief Propagation and Relational Training«. *Machine Learning* 92, pp. 91–132.

Commutative Factors

- A commutative factor encodes a symmetric function
  - E.g., $\phi(A, B) = \phi(B, A)$
- Might also apply to a subset of arguments only
  - E.g., $\phi(A, B, R) = \phi(B, A, R)$



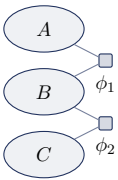
A	B	$\phi_1(A, B)$
high	high	φ_1
high	low	φ_2
low	high	φ_2
low	low	φ_3



$\#_X[R(X)]$	$\phi'_1(\#_X[R(X)])$
$[2, 0]$	φ_1
$[1, 1]$	φ_2
$[0, 2]$	φ_3

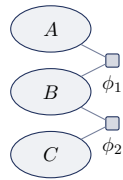
Exchangeable Factors

- Exchangeable factors encode equivalent underlying functions
  - However: Potential tables are not necessarily identical,
  - but only identical up to a permutation of arguments.



A	B	$\phi_1(A,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

C	B	$\phi_2(C,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

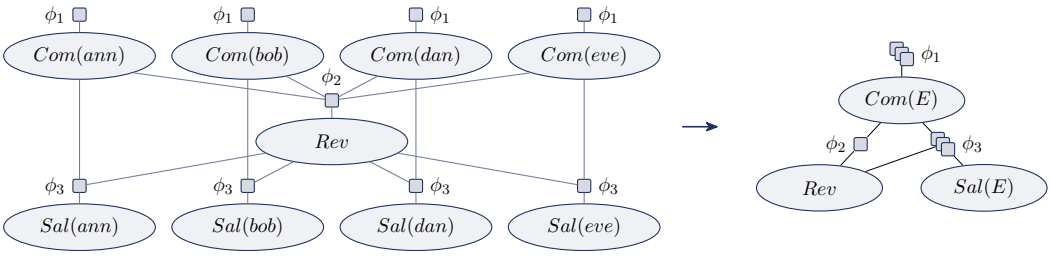


A	B	$\phi_1(A,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

B	C	$\phi_2(B,C)$
high	high	φ_1
high	low	φ_3
low	high	φ_2
low	low	φ_4

The Advanced Colour Passing Algorithm (ACP) – Overview

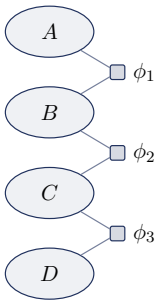
- ACP recognises commutative factors
- ACP detects exchangeable factors independent of argument orders
- ACP introduces logical variables



Malte Luttermann, Tanya Braun, Ralf Möller, and Marcel Gehrke (2024). »Colour Passing Revisited: Lifted Model Construction with Commutative Factors«. *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence (AAAI-2024)*. AAAI Press, pp. 20500–20507.

The Advanced Colour Passing Algorithm – Example Run

1. Check for exchangeable factors and rearrange arguments if necessary



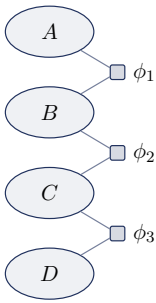
A	B	$\phi_1(A,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

$C \leftrightarrow D$		$\phi_3(C,D)$
high	high	φ_1
high	low	φ_3
low	high	φ_2
low	low	φ_4

B	C	$\phi_2(B,C)$
high	high	φ_5
high	low	φ_6
low	high	φ_6
low	low	φ_7

The Advanced Colour Passing Algorithm – Example Run

1. Check for exchangeable factors and rearrange arguments if necessary



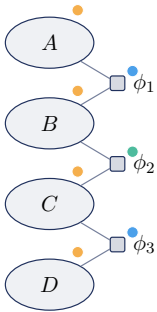
A	B	$\phi_1(A,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

D	C	$\phi_3(D,C)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

B	C	$\phi_2(B,C)$
high	high	φ_5
high	low	φ_6
low	high	φ_6
low	low	φ_7

The Advanced Colour Passing Algorithm – Example Run

2. Assign initial colours



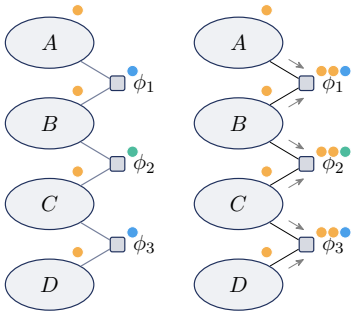
A	B	$\phi_1(A,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

D	C	$\phi_3(D,C)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

B	C	$\phi_2(B,C)$
high	high	φ_5
high	low	φ_6
low	high	φ_6
low	low	φ_7

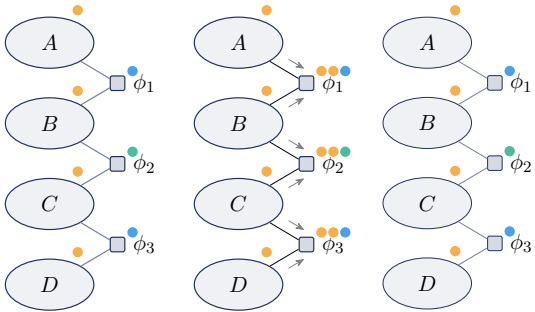
The Advanced Colour Passing Algorithm – Example Run

3. Pass colours from variable nodes to factor nodes



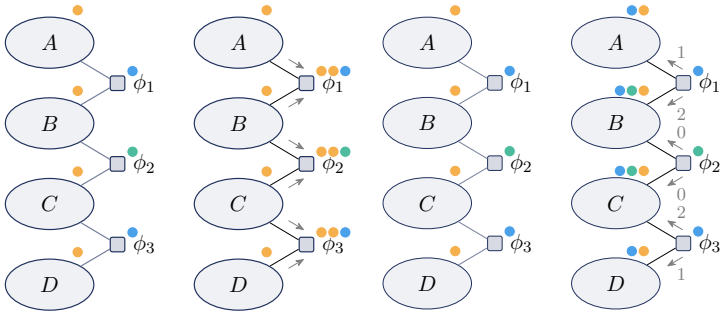
The Advanced Colour Passing Algorithm – Example Run

4. Recolour factor nodes



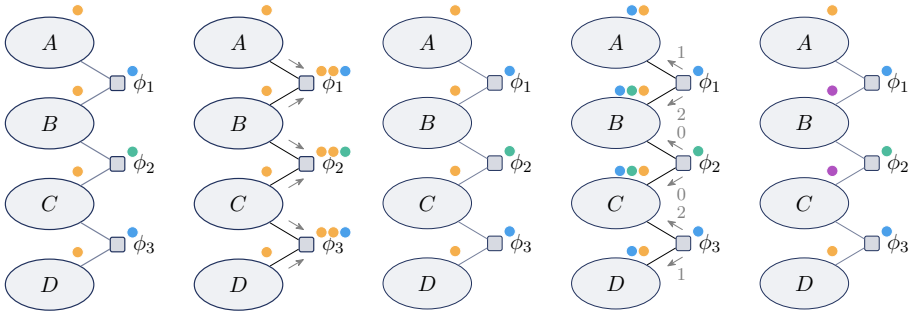
The Advanced Colour Passing Algorithm – Example Run

5. Pass colours from factor nodes to variable nodes (omit positions if commutative)



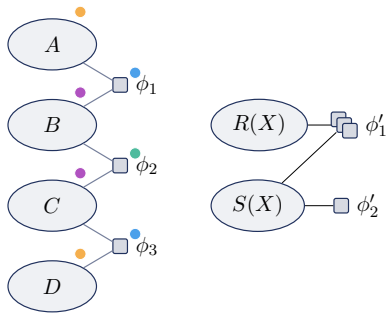
The Advanced Colour Passing Algorithm – Example Run

6. Recolour variable nodes



The Advanced Colour Passing Algorithm – Example Run

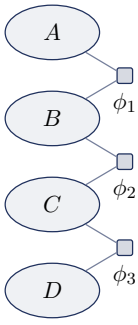
7. Introduce logical variables



$R(X)$	$S(X)$	$\phi'_1(R(X), S(X))$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

$\#_X[S(X)]$	$\phi'_2(\#_X[S(X)])$
$[2, 0]$	φ_5
$[1, 1]$	φ_6
$[0, 2]$	φ_7

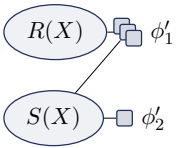
The Advanced Colour Passing Algorithm – Compression Result



A	B	$\phi_1(A,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

B	C	$\phi_2(B,C)$
high	high	φ_5
high	low	φ_6
low	high	φ_6
low	low	φ_7

C	D	$\phi_3(C,D)$
high	high	φ_1
high	low	φ_3
low	high	φ_2
low	low	φ_4

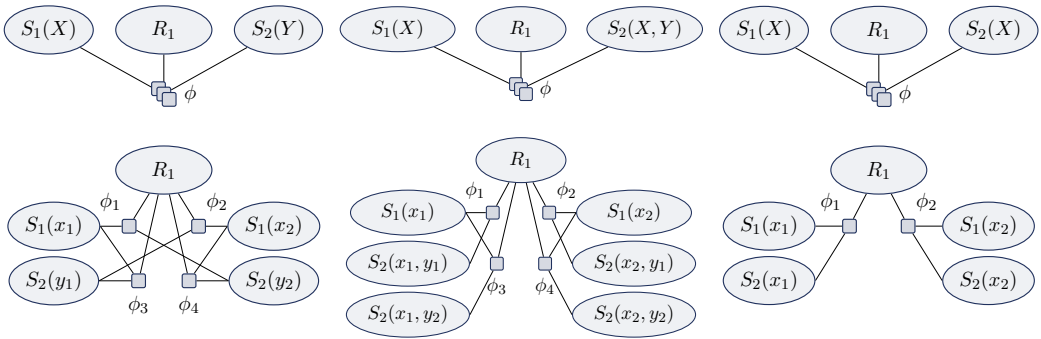


$R(X)$	$S(X)$	$\phi_1'(R(X),S(X))$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

$\#_X[S(X)]$	$\phi_2'(\#_X[S(X)])$
$[2,0]$	φ_5
$[1,1]$	φ_6
$[0,2]$	φ_7

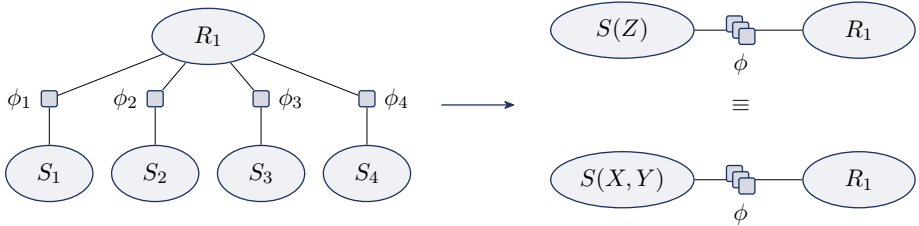
Introduction of Logical Variables I

- Introduction of logical variables to match groundings
- Illustration of scenarios (distinct, binary, shared) in the domain-liftable fragment:



Introduction of Logical Variables II

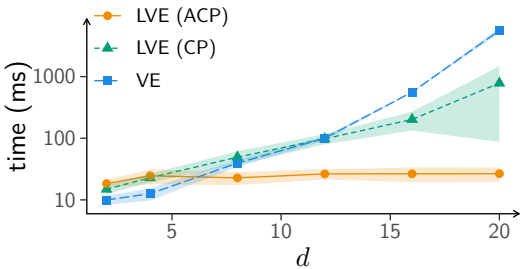
- The introduction of logical variables can be ambiguous (in the shared case)
- Example: Group of $|\mathbf{\Phi}| = 4$ exchangeable factors, matched either by $|\mathrm{dom}(Z)| = 4$ or by $|\mathrm{dom}(X)| \cdot |\mathrm{dom}(Y)| = 2 \cdot 2$

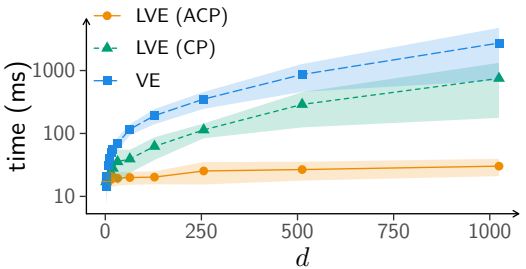


Both models ground to the same factor graph, i.e., they encode equivalent semantics.

The Advanced Colour Passing Algorithm – Experiments

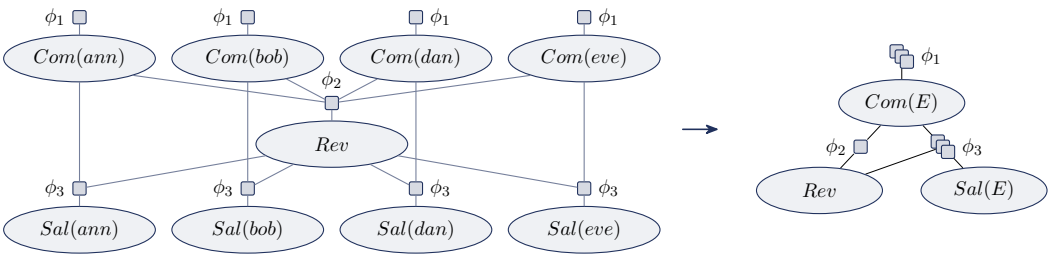
- Comparison of run times for lifted inference
- Left: Factor graphs with 1 commutative factor (no permuted argument orders)
- Right: Factor graphs where the arguments of 3 percent of the factors are permuted (no commutative factors)





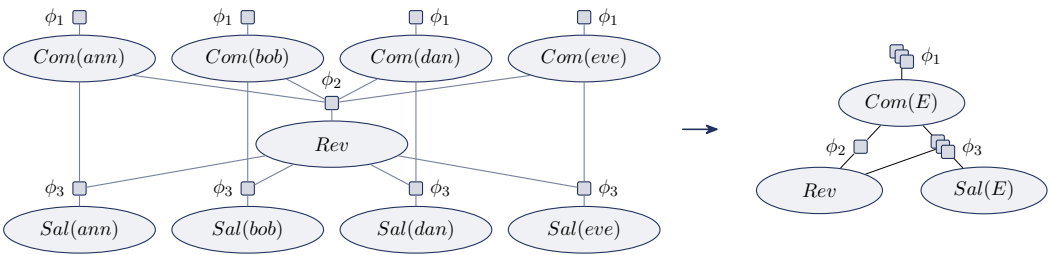
Advanced Colour Passing – Summary

1. Identification and compression of commutative factors
2. Identification and compression of exchangeable factors
3. Introduction of logical variables for the class of domain-liftable models



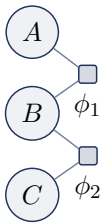
Advanced Colour Passing – Summary

1. Identification and compression of commutative factors ⇒ Next: Efficient realisation
2. Identification and compression of exchangeable factors ⇒ Next: Efficient realisation
3. Introduction of logical variables for the class of domain-liftable models



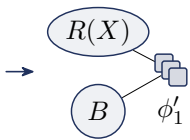
Efficient Detection of Exchangeable Factors I

- Goal: Efficiently detect exchangeable (i.e., equivalent) factors
- Exchangeable factors encode an equivalent probability distribution



A	B	$\phi_1(A,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

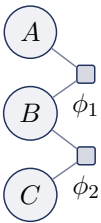
C	B	$\phi_2(C,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4



$R(X)$	B	$\phi_1'(R(X),B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

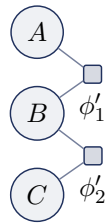
Efficient Detection of Exchangeable Factors II

- Goal: Efficiently detect exchangeable (i.e., equivalent) factors
- Exchangeable factors encode equivalent underlying functions



A	B	$\phi_1(A,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

C	B	$\phi_2(C,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4



A	B	$\phi'_1(A,B)$
high	high	φ_1
high	low	φ_2
low	high	φ_3
low	low	φ_4

B	C	$\phi'_2(B,C)$
high	high	φ_1
high	low	φ_3
low	high	φ_2
low	low	φ_4

Efficient Detection of Exchangeable Factors – Overview

Previously:

- Advanced Colour Passing algorithm to compress a factor graph
- Number of iterations in the worst-case for a factor with n arguments: $O(n!)$

Efficient Detection of Exchangeable Factors – Overview

Previously:

- Advanced Colour Passing algorithm to compress a factor graph
- Number of iterations in the worst-case for a factor with n arguments: $O(n!)$

Next:

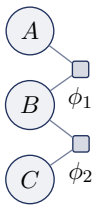
- Theoretical guarantees: *Buckets* to avoid iterating over permutations if possible
- Practical algorithm: Detection of Exchangeable Factors (DEFT) algorithm
- Number of iterations in the worst-case for a factor with n arguments: $O(d)$ with $d \ll n!$ in many practical settings

---

Malte Luttermann, Johann Machemer, and Marcel Gehrke (2024b). »Efficient Detection of Exchangeable Factors in Factor Graphs«. *Proceedings of the Thirty-Seventh International Florida Artificial Intelligence Research Society Conference (FLAIRS-2024)*. **Best Student Paper**. Florida Online Journals.

Buckets

- Buckets count the occurrences of range values in an assignment
- Each potential belongs to *exactly one* bucket
- Each bucket contains *at least one* potential



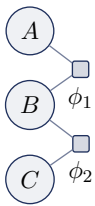
A	B	$\phi_1(A,B)$	b
high	high	φ_1	$[2,0]$
high	low	φ_2	$[1,1]$
low	high	φ_3	$[1,1]$
low	low	φ_4	$[0,2]$

B	C	$\phi_2(B,C)$	b
high	high	φ_1	$[2,0]$
high	low	φ_3	$[1,1]$
low	high	φ_2	$[1,1]$
low	low	φ_4	$[0,2]$

b	$\phi_1(b)$	$\phi_2(b)$
$[2,0]$	$\{\varphi_1\}$	$\{\varphi_1\}$
$[1,1]$	$\{\varphi_2,\varphi_3\}$	$\{\varphi_3,\varphi_2\}$
$[0,2]$	$\{\varphi_4\}$	$\{\varphi_4\}$

Properties of Buckets I

- If at least one bucket is mapped to different multisets of values by ϕ_1 and ϕ_2, then ϕ_1 and ϕ_2 are not exchangeable



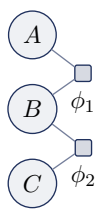
A	B	$\phi_1(A,B)$	b
high	high	φ_1	$[2,0]$
high	low	φ_2	$[1,1]$
low	high	φ_3	$[1,1]$
low	low	φ_4	$[0,2]$

B	C	$\phi_2(B,C)$	b
high	high	φ_1	$[2,0]$
high	low	φ_3	$[1,1]$
low	high	φ_2	$[1,1]$
low	low	φ_4	$[0,2]$

b	$\phi_1(b)$	$\phi_2(b)$
$[2,0]$	$\{\varphi_1\}$	$\{\varphi_1\}$
$[1,1]$	$\{\varphi_2,\varphi_3\}$	$\{\varphi_3,\varphi_2\}$
$[0,2]$	$\{\varphi_4\}$	$\{\varphi_4\}$

Properties of Buckets II

- Two factors are exchangeable iff there exists a permutation of their arguments such that each bucket is mapped to the same ordered multiset of values



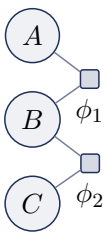
A	B	$\phi_1(A,B)$	b
high	high	φ_1	$[2,0]$
high	low	φ_2	$[1,1]$
low	high	φ_3	$[1,1]$
low	low	φ_4	$[0,2]$

B	C	$\phi_2(B,C)$	b
high	high	φ_1	$[2,0]$
high	low	φ_3	$[1,1]$
low	high	φ_2	$[1,1]$
low	low	φ_4	$[0,2]$

b	$\phi_1^{\succ}(b)$	$\phi_2^{\succ}(b)$
$[2,0]$	$\langle\varphi_1\rangle$	$\langle\varphi_1\rangle$
$[1,1]$	$\langle\varphi_2,\varphi_3\rangle$	$\langle\varphi_3,\varphi_2\rangle$
$[0,2]$	$\langle\varphi_4\rangle$	$\langle\varphi_4\rangle$

The Detection of Exchangeable Factors Algorithm – Example Run

- Iterate over buckets and search for possible rearrangements to obtain identically ordered multisets within each bucket



A	B	$\phi_1(A,B)$	b
high	high	φ_1	$[2,0]$
high	low	φ_2	$[1,1]$
low	high	φ_3	$[1,1]$
low	low	φ_4	$[0,2]$

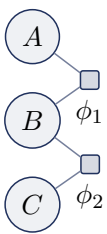
B	C	$\phi_2(B,C)$	b
high	high	φ_1	$[2,0]$
high	low	φ_3	$[1,1]$
low	high	φ_2	$[1,1]$
low	low	φ_4	$[0,2]$

b	$\phi_1^{\succ}(b)$	$\phi_2^{\succ}(b)$
$[2,0]$	$\langle\varphi_1\rangle$	$\langle\varphi_1\rangle$
$[1,1]$	$\langle\varphi_2,\varphi_3\rangle$	$\langle\varphi_3,\varphi_2\rangle$
$[0,2]$	$\langle\varphi_4\rangle$	$\langle\varphi_4\rangle$

Possible rearrangements:
$[2,0]$: $B \to \{1,2\}$, $C \to \{1,2\}$

The Detection of Exchangeable Factors Algorithm – Example Run

- Iterate over buckets and search for possible rearrangements to obtain identically ordered multisets within each bucket



A	B	$\phi_1(A,B)$	b
high	high	φ_1	$[2,0]$
high	low	φ_2	$[1,1]$
low	high	φ_3	$[1,1]$
low	low	φ_4	$[0,2]$

B	C	$\phi_2(B,C)$	b
high	high	φ_1	$[2,0]$
high	low	φ_3	$[1,1]$
low	high	φ_2	$[1,1]$
low	low	φ_4	$[0,2]$

b	$\phi_1^{\succ}(b)$	$\phi_2^{\succ}(b)$
$[2,0]$	$\langle\varphi_1\rangle$	$\langle\varphi_1\rangle$
$[1,1]$	$\langle\varphi_2,\varphi_3\rangle$	$\langle\varphi_3,\varphi_2\rangle$
$[0,2]$	$\langle\varphi_4\rangle$	$\langle\varphi_4\rangle$

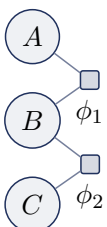
Possible rearrangements:

$[2,0]$: $B \to \{1,2\}$, $C \to \{1,2\}$

$[1,1]$: $B \to \{2\}$, $C \to \{1\}$

The Detection of Exchangeable Factors Algorithm – Example Run

- Iterate over buckets and search for possible rearrangements to obtain identically ordered multisets within each bucket



A	B	$\phi_1(A,B)$	b
high	high	φ_1	$[2,0]$
high	low	φ_2	$[1,1]$
low	high	φ_3	$[1,1]$
low	low	φ_4	$[0,2]$

B	C	$\phi_2(B,C)$	b
high	high	φ_1	$[2,0]$
high	low	φ_3	$[1,1]$
low	high	φ_2	$[1,1]$
low	low	φ_4	$[0,2]$

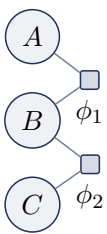
b	$\phi_1^{\succ}(b)$	$\phi_2^{\succ}(b)$
$[2,0]$	$\langle\varphi_1\rangle$	$\langle\varphi_1\rangle$
$[1,1]$	$\langle\varphi_2,\varphi_3\rangle$	$\langle\varphi_3,\varphi_2\rangle$
$[0,2]$	$\langle\varphi_4\rangle$	$\langle\varphi_4\rangle$

Possible rearrangements:

$[2,0]$: $B \to \{1,2\}$, $C \to \{1,2\}$
$[1,1]$: $B \to \{2\}$, $C \to \{1\}$
$[0,2]$: $B \to \{1,2\}$, $C \to \{1,2\}$

The Detection of Exchangeable Factors Algorithm – Example Run

- Iterate over buckets and search for possible rearrangements to obtain identically ordered multisets within each bucket



A	B	$\phi_1(A,B)$	b
high	high	φ_1	$[2,0]$
high	low	φ_2	$[1,1]$
low	high	φ_3	$[1,1]$
low	low	φ_4	$[0,2]$

B	C	$\phi_2(B,C)$	b
high	high	φ_1	$[2,0]$
high	low	φ_3	$[1,1]$
low	high	φ_2	$[1,1]$
low	low	φ_4	$[0,2]$

b	$\phi_1^{\succ}(b)$	$\phi_2^{\succ}(b)$
$[2,0]$	$\langle\varphi_1\rangle$	$\langle\varphi_1\rangle$
$[1,1]$	$\langle\varphi_2,\varphi_3\rangle$	$\langle\varphi_3,\varphi_2\rangle$
$[0,2]$	$\langle\varphi_4\rangle$	$\langle\varphi_4\rangle$

Possible rearrangements:

$[2,0]$: $B \to \{1,2\}$, $C \to \{1,2\}$

$[1,1]$: $B \to \{2\}$, $C \to \{1\}$

$[0,2]$: $B \to \{1,2\}$, $C \to \{1,2\}$

Intersection: $B \to \{2\}$, $C \to \{1\}$

Degree of Freedom (Theoretical Guarantees)

- Duplicate values in buckets increase complexity
- Degree of freedom of a bucket b:

$$\mathcal{F}(b) = \prod_{\varphi \in \text{unique}(\phi^{\succ}(b))} \text{count}(\phi^{\succ}(b), \varphi)!$$

- Degree of freedom of a factor ϕ:

$$\mathcal{F}(\phi) = \min_{b \in \{b | b \in \mathcal{B}(\phi) \wedge |\phi^{\succ}(b)| > 1\}} \mathcal{F}(b)$$

b	$\phi_1'^{\succ}(b)$	$\phi_2'^{\succ}(b)$
$[3,0]$	$\langle \varphi_1 \rangle$	$\langle \varphi_1 \rangle$
$[2,1]$	$\langle \varphi_2, \varphi_3, \varphi_5 \rangle$	$\langle \varphi_3, \varphi_5, \varphi_2 \rangle$
$[1,2]$	$\langle \varphi_4, \varphi_6, \varphi_6 \rangle$	$\langle \varphi_6, \varphi_4, \varphi_6 \rangle$
$[0,3]$	$\langle \varphi_7 \rangle$	$\langle \varphi_7 \rangle$

Degree of Freedom (Theoretical Guarantees)

- Duplicate values in buckets increase complexity
- Degree of freedom of a bucket b:

$$\mathcal{F}(b) = \prod_{\varphi \in \text{unique}(\phi^{\succ}(b))} \text{count}(\phi^{\succ}(b), \varphi)!$$

b	$\phi_1'^{\succ}(b)$	$\phi_2'^{\succ}(b)$
$[3,0]$	$\langle \varphi_1 \rangle$	$\langle \varphi_1 \rangle$
$[2,1]$	$\langle \varphi_2, \varphi_3, \varphi_5 \rangle$	$\langle \varphi_3, \varphi_5, \varphi_2 \rangle$
$[1,2]$	$\langle \varphi_4, \varphi_6, \varphi_6 \rangle$	$\langle \varphi_6, \varphi_4, \varphi_6 \rangle$
$[0,3]$	$\langle \varphi_7 \rangle$	$\langle \varphi_7 \rangle$

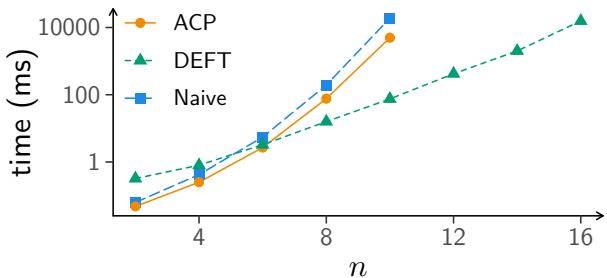
- Degree of freedom of a factor ϕ:

$$\mathcal{F}(\phi) = \min_{b \in \{b | b \in \mathcal{B}(\phi) \wedge |\phi^{\succ}(b)| > 1\}} \mathcal{F}(b)$$

- The number of table comparisons needed to check whether ϕ_1 and ϕ_2 are exchangeable is in $O(d)$, where $d = \min\{\mathcal{F}(\phi_1), \mathcal{F}(\phi_2)\}$ (i.e., d is factorial)
- The degree of freedom is upper-bounded by $n!$ (i.e., $d \leq n!$)
- In many practical settings it even holds that $d \ll n!$

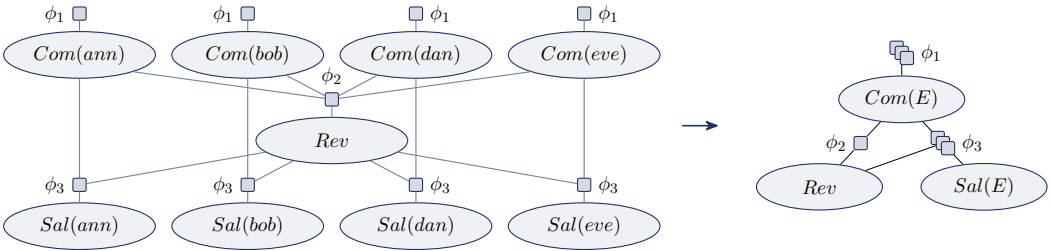
The Detection of Exchangeable Factors Algorithm – Experiments

- Comparison of run times of DEFT, ACP, and a »naive« approach
- Average over exchangeable and non-exchangeable factors with a proportion of identical potentials in $\{0.0, 0.1, 0.2, 0.5, 0.8, 0.9, 1.0\}$
- Timeout after 30 minutes per instance



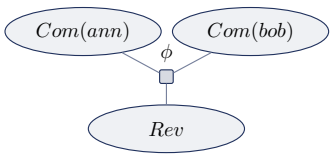
Efficient Detection of Exchangeable Factors – Summary

- Problem of detecting exchangeable factors efficiently solved
- Upper bound on the number of table comparisons depending on the number of duplicate potential values within the buckets of the factors
- The DEFT algorithm exploits this upper bound effectively in practice



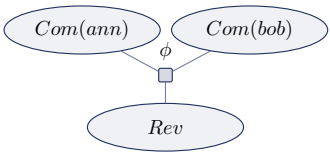
Efficient Detection of Commutative Factors I

- Goal: Efficiently compute subsets of commutative arguments
- Commutative factors encode symmetric functions

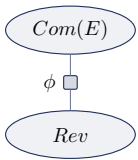


$Com(ann)$	$Com(bob)$	Rev	$\phi(Com(ann), Com(bob), Rev)$
high	high	high	φ_1
high	high	low	φ_4
high	low	high	φ_2
high	low	low	φ_5
low	high	high	φ_2
low	high	low	φ_5
low	low	high	φ_3
low	low	low	φ_6

Efficient Detection of Commutative Factors II



$Com(ann)$	$Com(bob)$	Rev	$\phi(Com(ann), Com(bob), Rev)$
high	high	high	φ_1
high	high	low	φ_4
high	low	high	φ_2
high	low	low	φ_5
low	high	high	φ_2
low	high	low	φ_5
low	low	high	φ_3
low	low	low	φ_6



$\#_E[Com(E)]$	Rev	$\phi(\#_E[Com(E)], Rev)$
$[2,0]$	high	φ_1
$[1,1]$	high	φ_2
$[0,2]$	high	φ_3
$[2,0]$	low	φ_4
$[1,1]$	low	φ_5
$[0,2]$	low	φ_6

Efficient Detection of Commutative Factors – Overview

Previously:

- Iteration over all subsets of arguments to find commutative arguments
- Number of iterations for a factor with n arguments is in $O(2^n)$

Efficient Detection of Commutative Factors – Overview

Previously:

- Iteration over all subsets of arguments to find commutative arguments
- Number of iterations for a factor with n arguments is in $O(2^n)$

Next:

- Theoretical guarantees: *Buckets* to avoid iterating over all subsets
- Practical algorithm: Detection of Commutative Factors (DECOR) algorithm

---

Malte Luttermann, Johann Machemer, and Marcel Gehrke (2024a). »Efficient Detection of Commutative Factors in Factor Graphs«. *Proceedings of the Twelfth International Conference on Probabilistic Graphical Models (PGM-2024)*. PMLR, pp. 38–56.

Malte Luttermann, Ralf Möller, and Marcel Gehrke (2026). »On the Detection of Commutative Factors in Factor Graphs: Necessary and Sufficient Conditions«. *Proceedings of the Thirteenth International Conference on Probabilistic Graphical Models (PGM-2026)*. Forthcoming.

Buckets

- Buckets count the occurrences of range values in an assignment
- Each potential belongs to *exactly one* bucket
- Each bucket contains *at least one* potential

A	B	R	$\phi(A,B,R)$	b
high	high	high	φ_1	$[3,0]$
high	high	low	φ_4	$[2,1]$
high	low	high	φ_2	$[2,1]$
high	low	low	φ_5	$[1,2]$
low	high	high	φ_2	$[2,1]$
low	high	low	φ_5	$[1,2]$
low	low	high	φ_3	$[1,2]$
low	low	low	φ_6	$[0,3]$

b	$\phi(b)$
$[3,0]$	$\langle\varphi_1\rangle$
$[2,1]$	$\langle\varphi_4,\varphi_2,\varphi_2\rangle$
$[1,2]$	$\langle\varphi_5,\varphi_5,\varphi_3\rangle$
$[0,3]$	$\langle\varphi_6\rangle$

Properties of Buckets I

For any subset $\boldsymbol{S}$ of commutative arguments:

- $|\boldsymbol{S}| \leq \min_{b \in \{b \mid b \in \mathcal{B}(\phi) \wedge |\phi(b)|>1\}} \max_{\varphi \in \phi(b)} \mathrm{count}(\phi(b), \varphi)$, i.e.,
- in each bucket b with $|\phi(b)| > 1$, there is a potential occurring at least $|\boldsymbol{S}|$ times

A	B	R	$\phi(A,B,R)$	b
high	high	high	φ_1	$[3,0]$
high	high	low	φ_4	$[2,1]$
high	low	high	φ_2	$[2,1]$
high	low	low	φ_5	$[1,2]$
low	high	high	φ_2	$[2,1]$
low	high	low	φ_5	$[1,2]$
low	low	high	φ_3	$[1,2]$
low	low	low	φ_6	$[0,3]$

b	$\phi(b)$
$[3,0]$	$\langle \varphi_1 \rangle$
$[2,1]$	$\langle \varphi_4, \varphi_2, \varphi_2 \rangle$
$[1,2]$	$\langle \varphi_5, \varphi_5, \varphi_3 \rangle$
$[0,3]$	$\langle \varphi_6 \rangle$

Properties of Buckets II

For a group of identical potentials in a bucket:

- Intersection of their corresponding assignments yields candidates
- E.g., for φ_2's in $[2, 1]$: $(\text{high}, \text{low}, \text{high}) \cap (\text{low}, \text{high}, \text{high}) = (\emptyset, \emptyset, \text{high})$

A	B	R	$\phi(A, B, R)$	b
high	high	high	φ_1	$[3, 0]$
high	high	low	φ_4	$[2, 1]$
high	low	high	φ_2	$[2, 1]$
high	low	low	φ_5	$[1, 2]$
low	high	high	φ_2	$[2, 1]$
low	high	low	φ_5	$[1, 2]$
low	low	high	φ_3	$[1, 2]$
low	low	low	φ_6	$[0, 3]$

b	$\phi(b)$
$[3, 0]$	$\langle \varphi_1 \rangle$
$[2, 1]$	$\langle \varphi_4, \varphi_2, \varphi_2 \rangle$
$[1, 2]$	$\langle \varphi_5, \varphi_5, \varphi_3 \rangle$
$[0, 3]$	$\langle \varphi_6 \rangle$

The Detection of Commutative Factors Algorithm – Example Run

- Iterate over buckets and compute candidates for commutative arguments

A	B	R	$\phi(A,B,R)$	b
high	high	high	φ_1	$[3,0]$
high	high	low	φ_4	$[2,1]$
high	low	high	φ_2	$[2,1]$
high	low	low	φ_5	$[1,2]$
low	high	high	φ_2	$[2,1]$
low	high	low	φ_5	$[1,2]$
low	low	high	φ_3	$[1,2]$
low	low	low	φ_6	$[0,3]$

b	$\phi(b)$
$[3,0]$	$\langle\varphi_1\rangle$
$[2,1]$	$\langle\varphi_4,\varphi_2,\varphi_2\rangle$
$[1,2]$	$\langle\varphi_5,\varphi_5,\varphi_3\rangle$
$[0,3]$	$\langle\varphi_6\rangle$

Initial candidates: $\{\{A,B,R\}\}$

The Detection of Commutative Factors Algorithm – Example Run

- Iterate over buckets and compute candidates for commutative arguments

A	B	R	$\phi(A,B,R)$	b
high	high	high	φ_1	$[3,0]$
high	high	low	φ_4	$[2,1]$
high	low	high	φ_2	$[2,1]$
high	low	low	φ_5	$[1,2]$
low	high	high	φ_2	$[2,1]$
low	high	low	φ_5	$[1,2]$
low	low	high	φ_3	$[1,2]$
low	low	low	φ_6	$[0,3]$

b	$\phi(b)$
$[3,0]$	$\langle\varphi_1\rangle$
$[2,1]$	$\langle\varphi_4,\varphi_2,\varphi_2\rangle$
$[1,2]$	$\langle\varphi_5,\varphi_5,\varphi_3\rangle$
$[0,3]$	$\langle\varphi_6\rangle$

Initial candidates: $\{\{A,B,R\}\}$

$[3,0]$ ($|\phi(b)| < 2$): $\{\{A,B,R\}\}$

The Detection of Commutative Factors Algorithm – Example Run

- Iterate over buckets and compute candidates for commutative arguments

A	B	R	$\phi(A,B,R)$	b
high	high	high	φ_1	$[3,0]$
high	high	low	φ_4	$[2,1]$
high	low	high	φ_2	$[2,1]$
high	low	low	φ_5	$[1,2]$
low	high	high	φ_2	$[2,1]$
low	high	low	φ_5	$[1,2]$
low	low	high	φ_3	$[1,2]$
low	low	low	φ_6	$[0,3]$

b	$\phi(b)$
$[3,0]$	$\langle\varphi_1\rangle$
$[2,1]$	$\langle\varphi_4,\varphi_2,\varphi_2\rangle$
$[1,2]$	$\langle\varphi_5,\varphi_5,\varphi_3\rangle$
$[0,3]$	$\langle\varphi_6\rangle$

Initial candidates: $\{\{A,B,R\}\}$
$[3,0]$ ($|\phi(b)| < 2$): $\{\{A,B,R\}\}$
$[2,1]$: $\{\{A,B\}\}$

$$(\text{high},\text{low},\text{high}) \cap (\text{low},\text{high},\text{high}) = (\emptyset,\emptyset,\text{high})$$

The Detection of Commutative Factors Algorithm – Example Run

- Iterate over buckets and compute candidates for commutative arguments

A	B	R	$\phi(A,B,R)$	b
high	high	high	φ_1	$[3,0]$
high	high	low	φ_4	$[2,1]$
high	low	high	φ_2	$[2,1]$
high	low	low	φ_5	$[1,2]$
low	high	high	φ_2	$[2,1]$
low	high	low	φ_5	$[1,2]$
low	low	high	φ_3	$[1,2]$
low	low	low	φ_6	$[0,3]$

b	$\phi(b)$
$[3,0]$	$\langle\varphi_1\rangle$
$[2,1]$	$\langle\varphi_4,\varphi_2,\varphi_2\rangle$
$[1,2]$	$\langle\varphi_5,\varphi_5,\varphi_3\rangle$
$[0,3]$	$\langle\varphi_6\rangle$

Initial candidates: $\{\{A,B,R\}\}$
$[3,0]$ ($|\phi(b)| < 2$): $\{\{A,B,R\}\}$
$[2,1]$: $\{\{A,B\}\}$
$[1,2]$: $\{\{A,B\}\}$

$$(\text{high},\text{low},\text{low}) \cap (\text{low},\text{high},\text{low}) = (\emptyset,\emptyset,\text{low})$$

The Detection of Commutative Factors Algorithm – Example Run

- Iterate over buckets and compute candidates for commutative arguments

A	B	R	$\phi(A,B,R)$	b
high	high	high	φ_1	$[3,0]$
high	high	low	φ_4	$[2,1]$
high	low	high	φ_2	$[2,1]$
high	low	low	φ_5	$[1,2]$
low	high	high	φ_2	$[2,1]$
low	high	low	φ_5	$[1,2]$
low	low	high	φ_3	$[1,2]$
low	low	low	φ_6	$[0,3]$

b	$\phi(b)$
$[3,0]$	$\langle\varphi_1\rangle$
$[2,1]$	$\langle\varphi_4,\varphi_2,\varphi_2\rangle$
$[1,2]$	$\langle\varphi_5,\varphi_5,\varphi_3\rangle$
$[0,3]$	$\langle\varphi_6\rangle$

Initial candidates: $\{\{A,B,R\}\}$
$[3,0]$ ($|\phi(b)| < 2$): $\{\{A,B,R\}\}$
$[2,1]$: $\{\{A,B\}\}$
$[1,2]$: $\{\{A,B\}\}$
$[0,3]$ ($|\phi(b)| < 2$): $\{\{A,B\}\}$

Buckets Are Necessary, But Not Sufficient

- The bucket condition is only *necessary*: A candidate set obtained from the buckets is not necessarily commutative
- Hence a verification step is required after computing the candidates

Counterexample ($n = 4$)

Let $\phi(\text{high}, \text{high}, \text{low}, \text{low}) = \phi(\text{low}, \text{low}, \text{high}, \text{high}) = \varphi_1$ and $\phi(\boldsymbol{r}) = \varphi_2$ for all other assignments $\boldsymbol{r}$. The buckets propose the candidate set $\{R_1, R_2, R_3, R_4\}$ — yet it is *not* commutative, e.g.,

$$\phi(\text{high}, \text{high}, \text{low}, \text{low}) = \varphi_1 \neq \varphi_2 = \phi(\text{low}, \text{high}, \text{high}, \text{low}),$$

although both assignments are permutations of one another.

The Detection of Commutative Factors Algorithm – Complexity I

- Avoids »naive« iteration over all 2^n subsets of arguments
- Time complexity is upper-bounded depending on the number of groups of identical potentials in the buckets
- Additional verification step required to check whether the candidates are indeed commutative arguments

b	$\phi(b)$
$[3,0]$	$\langle\varphi_1\rangle$
$[2,1]$	$\langle\varphi_4,\varphi_2,\varphi_2\rangle$
$[1,2]$	$\langle\varphi_5,\varphi_5,\varphi_3\rangle$
$[0,3]$	$\langle\varphi_6\rangle$

The Detection of Commutative Factors Algorithm – Complexity II

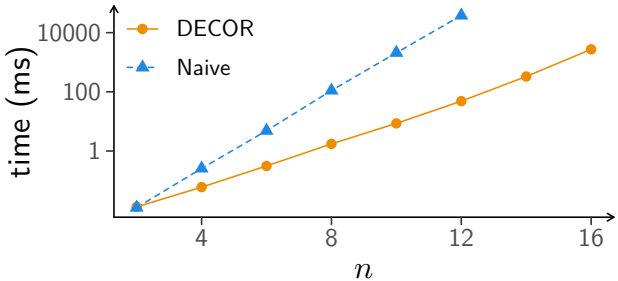
In practice, there are two possible scenarios:

1. A factor contains commutative arguments
   - Potential values are likely to contain a single group of duplicates
2. A factor contains no commutative arguments
   - Potential values are likely to be distinct

b	$\phi(b)$
$[3,0]$	$\langle\varphi_1\rangle$
$[2,1]$	$\langle\varphi_4,\varphi_2,\varphi_2\rangle$
$[1,2]$	$\langle\varphi_5,\varphi_5,\varphi_3\rangle$
$[0,3]$	$\langle\varphi_6\rangle$

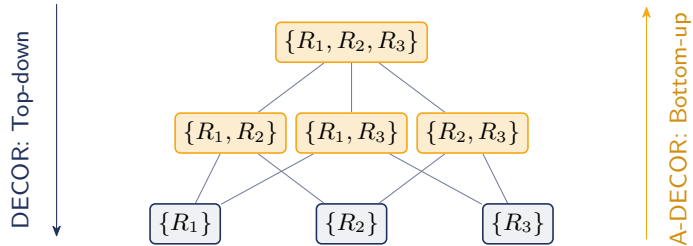
The Detection of Commutative Factors Algorithm – Experiments

- Comparison of run times of DECOR and the »naive« approach
- Average over factors with $k \in \{0, 2, \lfloor \frac{n}{2} \rfloor, n-1, n\}$ commutative arguments
- Timeout after five minutes per instance



A Bottom-Up Alternative: Apriori-Style DECOR (A-DECOR)

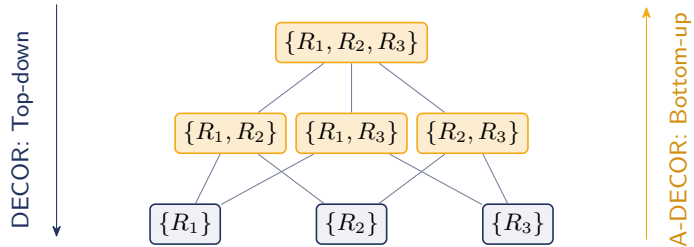
- DECOR searches top-down: Start with all arguments and narrow down
- Alternative: Search bottom-up, starting from commutative pairs



Malte Luttermann, Ralf Möller, and Marcel Gehrke (2026). »On the Detection of Commutative Factors in Factor Graphs: Necessary and Sufficient Conditions«. *Proceedings of the Thirteenth International Conference on Probabilistic Graphical Models (PGM-2026)*. Forthcoming.

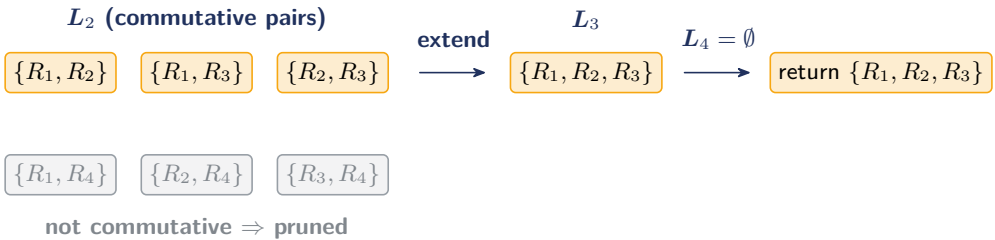
A Bottom-Up Alternative: Apriori-Style DECOR (A-DECOR)

- Two structural properties enable pruning:
  - *Downward closure*: If $\boldsymbol{C}$ is commutative, so is every subset of $\boldsymbol{C}$
  - *Transitivity*: $\boldsymbol{C}$ is commutative $\iff$ every pair in $\boldsymbol{C}$ is commutative



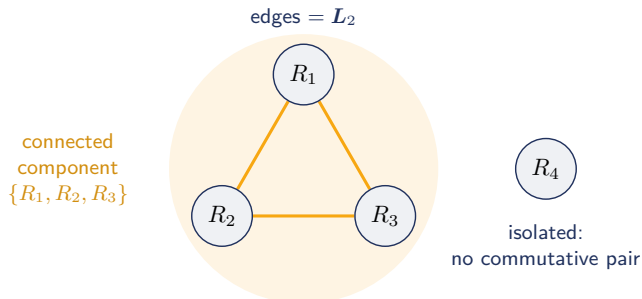
A-DECOR – Example Run

- A-DECOR enumerates commutative subsets level by level
- Extend a subset by an argument only if all newly formed pairs are in $\boldsymbol{L_2}$



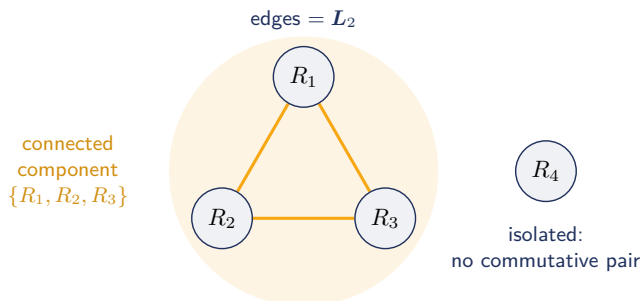
Efficient Realisation of A-DECOR via Connected Components I

- *Transitivity*: If $\boldsymbol{C}_1$ and $\boldsymbol{C}_2$ are commutative and $\boldsymbol{C}_1 \cap \boldsymbol{C}_2 \neq \emptyset$, then $\boldsymbol{C}_1 \cup \boldsymbol{C}_2$ is commutative
- Merging matches computing connected components in the commutativity graph
- Efficient merging via union-find data structure



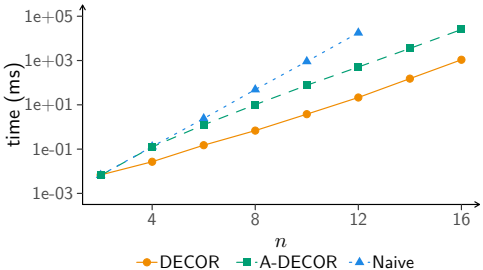
Efficient Realisation of A-DECOR via Connected Components II

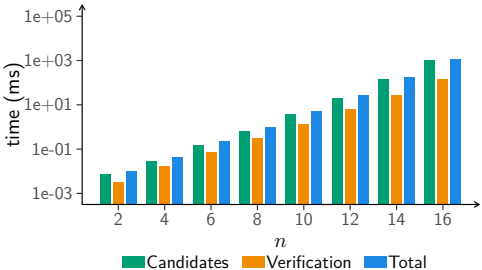
- Advantage over original DECOR algorithm:
  - Only $O(n^2)$ commutativity checks in the worst case (vs. $O(2^n)$ for DECOR — even though this is a highly pessimistic bound)
  - But: $\Omega(n^2)$ commutativity checks (vs. $\Omega(1)$ for DECOR)



A-DECOR – Experiments

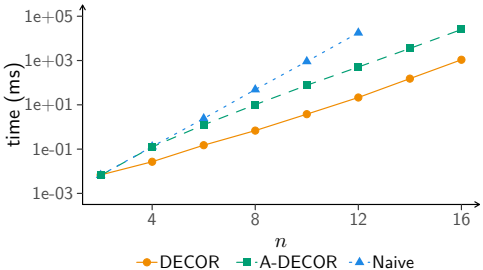
- Run times of DECOR, A-DECOR, and the »naive« approach
- Average over factors with k (varied between 0 and n) commutative arguments
- Timeout after five minutes per instance

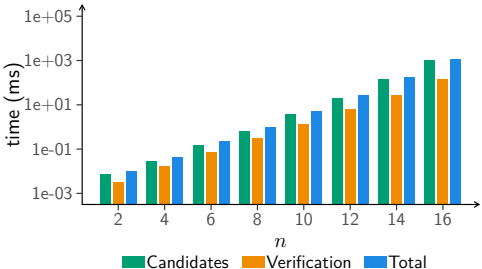




A-DECOR – Experiments

- Run times of DECOR, A-DECOR, and the »naive« approach
- Average over factors with k (varied between 0 and n) commutative arguments
- Timeout after five minutes per instance

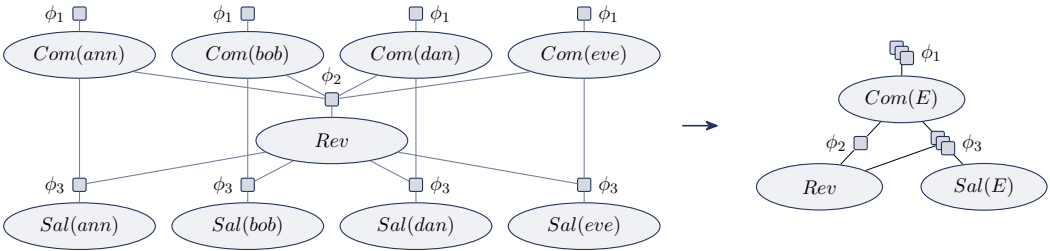




- Takeaway: A-DECOR's tighter $O(n^2)$ bound does *not* win in practice — it always performs $\Omega(n^2)$ full-table scans (DECOR usually only a few)

Efficient Detection of Commutative Factors – Summary

- Problem of efficiently detecting commutative factors solved
- Upper bound on the number of iterations depending on the number and size of groups of identical potential values within the buckets of a factor
- The DECOR algorithm effectively exploits this upper bound in practice



Agenda

1. Introduction and background [Tanya]
   - Motivation, use cases
   - Probabilistic (relational) models
   - Lifted probabilistic inference
2. Compressing models [Malte]
   - The basics of colour passing
   - Recent advances in colour passing
3. Accuracy considerations [Jan]
   - Hierarchical colour passing
   - Geometric interpretation
4. Summary and future directions [Tanya]

Bibliography I

- Babak Ahmadi, Kristian Kersting, Martin Mladenov, and Sriraam Natarajan (2013). »Exploiting Symmetries for Scaling Loopy Belief Propagation and Relational Training«. *Machine Learning* 92, pp. 91–132.
- Kristian Kersting, Babak Ahmadi, and Sriraam Natarajan (2009). »Counting Belief Propagation«. *Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI-2009)*. AUAI Press, pp. 277–284.
- Malte Luttermann, Tanya Braun, Ralf Möller, and Marcel Gehrke (2024). »Colour Passing Revisited: Lifted Model Construction with Commutative Factors«. *Proceedings of the Thirty-Eighth AAAI Conference on Artificial Intelligence (AAAI-2024)*. AAAI Press, pp. 20500–20507.

Bibliography II

- Malte Luttermann, Johann Machemer, and Marcel Gehrke (2024a). »Efficient Detection of Commutative Factors in Factor Graphs«. *Proceedings of the Twelfth International Conference on Probabilistic Graphical Models (PGM-2024)*. PMLR, pp. 38–56.
- Malte Luttermann, Johann Machemer, and Marcel Gehrke (2024b). »Efficient Detection of Exchangeable Factors in Factor Graphs«. *Proceedings of the Thirty-Seventh International Florida Artificial Intelligence Research Society Conference (FLAIRS-2024)*. **Best Student Paper**. Florida Online Journals.
- Malte Luttermann, Ralf Möller, and Marcel Gehrke (2026). »On the Detection of Commutative Factors in Factor Graphs: Necessary and Sufficient Conditions«. *Proceedings of the Thirteenth International Conference on Probabilistic Graphical Models (PGM-2026)*. Forthcoming.