JOHANNES KEPLER
UNIVERSITY LINZ

Boolean Quantifier Shifting as an Optimization Problem

Adrian Rebola-Pardo
July 1, 2024
QUANTIFY 2024, Nancy (France)

Some pictures by HashiCorp, Inc. and The Walt Disney Company.

JOHANNES KEPLER
UNIVERSITY LINZ
Altenberger Straße 69
4040 Linz, Austria
jku.at

The larger context

The work presented here is contained in:

Quantifier Shifting for Quantified Boolean Formulas Revisited
Simone Heisinger, Maximilian Heisinger, Adrian Rebola-Pardo, Martina Seidl

Today theoretical basis

Thursday 14:30 IJCAR talk by **Simone Heisinger**

Friday 17:10 IJCAR talk by **Maximilian Heisinger**

QBF solvers and prenex formulas

QBF solvers are decision procedures for the satisfiability problem on quantified Boolean formulas...

$$\exists a.\, (\forall x.\, a \vee \neg x) \wedge (\forall y.\, \exists b.\, y \vee (\neg a \wedge b))$$

QBF solvers and prenex formulas

QBF solvers are decision procedures for the satisfiability problem on quantified Boolean formulas...

$$\exists a. (\forall x. a \vee \neg x) \wedge (\forall y. \exists b. y \vee (\neg a \wedge b))$$

... on *prenex form*...

$$\exists a. \forall x. \forall y. \exists b. (a \vee \neg x) \wedge (y \vee (\neg a \wedge b))$$

QBF solvers and prenex formulas

QBF solvers are decision procedures for the satisfiability problem on quantified Boolean formulas...

$$\exists a.\, (\forall x.\, a \vee \neg x) \wedge (\forall y.\, \exists b.\, y \vee (\neg a \wedge b))$$

... on *prenex form*...

$$\exists a.\, \forall x.\, \forall y.\, \exists b.\, (a \vee \neg x) \wedge (y \vee (\neg a \wedge b))$$

... and in many cases in *prenex conjunctive normal form (PCNF)*...

$$\exists a.\, \forall x.\, \forall y.\, \exists b.\, (a \vee \neg x) \wedge (y \vee \neg a) \wedge (y \vee b)$$

Do prenexing methods matter?

real-world problem

Do prenexing methods matter?

$$\text{real-world problem} \xrightarrow{\text{encoder}} \text{QBF encoding}$$

Do prenexing methods matter?

$$
\begin{tikzcd}
\text{real-world problem} \arrow[d, "\text{encoder}"] \\
\text{QBF encoding} \arrow[d, "\text{quantifier shift technique}"] \\
\text{PCNF QBF encoding}
\end{tikzcd}
$$

Do prenexing methods matter?

real-world problem

↓ encoder

QBF encoding

↓ quantifier shift technique

PCNF QBF encoding

↓ QBF solver

solution

Do prenexing methods matter?

real-world problem

negligible ↓ encoder

QBF encoding

negligible ↓ quantifier shift technique

PCNF QBF encoding

expensive ↓ QBF solver

solution

Do prenexing methods matter?

real-world problem

negligible ↓ encoder

QBF encoding

negligible ↓ quantifier shift technique

PCNF QBF encoding

expensive ↓ QBF solver

solution

does the specific quantifier shift technique affect QBF solver runtime?

Do prenexing methods matter?

real-world problem

negligible $\downarrow$ encoder

QBF encoding

negligible $\downarrow$ quantifier shift technique

PCNF QBF encoding

we are only measuring this!

expensive $\downarrow$ QBF solver

solution

does the specific quantifier shift technique affect QBF solver runtime?

Do prenexing methods matter?

real-world problem

negligible ↓ encoder

QBF encoding

negligible ↓ quantifier shift technique

PCNF QBF encoding

expensive ↓ QBF solver

solution

... how does one shift quantifiers anyway?

we are only measuring this!

does the specific quantifier shift technique affect QBF solver runtime?

Quantifier shifting

We can assume the target QBF:

- is in *negation normal form* (i.e. negation only occurs in front of variables)
- follows the *Barendregt convention* (i.e. variables are renamed apart)

Quantifier shifting

We can assume the target QBF:

- is in *negation normal form* (i.e. negation only occurs in front of variables)
- follows the *Barendregt convention* (i.e. variables are renamed apart)

Quantifiers $\mathsf{Q} \in \{\forall, \exists\}$ can then be “pulled” across Q and connectives $\circ \in \{\wedge, \vee\}$:

$$(\mathsf{Q}x.\, \varphi) \circ \psi \equiv \varphi \circ (\mathsf{Q}x.\, \psi) \equiv \mathsf{Q}x.\, \varphi \circ \psi \qquad \mathsf{Q}x.\, \mathsf{Q}y.\, \varphi \equiv \mathsf{Q}y.\, \mathsf{Q}x.\, \varphi$$

Quantifier shifting

We can assume the target QBF:

- is in *negation normal form* (i.e. negation only occurs in front of variables)
- follows the *Barendregt convention* (i.e. variables are renamed apart)

Quantifiers $\mathsf{Q} \in \{\forall, \exists\}$ can then be “pulled” across Q and connectives $\circ \in \{\wedge, \vee\}$:

$$(\mathsf{Q}x.\, \varphi) \circ \psi \equiv \varphi \circ (\mathsf{Q}x.\, \psi) \equiv \mathsf{Q}x.\, \varphi \circ \psi \qquad \mathsf{Q}x.\, \mathsf{Q}y.\, \varphi \equiv \mathsf{Q}y.\, \mathsf{Q}x.\, \varphi$$

Lesson 1 Connective structure is irrelevant for quantifier shifting.

Quantifier shifting

We can assume the target QBF:

- is in *negation normal form* (i.e. negation only occurs in front of variables)
- follows the *Barendregt convention* (i.e. variables are renamed apart)

Quantifiers $\mathsf{Q} \in \{\forall, \exists\}$ can then be "pulled" across Q and connectives $\circ \in \{\wedge, \vee\}$:

$$(\mathsf{Q}x.\,\varphi) \circ \psi \equiv \varphi \circ (\mathsf{Q}x.\,\psi) \equiv \mathsf{Q}x.\,\varphi \circ \psi \qquad \mathsf{Q}x.\,\mathsf{Q}y.\,\varphi \equiv \mathsf{Q}y.\,\mathsf{Q}x.\,\varphi$$

Lesson 1 Connective structure is irrelevant for quantifier shifting.

Lesson 2 Only choices in distinct, branched quantifiers matter:

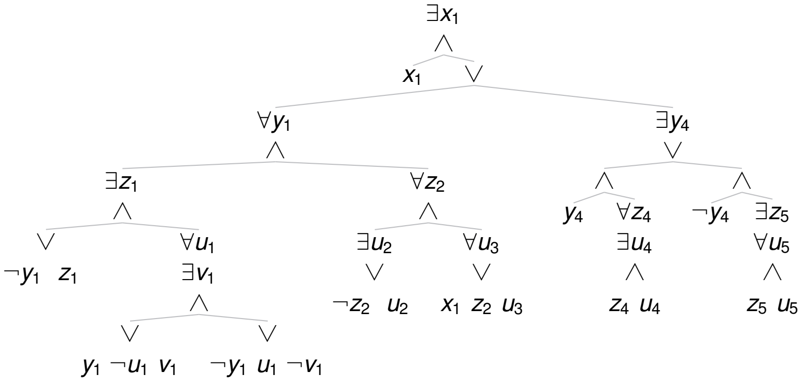
$$(\mathsf{Q}x.\,\varphi) \circ (\overline{\mathsf{Q}}y.\,\psi) \equiv \mathsf{Q}x.\,\overline{\mathsf{Q}}y.\,\varphi \circ \psi \equiv \overline{\mathsf{Q}}y.\,\mathsf{Q}x.\,\varphi \circ \psi$$

Quantifier shifting: how the sausage is made

$$\exists x_1 .\, x_1 \wedge ((\forall y_1 .\, (\exists z_1 .\, (\neg y_1 \vee z_1) \wedge \forall u_1 .\, \exists v_1 .\, (y_1 \vee \neg u_1 \vee v_1) \wedge (\neg y_1 \vee u_1 \vee \neg v_1)) \wedge (\forall z_2 .\, (\exists u_2 .\, \neg z_2 \vee u_2) \wedge (\forall u_3 .\, x_1 \vee z_2 \vee u_3))) \vee (\exists y_4 .\, (y_4 \wedge \forall z_4 .\, z_4 \wedge u_4) \vee (\neg y_4 \wedge \exists z_5 .\, \forall u_5 .\, z_5 \wedge u_5)))$$

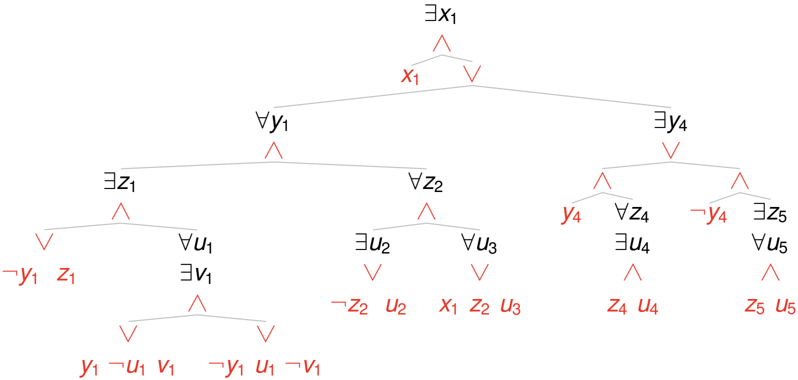
QBF formula φ

Quantifier shifting: how the sausage is made



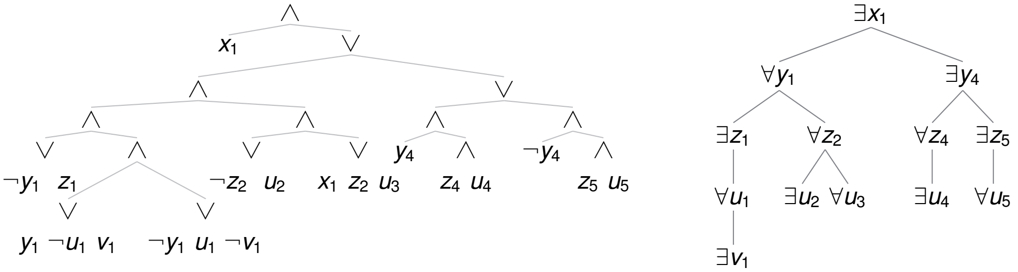
QBF formula φ

Quantifier shifting: how the sausage is made

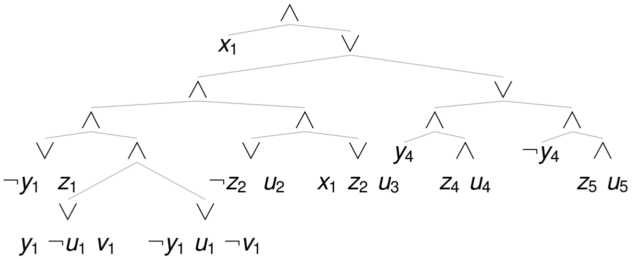


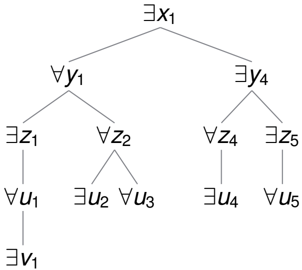
QBF formula φ

Quantifier shifting: how the sausage is made



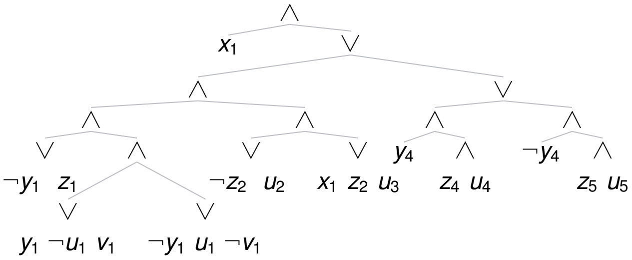
Quantifier shifting: how the sausage is made



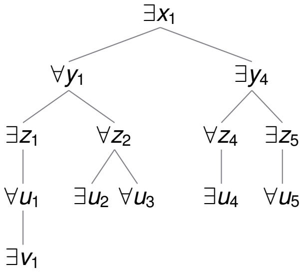


propositional skeleton φ^{psk}

Quantifier shifting: how the sausage is made

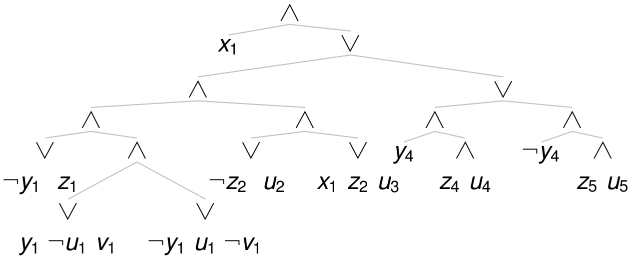


propositional skeleton φ^{psk}

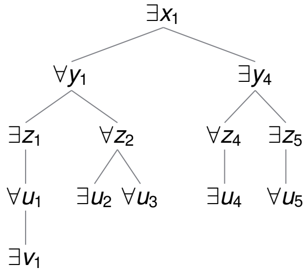


quantifier tree

Quantifier shifting: how the sausage is made



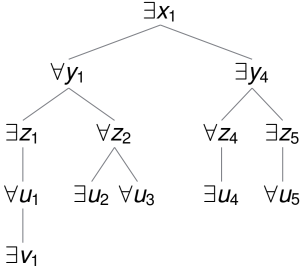
propositional skeleton φ^{psk}



quantifier tree

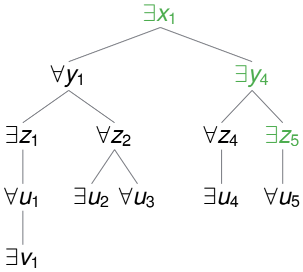
goal: $\exists X_1 . \forall X_2 . \exists X_3 . \forall X_4 . \exists X_5 . \varphi^{psk}$

Quantifier shifting: how the sausage is made



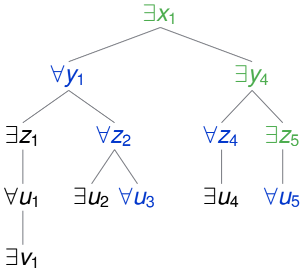
$$\exists \qquad . \forall \qquad . \exists \qquad . \forall \quad . \exists \quad . \varphi^{\mathrm{psk}}$$

Quantifier shifting: how the sausage is made



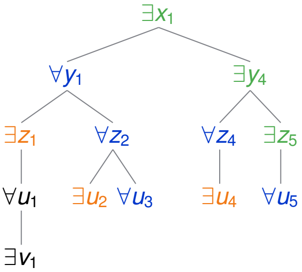
$$\exists x_1 y_4 z_5 . \forall \qquad . \exists \qquad . \forall \quad . \exists \quad . \varphi^{\mathrm{psk}}$$

Quantifier shifting: how the sausage is made



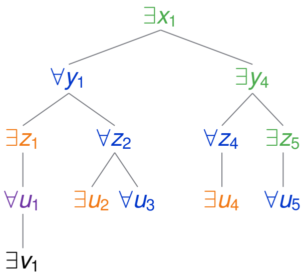
$$\exists x_1 y_4 z_5 . \forall y_1 z_2 u_3 z_4 u_5 . \exists \qquad . \forall \quad . \exists \quad . \varphi^{\mathrm{psk}}$$

Quantifier shifting: how the sausage is made



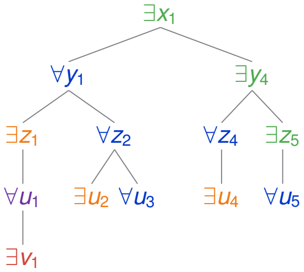
$$\exists x_1 y_4 z_5 . \forall y_1 z_2 u_3 z_4 u_5 . \exists z_1 u_2 u_4 . \forall \quad . \exists \quad . \varphi^{\mathrm{psk}}$$

Quantifier shifting: how the sausage is made



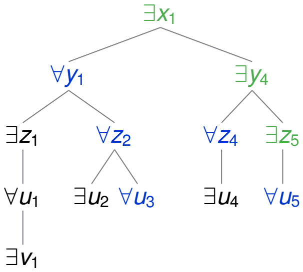
$$\exists x_1\, y_4\, z_5 .\, \forall y_1\, z_2\, u_3\, z_4\, u_5 .\, \exists z_1\, u_2\, u_4 .\, \forall u_1 .\, \exists \quad .\, \varphi^{\mathrm{psk}}$$

Quantifier shifting: how the sausage is made



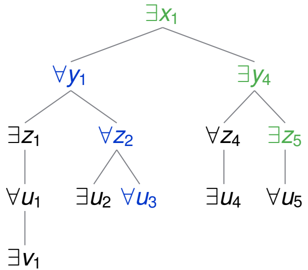
$$\exists x_1\, y_4\, z_5 .\, \forall y_1\, z_2\, u_3\, z_4\, u_5 .\, \exists z_1\, u_2\, u_4 .\, \forall u_1 .\, \exists v_1 .\, \varphi^{\mathrm{psk}}$$

Quantifier shifting: how the sausage is made



$$\exists x_1\, y_4\, z_5 \,.\, \forall y_1\, z_2\, u_3\, z_4\, u_5 \,.\, \exists z_1\, u_2\, u_4 \,.\, \forall u_1 \,.\, \exists v_1 \,.\, \varphi^{\mathrm{psk}}$$

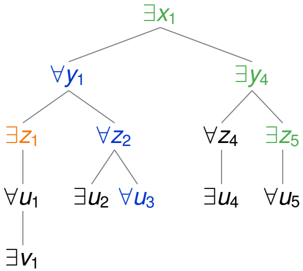
Quantifier shifting: how the sausage is made



$$\exists x_1\, y_4\, z_5 .\ \forall y_1\, z_2\, u_3\, z_4\, u_5 .\ \exists z_1\, u_2\, u_4 .\ \forall u_1 .\ \exists v_1 .\ \varphi^{\mathrm{psk}}$$

$$\exists x_1\, y_4\, z_5 .\ \forall y_1\, z_2\, u_3 .\ \exists \quad .\ \forall \qquad .\ \exists \qquad .\ \varphi^{\mathrm{psk}}$$

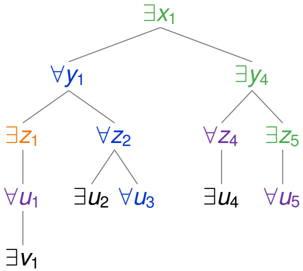
Quantifier shifting: how the sausage is made



$$\exists x_1 y_4 z_5 . \forall y_1 z_2 u_3 z_4 u_5 . \exists z_1 u_2 u_4 . \forall u_1 . \exists v_1 . \varphi^{psk}$$

$$\exists x_1 y_4 z_5 . \forall y_1 z_2 u_3 . \exists z_1 . \forall \qquad . \exists \qquad . \varphi^{psk}$$

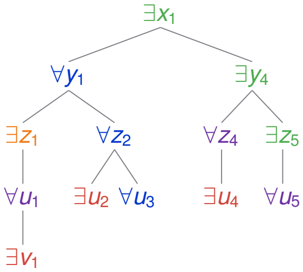
Quantifier shifting: how the sausage is made



$$\exists x_1\, y_4\, z_5 \,.\, \forall y_1\, z_2\, u_3\, z_4\, u_5 \,.\, \exists z_1\, u_2\, u_4 \,.\, \forall u_1 \,.\, \exists v_1 \,.\, \varphi^{\mathrm{psk}}$$

$$\exists x_1\, y_4\, z_5 \,.\, \forall y_1\, z_2\, u_3 \,.\, \exists z_1 \,.\, \forall u_1\, z_4\, u_5 \,.\, \exists \qquad \,.\, \varphi^{\mathrm{psk}}$$

Quantifier shifting: how the sausage is made



$$\exists x_1\, y_4\, z_5 .\, \forall y_1\, z_2\, u_3\, z_4\, u_5 .\, \exists z_1\, u_2\, u_4 .\, \forall u_1 .\, \exists v_1 .\, \varphi^{\mathrm{psk}}$$

$$\exists x_1\, y_4\, z_5 .\, \forall y_1\, z_2\, u_3 .\, \exists z_1 .\, \forall u_1\, z_4\, u_5 .\, \exists v_1\, u_2\, u_4 .\, \varphi^{\mathrm{psk}}$$

Digression: Thoralf Skolem would like a word with us

In FOL we can eliminate existential quantifiers via Skolem functions:

$$\forall x.\,(\forall y.\,\exists a.\,\varphi(x, y, a)) \wedge (\forall z.\,\exists b.\,\psi(x, z, b))$$

$$\downarrow$$

$$\forall x.\,\forall y.\,\forall z.\,\varphi(x, y, a(x, y)) \wedge \psi(x, z, b(x, z))$$

Digression: Thoralf Skolem would like a word with us

In FOL we can eliminate existential quantifiers via Skolem functions:

$$\forall x.\, (\forall y.\, \exists a.\, \varphi(x, y, a)) \wedge (\forall z.\, \exists b.\, \psi(x, z, b))$$
$$\downarrow$$
$$\forall x.\, \forall y.\, \forall z.\, \varphi(x, y, a(x, y)) \wedge \psi(x, z, b(x, z))$$

Explicit variable dependence cannot be expressed in QBF!

Digression: Thoralf Skolem would like a word with us

In FOL we can eliminate existential quantifiers via Skolem functions:

$$\forall x.\, (\forall y.\, \exists a.\, \varphi(x, y, a)) \wedge (\forall z.\, \exists b.\, \psi(x, z, b))$$
$$\downarrow$$
$$\forall x.\, \forall y.\, \forall z.\, \varphi(x, y, a(x, y)) \wedge \psi(x, z, b(x, z))$$

Explicit variable dependence cannot be expressed in QBF!

Myth: QBF $\subseteq$ FOL

Digression: Thoralf Skolem would like a word with us

In FOL we can eliminate existential quantifiers via Skolem functions:

$$\forall x.\, (\forall y.\, \exists a.\, \varphi(x, y, a)) \wedge (\forall z.\, \exists b.\, \psi(x, z, b))$$
$$\downarrow$$
$$\forall x.\, \forall y.\, \forall z.\, \varphi(x, y, a(x, y)) \wedge \psi(x, z, b(x, z))$$

Explicit variable dependence cannot be expressed in QBF!

Myth: QBF $\subseteq$ FOL

Reality: QBF $\subseteq$ SOL

Skolemization over QBF operates on HOL

Methods for quantifier shifting

- Egly, Seidl, Tompits, Woltran, Zolda. *Comparing different prenexing strategies for quantified Boolean formulas*. SAT 2003.

  Quantifiers are shifted following strategies $\exists^{\dagger}\forall^{\ddagger}$ with $\dagger, \ddagger \in \{\uparrow, \downarrow\}$.
  - $\uparrow$ means "push this quantifier rootwards"
  - $\downarrow$ means "push this quantifier matrixwards"

Methods for quantifier shifting

- Egly, Seidl, Tompits, Woltran, Zolda. *Comparing different prenexing strategies for quantified Boolean formulas*. SAT 2003.

  Quantifiers are shifted following strategies $\exists^{\dagger}\forall^{\ddagger}$ with $\dagger, \ddagger \in \{\uparrow, \downarrow\}$.
  - $\uparrow$ means "push this quantifier rootwards"
  - $\downarrow$ means "push this quantifier matrixwards"

  Problem: strategies $\exists^{\uparrow}\forall^{\downarrow}$ and $\exists^{\downarrow}\forall^{\uparrow}$ are ambiguously defined.

Methods for quantifier shifting

- Egly, Seidl, Tompits, Woltran, Zolda. *Comparing different prenexing strategies for quantified Boolean formulas*. SAT 2003.

  Quantifiers are shifted following strategies $\exists^{\dagger}\forall^{\ddagger}$ with $\dagger, \ddagger \in \{\uparrow, \downarrow\}$.
  - $\uparrow$ means "push this quantifier rootwards"
  - $\downarrow$ means "push this quantifier matrixwards"

  Problem: strategies $\exists^{\uparrow}\forall^{\downarrow}$ and $\exists^{\downarrow}\forall^{\uparrow}$ are ambiguously defined.

- $\sim$*handwaves enthusiastically*$\sim$

Methods for quantifier shifting

- Egly, Seidl, Tompits, Woltran, Zolda. *Comparing different prenexing strategies for quantified Boolean formulas*. SAT 2003.

  Quantifiers are shifted following strategies $\exists^{\dagger}\forall^{\ddagger}$ with $\dagger, \ddagger \in \{\uparrow, \downarrow\}$.
  - $\uparrow$ means "push this quantifier rootwards"
  - $\downarrow$ means "push this quantifier matrixwards"

  Problem: strategies $\exists^{\uparrow}\forall^{\downarrow}$ and $\exists^{\downarrow}\forall^{\uparrow}$ are ambiguously defined.

- $\sim$*handwaves enthusiastically*$\sim$

A systematic study of the effect of quantifier shifting on solver performance requires well-defined quantifier shifting algorithms

Digression: Hamilton would like a word with us

Digression: William Hamilton would like a word with us

Digression: William Hamilton would like a word with us

Newtonian mechanics describe the configuration of a system depending on previous configuration

- ... if the system is simple enough and you can solve it!

Digression: William Hamilton would like a word with us

Newtonian mechanics describe the configuration of a system depending on previous configuration

- ... if the system is simple enough and you can solve it!

Hamiltonian mechanics describe extra energy needed for a configuration; minima are solutions

- ... you can solve *weirder* problems!

Digression: William Hamilton would like a word with us

Newtonian mechanics describe the configuration of a system depending on previous configuration

- ... if the system is simple enough and you can solve it!

Hamiltonian mechanics describe extra energy needed for a configuration; minima are solutions

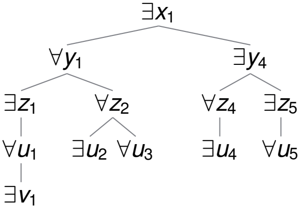
- ... you can solve *weirder* problems!

Newton rules modify a configuration until halt
Hamilton rules describe configuration preference

Quantifier trees and linearizations

We can regard quantifier trees as partially ordered sets

$x < y$ iff y is a descendent of x

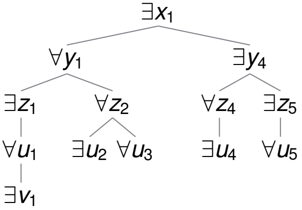


Quantifier trees and linearizations

We can regard quantifier trees as partially ordered sets

$x < y$ iff y is a descendent of x

The QBF prefix is also a tree (in this case, linear)



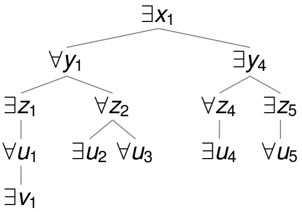
Quantifier trees and linearizations

We can regard quantifier trees as partially ordered sets

$x < y$ iff y is a descendent of x

The QBF prefix is also a tree (in this case, linear)

Linearizations: quantifier tree maps that preserve (non-strict) order and quantifiers



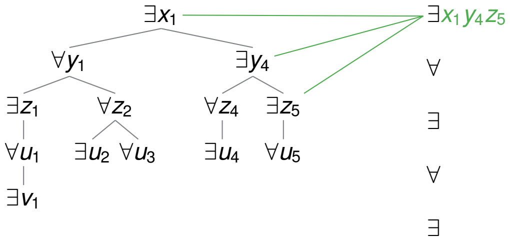
Quantifier trees and linearizations

We can regard quantifier trees as partially ordered sets

$x < y$ iff y is a descendent of x

The QBF prefix is also a tree (in this case, linear)

Linearizations: quantifier tree maps that preserve (non-strict) order and quantifiers



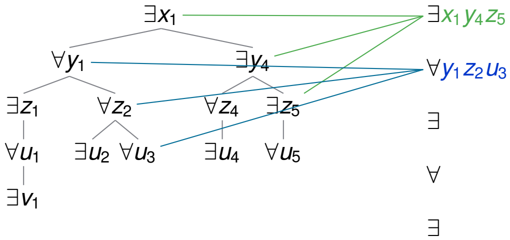
Quantifier trees and linearizations

We can regard quantifier trees as partially ordered sets

$x < y$ iff y is a descendent of x

The QBF prefix is also a tree (in this case, linear)

Linearizations: quantifier tree maps that preserve (non-strict) order and quantifiers



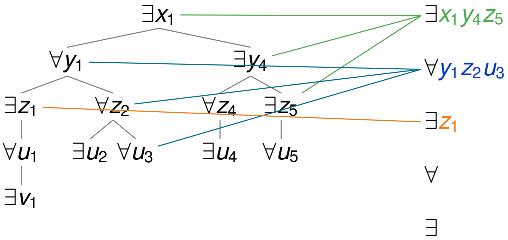
Quantifier trees and linearizations

We can regard quantifier trees as partially ordered sets

$x < y$ iff y is a descendent of x

The QBF prefix is also a tree (in this case, linear)

Linearizations: quantifier tree maps that preserve (non-strict) order and quantifiers



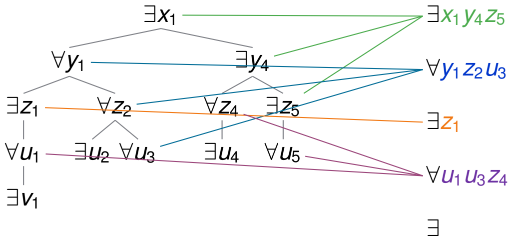
Quantifier trees and linearizations

We can regard quantifier trees as partially ordered sets

$x < y$ iff y is a descendent of x

The QBF prefix is also a tree (in this case, linear)

Linearizations: quantifier tree maps that preserve (non-strict) order and quantifiers



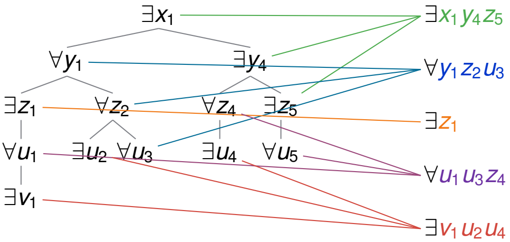
Quantifier trees and linearizations

We can regard quantifier trees as partially ordered sets

$x < y$ iff y is a descendent of x

The QBF prefix is also a tree (in this case, linear)

Linearizations: quantifier tree maps that preserve (non-strict) order and quantifiers



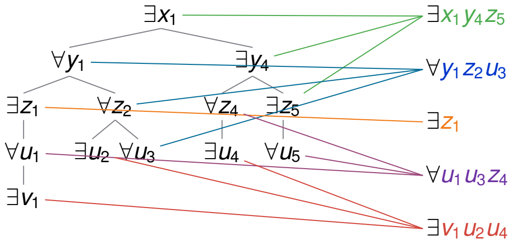
Quantifier trees and linearizations

We can regard quantifier trees as partially ordered sets

$x < y$ iff y is a descendent of x

The QBF prefix is also a tree (in this case, linear)

Linearizations: quantifier tree maps that preserve (non-strict) order and quantifiers



linearizations induce semantically equivalent quantifier shifts

Preferences over linearizations

The plan

Preferences over linearizations

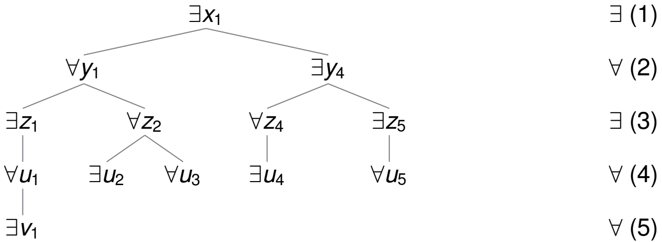
The plan

- Define a partial order over linearizations (*preferences*)

Preferences over linearizations

The plan

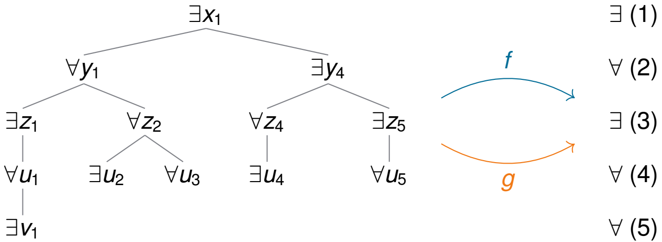
- Define a partial order over linearizations (*preferences*)



Preferences over linearizations

The plan

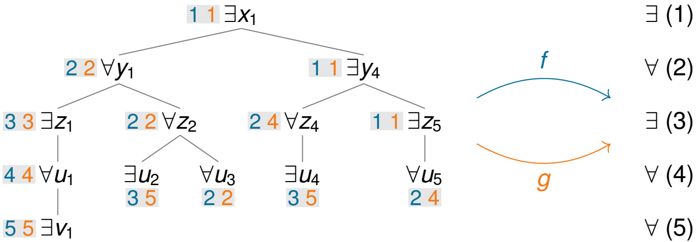
- Define a partial order over linearizations (*preferences*)



Preferences over linearizations

The plan

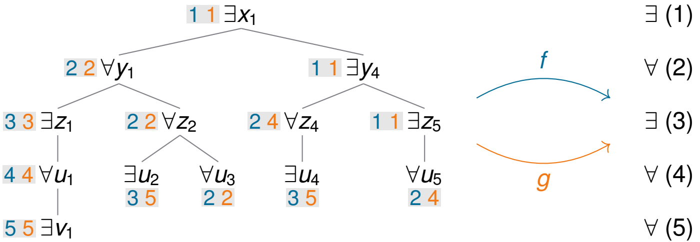
- Define a partial order over linearizations (*preferences*)



Preferences over linearizations

The plan

- Define a partial order over linearizations (*preferences*)



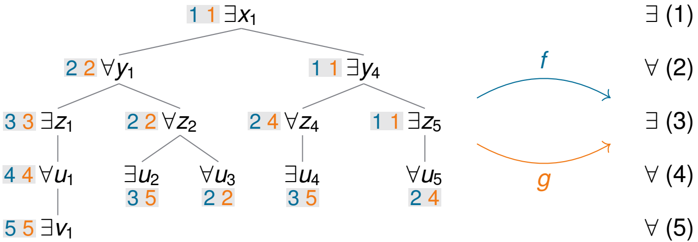
"push ∀ rootwards"

$f > g$

Preferences over linearizations

The plan

- Define a partial order over linearizations (*preferences*)



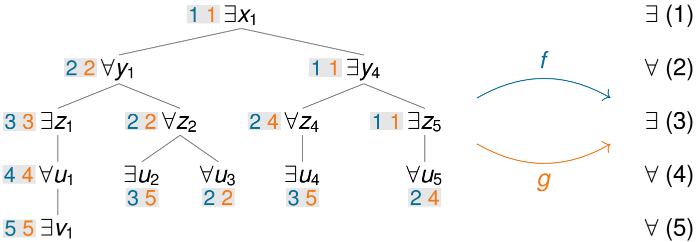
"push $\forall$ rootwards"
$f > g$

"push $\exists$ matrixwards"
$g > f$

Preferences over linearizations

The plan

- Define a partial order over linearizations (*preferences*)
- Show that the maximal linearization is unique



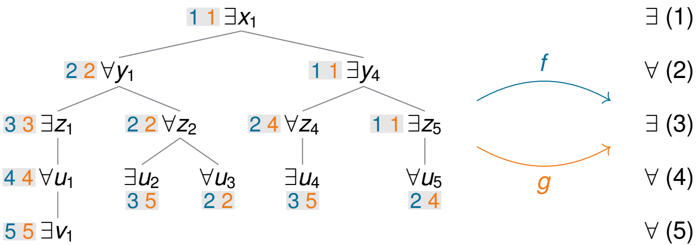
"push $\forall$ rootwards"
$f > g$

"push $\exists$ matrixwards"
$g > f$

Preferences over linearizations

The plan

- Define a partial order over linearizations (*preferences*)
- Show that the maximal linearization is unique (and pray that it is easy to compute)



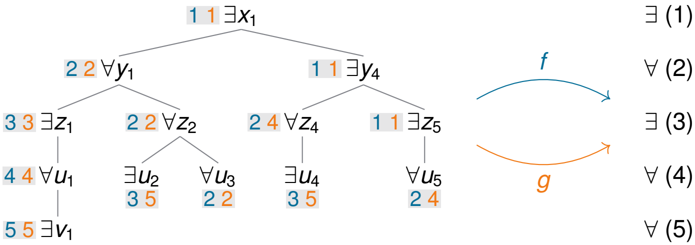
“push ∀ rootwards”

$f > g$

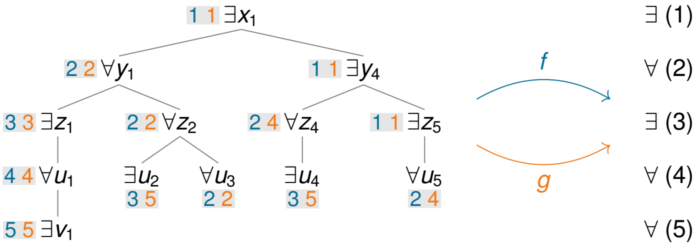
“push ∃ matrixwards”

$g > f$

Semipreferences

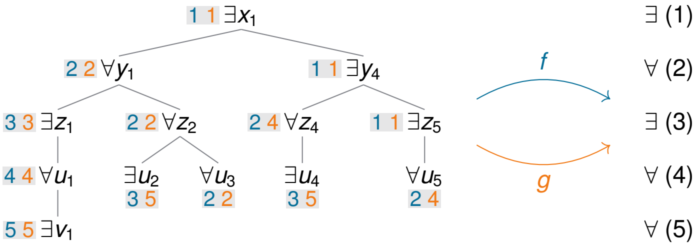


Semipreferences



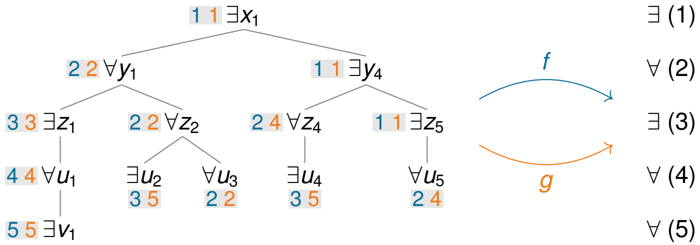
- $\forall\!\uparrow$: $f \leq^{\forall\uparrow} g$ iff $f(a) \geq g(a)$ for all $\forall$-quantified nodes a

Semipreferences



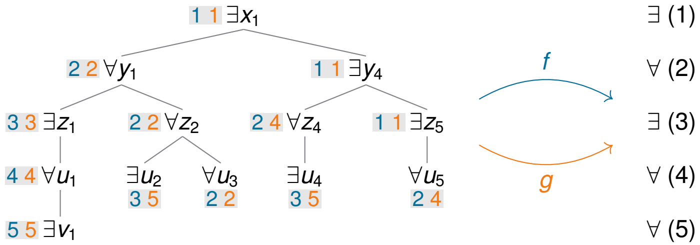
- $\forall\!\uparrow$: $f \leq^{\forall\uparrow} g$ iff $f(a) \geq g(a)$ for all $\forall$-quantified nodes a $\quad g \leq^{\forall\uparrow} f$

Semipreferences



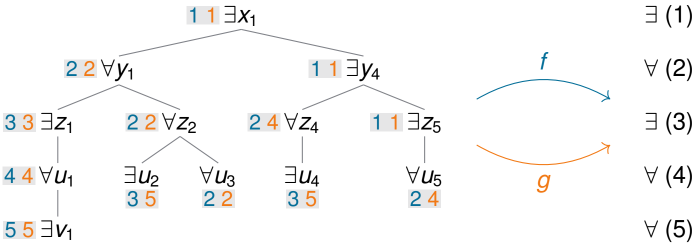
- $\forall\uparrow$: $f \leq^{\forall\uparrow} g$ iff $f(a) \geq g(a)$ for all $\forall$-quantified nodes a $\quad$ $g \leq^{\forall\uparrow} f$
- $\forall\downarrow$: $f \leq^{\forall\downarrow} g$ iff $f(a) \leq g(a)$ for all $\forall$-quantified nodes a

Semipreferences



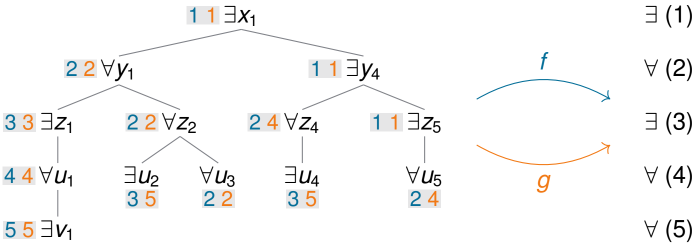
- $\forall\uparrow$: $f \leq^{\forall\uparrow} g$ iff $f(a) \geq g(a)$ for all $\forall$-quantified nodes a $\quad g \leq^{\forall\uparrow} f$
- $\forall\downarrow$: $f \leq^{\forall\downarrow} g$ iff $f(a) \leq g(a)$ for all $\forall$-quantified nodes a $\quad f \leq^{\forall\downarrow} g$

Semipreferences



- $\forall\uparrow$: $f \leq^{\forall\uparrow} g$ iff $f(a) \geq g(a)$ for all $\forall$-quantified nodes a $\quad g \leq^{\forall\uparrow} f$
- $\forall\downarrow$: $f \leq^{\forall\downarrow} g$ iff $f(a) \leq g(a)$ for all $\forall$-quantified nodes a $\quad f \leq^{\forall\downarrow} g$
- Similarly for $\exists\uparrow$ and $\exists\downarrow$ $\quad g \leq^{\exists\uparrow} f$ and $f \leq^{\exists\downarrow} g$

Semipreferences



- $\forall\!\uparrow$: $f \leq^{\forall\uparrow} g$ iff $f(a) \geq g(a)$ for all $\forall$-quantified nodes a $\quad g \leq^{\forall\uparrow} f$
- $\forall\!\downarrow$: $f \leq^{\forall\downarrow} g$ iff $f(a) \leq g(a)$ for all $\forall$-quantified nodes a $\quad f \leq^{\forall\downarrow} g$
- Similarly for $\exists\!\uparrow$ and $\exists\!\downarrow$ $\quad g \leq^{\exists\uparrow} f$ and $f \leq^{\exists\downarrow} g$

these are not partial orders, though...

Strategies, but better

We can now express the strategies from [Egly *et al*, 2003] as preference relations:

$$f \leq^{\exists\downarrow\forall\uparrow} g \text{ iff } f \leq^{\exists\downarrow} g \text{ and } f \leq^{\forall\uparrow} g$$

Strategies, but better

We can now express the strategies from [Egly *et al*, 2003] as preference relations:

$$f \leq^{\exists\downarrow\forall\uparrow} g \text{ iff } f \leq^{\exists\downarrow} g \text{ and } f \leq^{\forall\uparrow} g$$

This is a partial order ☺

Strategies, but better

We can now express the strategies from [Egly *et al*, 2003] as preference relations:

$$f \leq^{\exists\downarrow\forall\uparrow} g \text{ iff } f \leq^{\exists\downarrow} g \text{ and } f \leq^{\forall\uparrow} g$$

This is a partial order ☺ but its maximal linearization is not unique ☹

Strategies, but better

We can now express the strategies from [Egly *et al*, 2003] as preference relations:

$$f \leq^{\exists\downarrow\forall\uparrow} g \text{ iff } f \leq^{\exists\downarrow} g \text{ and } f \leq^{\forall\uparrow} g$$

This is a partial order ☺ but its maximal linearization is not unique ☹

We need to specify how one quantifier is "prioritized" over the other

Strategies, but better

We can now express the strategies from [Egly *et al*, 2003] as preference relations:

$$f \leq^{\exists\downarrow\forall\uparrow} g \text{ iff } f \leq^{\exists\downarrow} g \text{ and } f \leq^{\forall\uparrow} g$$

This is a partial order ☺ but its maximal linearization is not unique ☹

We need to specify how one quantifier is "prioritized" over the other

We propose strategies of the form $Q\dagger\ddagger$, where $Q \in \{\forall, \exists\}$ and $\dagger, \ddagger \in \{\uparrow, \downarrow\}$:

- Q-quantified nodes are pushed $\dagger$-wards
- $\overline{Q}$-quantified nodes are pushed $\ddagger$-wards
- In case of a trade-off, Q quantifiers are prioritized over $\overline{Q}$

Combining semipreferences

We propose strategies of the form $Q\dagger\ddagger$, where $Q \in \{\forall, \exists\}$ and $\dagger, \ddagger \in \{\uparrow, \downarrow\}$:

- Q-quantified nodes are pushed $\dagger$-wards
- $\overline{Q}$-quantified nodes are pushed $\ddagger$-wards
- In case of a trade-off, Q quantifiers are prioritized over $\overline{Q}$

This sounds a lot like a lexicographic ordering...

Combining semipreferences

We propose strategies of the form $\mathsf{Q}{\dagger}{\ddagger}$, where $\mathsf{Q} \in \{\forall, \exists\}$ and $\dagger, \ddagger \in \{\uparrow, \downarrow\}$:

- Q-quantified nodes are pushed $\dagger$-wards
- $\overline{\mathsf{Q}}$-quantified nodes are pushed $\ddagger$-wards
- In case of a trade-off, Q quantifiers are prioritized over $\overline{\mathsf{Q}}$

This sounds a lot like a lexicographic ordering...

$$f \leq^{\mathsf{Q}\dagger\ddagger} g \text{ iff } f \leq^{\mathsf{Q}\dagger} g \text{ and if } g \leq^{\mathsf{Q}\dagger} f \text{ then also } f \leq^{\overline{\mathsf{Q}}\ddagger} g$$

Combining semipreferences

We propose strategies of the form Q$\dagger\ddagger$, where Q $\in \{\forall, \exists\}$ and $\dagger, \ddagger \in \{\uparrow, \downarrow\}$:

- Q-quantified nodes are pushed $\dagger$-wards
- $\overline{\mathrm{Q}}$-quantified nodes are pushed $\ddagger$-wards
- In case of a trade-off, Q quantifiers are prioritized over $\overline{\mathrm{Q}}$

This sounds a lot like a lexicographic ordering...

$$f \leq^{\mathrm{Q}\dagger\ddagger} g \text{ iff } f \leq^{\mathrm{Q}\dagger} g \text{ and if } g \leq^{\mathrm{Q}\dagger} f \text{ then also } f \leq^{\overline{\mathrm{Q}}\ddagger} g$$

This is a partial order ☺

Combining semipreferences

We propose strategies of the form $Q{\dagger}{\ddagger}$, where $Q \in \{\forall, \exists\}$ and $\dagger, \ddagger \in \{\uparrow, \downarrow\}$:

- Q-quantified nodes are pushed $\dagger$-wards
- $\overline{Q}$-quantified nodes are pushed $\ddagger$-wards
- In case of a trade-off, Q quantifiers are prioritized over $\overline{Q}$

This sounds a lot like a lexicographic ordering...

$$f \leq^{Q\dagger\ddagger} g \text{ iff } f \leq^{Q\dagger} g \text{ and if } g \leq^{Q\dagger} f \text{ then also } f \leq^{\overline{Q}\ddagger} g$$

This is a partial order ☺ and its maximal linearization is unique ☺☺☺

Are we there yet?

$$f \leq^{Q\dagger\ddagger} g \text{ iff } f \leq^{Q\dagger} g \text{ and if } g \leq^{Q\dagger} f \text{ then also } f \leq^{\overline{Q}\ddagger} g$$

Proof of maximal linearization uniqueness:

Are we there yet?

$$f \leq^{Q\dagger\ddagger} g \text{ iff } f \leq^{Q\dagger} g \text{ and if } g \leq^{Q\dagger} f \text{ then also } f \leq^{\overline{Q}\ddagger} g$$

Proof of maximal linearization uniqueness:

- All $\leq^{Q\dagger}$-maximal linearizations are identical over Q-quantified nodes.

Are we there yet?

$$f \leq^{Q\dagger\ddagger} g \text{ iff } f \leq^{Q\dagger} g \text{ and if } g \leq^{Q\dagger} f \text{ then also } f \leq^{\overline{Q}\ddagger} g$$

Proof of maximal linearization uniqueness:

- All $\leq^{Q\dagger}$-maximal linearizations are identical over Q-quantified nodes.
- For fixed linearization values over Q-quantified nodes, $\leq^{\overline{Q}\ddagger}$-maximal linearizations are identical over $\overline{Q}$-quantified nodes.

Are we there yet?

$$f \leq^{Q\dagger\ddagger} g \text{ iff } f \leq^{Q\dagger} g \text{ and if } g \leq^{Q\dagger} f \text{ then also } f \leq^{\overline{Q}\ddagger} g$$

Proof of maximal linearization uniqueness:

- All $\leq^{Q\dagger}$-maximal linearizations are identical over Q-quantified nodes.
- For fixed linearization values over Q-quantified nodes, $\leq^{\overline{Q}\ddagger}$-maximal linearizations are identical over $\overline{Q}$-quantified nodes.
- This only works if the target prefix has as many quantifier alternations as the maximum quantifier alternations in the source quantifier tree (but that's ok)

Are we there yet?

$$f \leq^{\mathrm{Q}\dagger\ddagger} g \text{ iff } f \leq^{\mathrm{Q}\dagger} g \text{ and if } g \leq^{\mathrm{Q}\dagger} f \text{ then also } f \leq^{\overline{\mathrm{Q}}\ddagger} g$$

Proof of maximal linearization uniqueness:

- All $\leq^{\mathrm{Q}\dagger}$-maximal linearizations are identical over Q-quantified nodes.
- For fixed linearization values over Q-quantified nodes, $\leq^{\overline{\mathrm{Q}}\ddagger}$-maximal linearizations are identical over $\overline{\mathrm{Q}}$-quantified nodes.
- This only works if the target prefix has as many quantifier alternations as the maximum quantifier alternations in the source quantifier tree (but that's ok)

So now I only need to develop a search algorithm to find the maximum!

Are we there yet?

$$f \leq^{Q\dagger\ddagger} g \text{ iff } f \leq^{Q\dagger} g \text{ and if } g \leq^{Q\dagger} f \text{ then also } f \leq^{\overline{Q}\ddagger} g$$

Proof of maximal linearization uniqueness:

- All $\leq^{Q\dagger}$-maximal linearizations are identical over Q-quantified nodes.
- For fixed linearization values over Q-quantified nodes, $\leq^{\overline{Q}\ddagger}$-maximal linearizations are identical over $\overline{Q}$-quantified nodes.
- This only works if the target prefix has as many quantifier alternations as the maximum quantifier alternations in the source quantifier tree (but that's ok)

So now I only need to develop a search algorithm to find the maximum!

- No.

Are we there yet?

$f \leq^{Q\dagger\ddagger} g$ iff $f \leq^{Q\dagger} g$ and if $g \leq^{Q\dagger} f$ then also $f \leq^{\overline{Q}\ddagger} g$

Proof of maximal linearization uniqueness:

- All $\leq^{Q\dagger}$-maximal linearizations are identical over Q-quantified nodes.
- For fixed linearization values over Q-quantified nodes, $\leq^{\overline{Q}\ddagger}$-maximal linearizations are identical over $\overline{Q}$-quantified nodes.
- This only works if the target prefix has as many quantifier alternations as the maximum quantifier alternations in the source quantifier tree (but that's ok)

So now I only need to develop a search algorithm to find the maximum!

- No. We give a single-pass recursive algorithm that computes the maximum with negligible overhead

The algorithm

Theoretical results. Let us consider a quantree $(T, \leq, q)$ and a strategy $\mathsf{Q}\dagger\ddagger$. We define the mapping $\Gamma_\dagger : T \to \{1, \ldots, \mathrm{aht}(T)\}$ given by

$$\Gamma_\dagger(x) = \big\lfloor |\max{}^\dagger\{1, \ldots, \mathrm{aht}(T)\} - \mathrm{aht}(T_x^\dagger)| + 1 \big\rfloor^\dagger_{q^\star(x)} . \tag{1}$$

Furthermore, we define the mapping $[f]^{\mathsf{Q}\ddagger} : T \to \{1, \ldots, \mathrm{aht}(T)\}$ for $f \in \mathrm{Lin}(T)$ given by

$$[f]^{\mathsf{Q}\ddagger}(x) = \big\lfloor \min{}^\ddagger\{f(y) \mid y \in T_x^\ddagger \text{ and } q(y) = \mathsf{Q}\} \big\rfloor^\ddagger_{q^\star(x)} . \tag{2}$$

Example 7 suggests that $[f]^{\mathsf{Q}\ddagger}$ can be computed recursively. Indeed, the rank of a node can be computed based on the ranks of its children or parent.

Corollary 1. *Let $x \in T$ such that $q(x) \neq \mathsf{Q}$. Then,*

$$[f]^{\mathsf{Q}\ddagger}(x) = \Big\lfloor \min{}^\ddagger\{[f]^{\mathsf{Q}\ddagger}(y) \mid x \textit{ is covered by } y \in T \textit{ w.r.t. } \leq^\ddagger\}. \Big\rfloor^\ddagger_{q^\star(x)}$$

Conclusion

QBF performance benchmarking does not currently take into account all that happens before the formula is prenexed.

Conclusion

QBF performance benchmarking does not currently take into account all that happens before the formula is prenexed.

A systematic study of the interaction between quantifier shifting and QBF performance requires well-defined quantifier shifting methods.

Conclusion

QBF performance benchmarking does not currently take into account all that happens before the formula is prenexed.

A systematic study of the interaction between quantifier shifting and QBF performance requires well-defined quantifier shifting methods.

Sometimes declarative definitions are better than procedural definitions.

Conclusion

QBF performance benchmarking does not currently take into account all that happens before the formula is prenexed.

A systematic study of the interaction between quantifier shifting and QBF performance requires well-defined quantifier shifting methods.

Sometimes declarative definitions are better than procedural definitions.

We define quantifier shifting strategies that can now be experimentally evaluated.

Conclusion

QBF performance benchmarking does not currently take into account all that happens before the formula is prenexed.

A systematic study of the interaction between quantifier shifting and QBF performance requires well-defined quantifier shifting methods.

Sometimes declarative definitions are better than procedural definitions.

We define quantifier shifting strategies that can now be experimentally evaluated.

Wait, and the results? No spoilers here. Go to **Simone Heisinger**'s talk on Thursday at 14:30

Conclusion

QBF performance benchmarking does not currently take into account all that happens before the formula is prenexed.

A systematic study of the interaction between quantifier shifting and QBF performance requires well-defined quantifier shifting methods.

Sometimes declarative definitions are better than procedural definitions.

We define quantifier shifting strategies that can now be experimentally evaluated.

Wait, and the results? No spoilers here. Go to **Simone Heisinger**'s talk on Thursday at 14:30 *... or, you know, just ask us* ☺

Conclusion

QBF performance benchmarking does not currently take into account all that happens before the formula is prenexed.

A systematic study of the interaction between quantifier shifting and QBF performance requires well-defined quantifier shifting methods.

Sometimes declarative definitions are better than procedural definitions.

We define quantifier shifting strategies that can now be experimentally evaluated.

Wait, and the results? No spoilers here. Go to **Simone Heisinger**'s talk on Thursday at 14:30 *... or, you know, just ask us* ☺

Is there a tool? Yes. Go to **Maximilian Heisinger**'s talk on Friday at 17:10.

JKU

JOHANNES KEPLER
UNIVERSITY LINZ