Order-sorted intensional logic

Expressing subtyping polymorphism with typing assertions and quantification over concepts

Đorđe Markovic, Marc Denecker

KU Leuven

ICLP 2024 Dallas, TX, USA

Logic and subtyping polymorphism

- Consider the natural language sentence:

  An animal produces sound iff it produces a sound characteristic for its species.

- What we mean:

  An animal *produces sound* iff

  that animal is a dog and it barks,

  or that animal is a cat and it meows, ...

Logic and subtyping polymorphism

- Consider the natural language sentence:

  An animal produces sound iff it produces a sound characteristic for its species.

- What we mean:

  An animal *produces sound* iff

  that animal is a dog and it barks,

  or that animal is a cat and it meows, ...

Logic and subtyping polymorphism

- Consider the natural language sentence:

  An animal *produces sound* iff it produces a sound characteristic for its species.

- What we mean:

  An animal *produces sound* iff

  that animal is a dog and it barks,

  or that animal is a cat and it meows, ...

Logic and subtyping polymorphism

- Consider the natural language sentence:

  An animal *produces sound* iff it produces a sound characteristic for its species.

- What we mean:

  An animal *produces sound* iff

  that animal is a dog and it barks,

  or that animal is a cat and it meows, ...

Logic and subtyping polymorphism

- Consider the natural language sentence:

  An animal *produces sound* iff it produces a sound characteristic for its species.

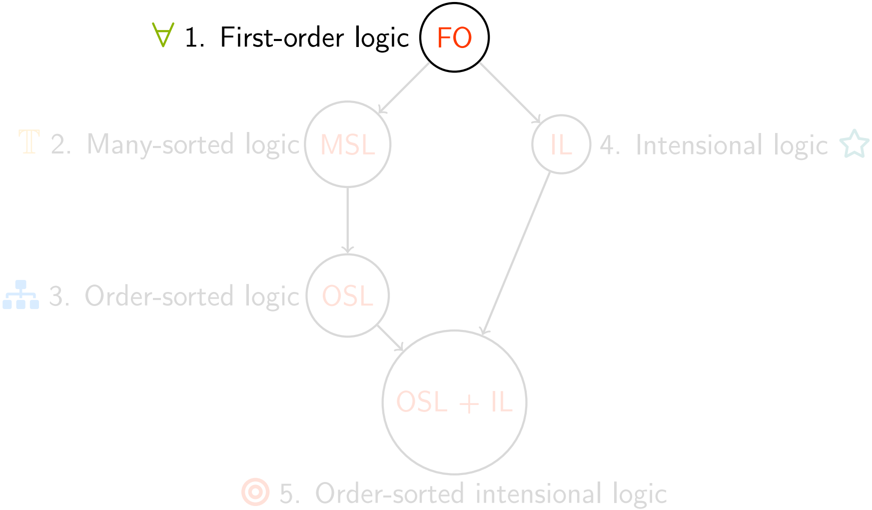
- What we mean:

  An animal *produces sound* iff

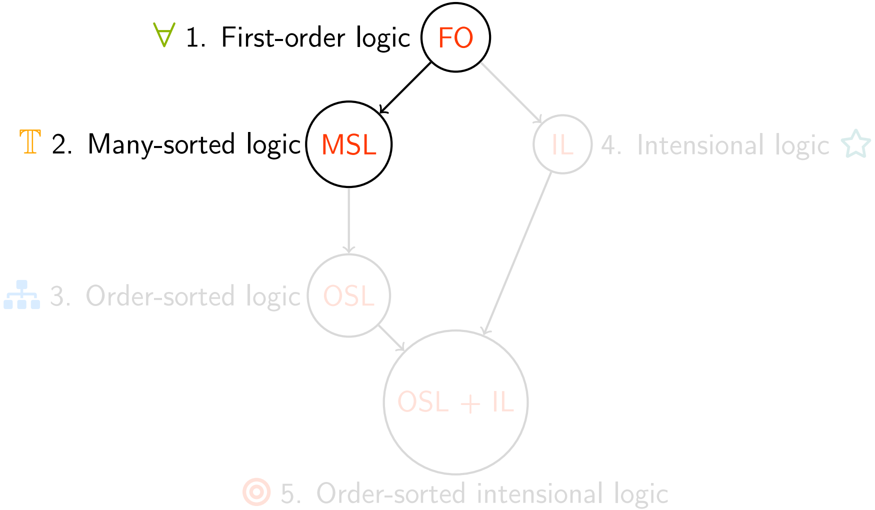
  that animal is a dog and it barks,

  or that animal is a cat and it meows, ...

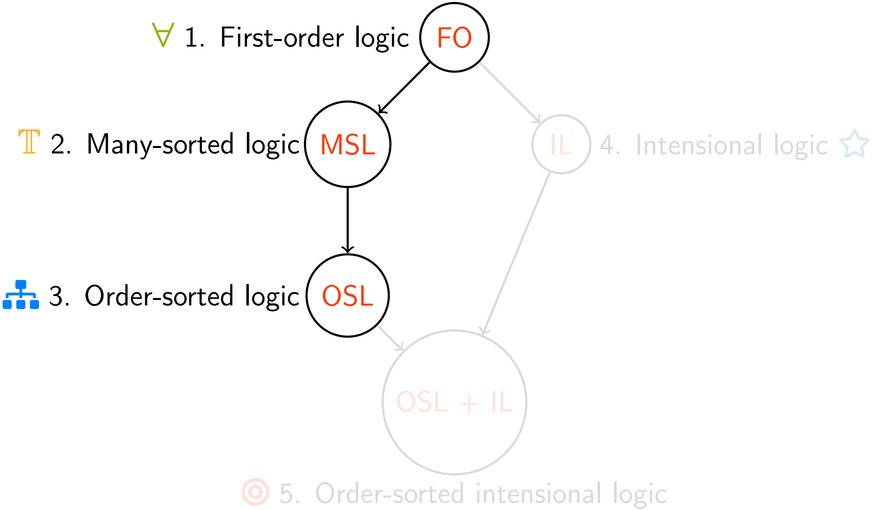
Roadmap


1. First-order logic
FO
2. Many-sorted logic
MSL
IL
4. Intensional logic
3. Order-sorted logic
OSL
OSL + IL
5. Order-sorted intensional logic

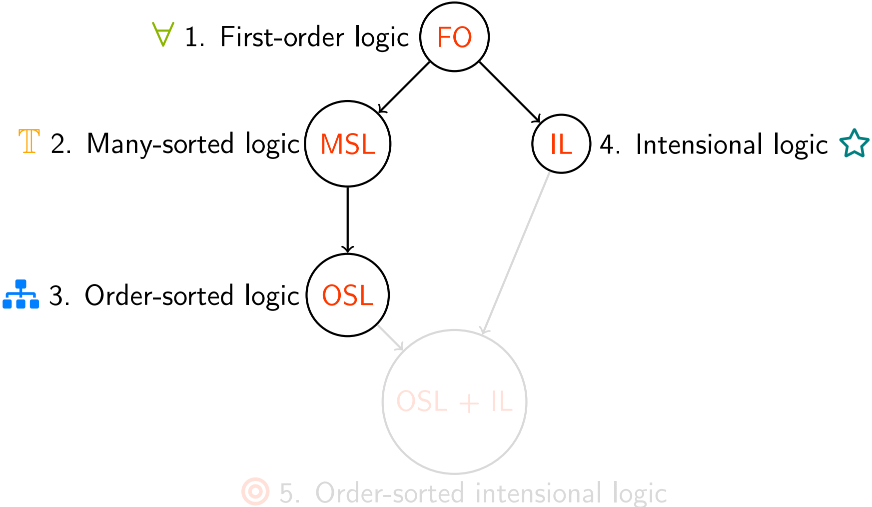
Roadmap


1. First-order logic
FO
2. Many-sorted logic
MSL
IL
4. Intensional logic
3. Order-sorted logic
OSL
OSL + IL
5. Order-sorted intensional logic

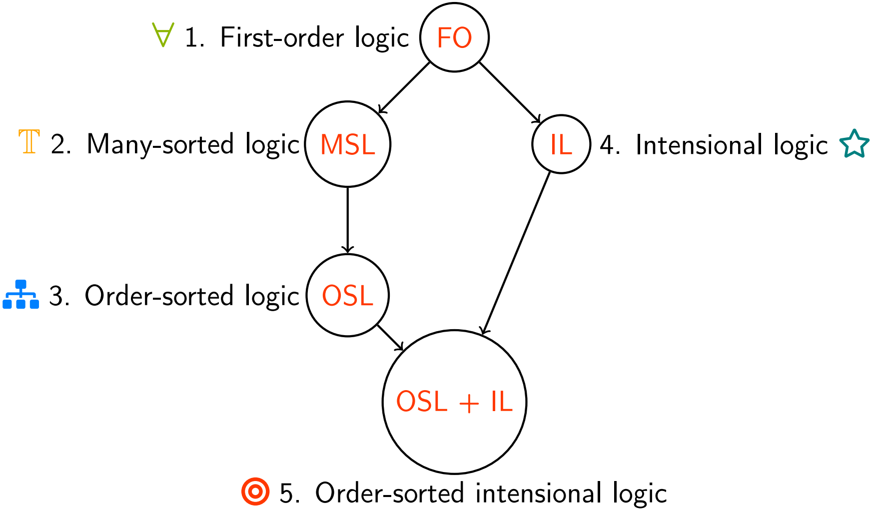
Roadmap


1. First-order logic
FO
2. Many-sorted logic
MSL
IL
4. Intensional logic
3. Order-sorted logic
OSL
OSL + IL
5. Order-sorted intensional logic

Roadmap


1. First-order logic
FO
2. Many-sorted logic
MSL
4. Intensional logic
IL
3. Order-sorted logic
OSL
OSL + IL
5. Order-sorted intensional logic

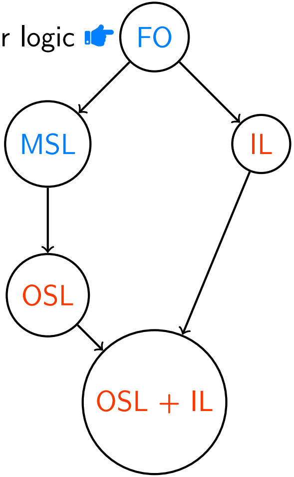
Roadmap


1. First-order logic
FO
2. Many-sorted logic
MSL
IL
4. Intensional logic
3. Order-sorted logic
OSL
OSL + IL
5. Order-sorted intensional logic

Roadmap

$\forall$ 1. First-order logic

$\mathbb{T}$ 2. Many-sorted logic



$\forall x \in \mathbb{T}$ First-order and many-sorted logic

- Example:

"There is a *Cat* eating."

- **First-order logic** is not typed! Each object is part of the "Universe".

$$\exists c : Cat(c) \land Eats(c).$$

- In **many-sorted logic**, we can introduce sort/type *Cat*.
- Then we can declare eats predicate as: $Eats : (\underline{Cat}) \to \mathbb{B}$
- Finally, we express the example statement as:

$$\exists c[Cat] : Eats(\underline{c}).$$

$\forall x \in \mathbb{T}$ First-order and many-sorted logic

- Example:

"There is a *Cat* eating."

- **First-order logic** is not typed! Each object is part of the "Universe".

$$\exists c : Cat(c) \wedge Eats(c).$$

- In **many-sorted logic**, we can introduce sort/type *Cat*.
- Then we can declare eats predicate as: $Eats : (\underline{\mathrm{Cat}}) \to \mathbb{B}$
- Finally, we express the example statement as:

$$\exists c[Cat] : Eats(\underline{c}).$$

$\forall x \in \mathbb{T}$ First-order and many-sorted logic

- Example:

"There is a *Cat* eating."

- **First-order logic** is not typed! Each object is part of the "Universe".

$$\exists c : Cat(c) \land Eats(c).$$

- In **many-sorted logic**, we can introduce sort/type *Cat*.
- Then we can declare eats predicate as: $Eats : (\underline{Cat}) \to \mathbb{B}$
- Finally, we express the example statement as:

$$\exists c[Cat] : Eats(\underline{c}).$$

$\forall x \in \mathbb{T}$ First-order and many-sorted logic

- Example:

  "There is a *Cat* eating."

- **First-order logic** is not typed! Each object is part of the "Universe".

$$\exists c : Cat(c) \land Eats(c).$$

- In **many-sorted logic**, we can introduce sort/type *Cat*.
- Then we can declare eats predicate as: $Eats : (\underline{\text{Cat}}) \to \mathbb{B}$
- Finally, we express the example statement as:

$$\exists c[Cat] : Eats(\underline{c}).$$

$\forall x \in \mathbb{T}$ First-order and many-sorted logic

- Example:

"There is a *Cat* eating."

- **First-order logic** is not typed! Each object is part of the "Universe".

$$\exists c : Cat(c) \land Eats(c).$$

- In **many-sorted logic**, we can introduce sort/type *Cat*.
- Then we can declare eats predicate as: $Eats : (\underline{\mathrm{Cat}}) \to \mathbb{B}$
- Finally, we express the example statement as:

$$\exists c[Cat] : Eats(\underline{\mathrm{c}}).$$

$\forall x \in \mathbb{T}$ First-order and many-sorted logic

- Any problems with $\exists c[Cat] : Eats(\underline{c})$?

- How to express that there is a *Dog* eating?

$$\exists d[Dog] : Eats(\underline{d})$$

$\forall x \in \mathbb{T}$ First-order and many-sorted logic

- Any problems with $\exists c[Cat] : Eats(\underline{c})$?
- How to express that there is a *Dog* eating?

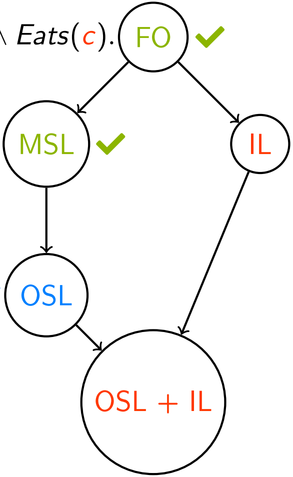
$$\exists d[Dog] : Eats(\underline{d})$$

Roadmap

$\exists c : Cat(c) \wedge Eats(c).$

$\exists c[Cat] : Eats(c).$

Order-sorted logic



Order-sorted logic

- Hierarchy of types ($<:$ subtype relation):

$$\mathit{Dog} <: \mathit{Animal} \qquad \mathit{Cat} <: \mathit{Animal}$$

- Predicative form:

$$\mathit{Dog} : (\underline{\mathit{Animal}}) \to \mathbb{B}$$

Order-sorted logic

- Hierarchy of types ($<:$ subtype relation):

$$Dog <: Animal \qquad\qquad Cat <: Animal$$

- Predicative form:

$$Dog : (\underline{\text{Animal}}) \to \mathbb{B}$$

Order-sorted logic

- Back to the eating example:

$$\mathit{Dog} <: \mathit{Animal} \qquad \mathit{Cat} <: \mathit{Animal}$$

$$\mathit{Eats} : (\underline{\text{Animal}}) \to \mathbb{B}$$

- "There is an *Animal* eating" is expressed as:

$$\exists a[\mathit{Animal}] : \mathit{Eats}(\underline{\text{a}}).$$

- Both of the following statements are well-typed:

$$\exists c[\mathit{Cat}] : \mathit{Eats}(\underline{\text{c}}). \qquad \exists d[\mathit{Dog}] : \mathit{Eats}(\underline{\text{d}}).$$

Order-sorted logic

- Back to the eating example:

$$Dog <: Animal \qquad\qquad Cat <: Animal$$

$$Eats : (\underline{Animal}) \rightarrow \mathbb{B}$$

- "There is an *Animal* eating" is expressed as:

$$\exists a[Animal] : Eats(\underline{a}).$$

- Both of the following statements are well-typed:

$$\exists c[Cat] : Eats(\underline{c}). \qquad \exists d[Dog] : Eats(\underline{d}).$$

Order-sorted logic

- Back to the eating example:

$$Dog <: Animal \qquad\qquad Cat <: Animal$$

$$Eats : (\underline{\text{Animal}}) \rightarrow \mathbb{B}$$

- "There is an *Animal* eating" is expressed as:

$$\exists a[Animal] : Eats(\underline{\text{a}}).$$

- Both of the following statements are well-typed:

$$\exists c[Cat] : Eats(\underline{\text{c}}). \qquad \exists d[Dog] : Eats(\underline{\text{d}}).$$

Order-sorted logic

- Any problems?
- Given predicates:

$$Bark : (\underline{Dog}) \to \mathbb{B} \qquad Meow : (\underline{Cat}) \to \mathbb{B}$$

- Can we define the predicate:

$$ProducingSound(\underline{Animal}) \to \mathbb{B}$$

Order-sorted logic

- Any problems?
- Given predicates:

$$Bark : (\underline{Dog}) \to \mathbb{B} \qquad Meow : (\underline{Cat}) \to \mathbb{B}$$

- Can we define the predicate:

$$ProducingSound(\underline{Animal}) \to \mathbb{B}$$

Order-sorted logic

- Any problems?
- Given predicates:

$$Bark : (\underline{Dog}) \rightarrow \mathbb{B} \qquad\qquad Meow : (\underline{Cat}) \rightarrow \mathbb{B}$$

- Can we define the predicate:

$$ProducingSound(\underline{Animal}) \rightarrow \mathbb{B}$$

Order-sorted logic

- Yes we can!

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (\exists d[Dog] : \underline{a} =^{*1} \underline{d} \wedge Bark(\underline{d})) \vee (\exists c[Cat] : \underline{a} =^{*2} \underline{c} \wedge Meow(\underline{c})).$$

- Can we do better?

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \wedge Bark(\underline{a})) \vee (Cat(\underline{a}) \wedge Meow(\underline{a})).$$

Order-sorted logic

- Yes we can!

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow$$
$$(\exists d[Dog] : \underline{a} =^{*1} \underline{d} \wedge Bark(\underline{d})) \vee (\exists c[Cat] : \underline{a} =^{*2} \underline{c} \wedge Meow(\underline{c})).$$

- Can we do better?

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow$$
$$(Dog(\underline{a}) \wedge Bark(\underline{a})) \vee (Cat(\underline{a}) \wedge Meow(\underline{a})).$$

Guarded order-sorted logic

- Let: $S <: T$ $\quad S : (\underline{T}) \to \mathbb{B}$ $\quad a : T$
- Then: $S(a) \Rightarrow \phi[a]$ and $S(a) \land \phi[a]$ are well typed formulas!
- Abbreviated: $\langle\langle \phi(a) \rangle\rangle$ and $[[\phi(a)]]$
- Example:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \land Bark(\underline{a})) \lor (Cat(\underline{a}) \land Meow(\underline{a})).$$

- The compact version:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow: [[Bark(\underline{a})]] \lor [[Meow(\underline{a})]].$$

Guarded order-sorted logic

- Let: $S <: T$ $\quad S : (\underline{T}) \to \mathbb{B}$ $\quad a : T$
- Then: $S(a) \Rightarrow \phi[a]$ and $S(a) \land \phi[a]$ are well typed formulas!
- Abbreviated: $\langle\langle \phi(a) \rangle\rangle$ and $[[\phi(a)]]$
- Example:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \land Bark(\underline{a})) \lor (Cat(\underline{a}) \land Meow(\underline{a})).$$

- The compact version:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow: [[Bark(\underline{a})]] \lor [[Meow(\underline{a})]].$$

Guarded order-sorted logic

- Let: $S <: T$ $\quad S : (\underline{T}) \to \mathbb{B}$ $\quad a : T$
- Then: $S(a) \Rightarrow \phi[a]$ and $S(a) \wedge \phi[a]$ are well typed formulas!
- Abbreviated: $\langle\langle \phi(a) \rangle\rangle$ and $[[\phi(a)]]$
- Example:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \wedge Bark(\underline{a})) \vee (Cat(\underline{a}) \wedge Meow(\underline{a})).$$

- The compact version:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow: [[Bark(\underline{a})]] \vee [[Meow(\underline{a})]].$$

Guarded order-sorted logic

- Let: $S <: T$ $\quad S : (\underline{T}) \to \mathbb{B}$ $\quad a : T$
- Then: $S(a) \Rightarrow \phi[a]$ and $S(a) \wedge \phi[a]$ are well typed formulas!
- Abbreviated: $\langle\langle \phi(a) \rangle\rangle$ and $[[\phi(a)]]$

- Example:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \wedge Bark(\underline{a})) \vee (Cat(\underline{a}) \wedge Meow(\underline{a})).$$

- The compact version:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow: [[Bark(\underline{a})]] \vee [[Meow(\underline{a})]].$$

Guarded order-sorted logic

- Let: $S <: T$ $\quad S : (\underline{T}) \to \mathbb{B}$ $\quad a : T$
- Then: $S(a) \Rightarrow \phi[a]$ and $S(a) \land \phi[a]$ are well typed formulas!
- Abbreviated: $\langle\langle \phi(a) \rangle\rangle$ and $[[\phi(a)]]$

- Example:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \land Bark(\underline{a})) \lor (Cat(\underline{a}) \land Meow(\underline{a})).$$

- The compact version:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow: [[Bark(\underline{a})]] \lor [[Meow(\underline{a})]].$$

Guarded order-sorted logic

- $\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow: [[Bark(\underline{a})]] \lor [[Meow(\underline{a})]]$.

  An animal *produces sound* iff

  that animal is a dog and it barks, or that animal is a cat and it meows, ...

- Recall that we want:

  An animal *produces sound* iff it produces a sound characteristic for its species.

- Here: sound characteristic for species is collection of **concepts** as Barks, Meows, ...

Guarded order-sorted logic

- $\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow: [[Bark(\underline{a})]] \lor [[Meow(\underline{a})]]$.

  An animal *produces sound* iff

  that animal is a dog and it barks, or that animal is a cat and it meows, ...

- Recall that we want:

  An animal *produces sound* iff it produces a sound characteristic for its species.

- Here: sound characteristic for species is collection of **concepts** as Barks, Meows, ...

Guarded order-sorted logic

- $\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow: [[Bark(\underline{a})]] \lor [[Meow(\underline{a})]].$

  An animal <u>*produces sound*</u> iff

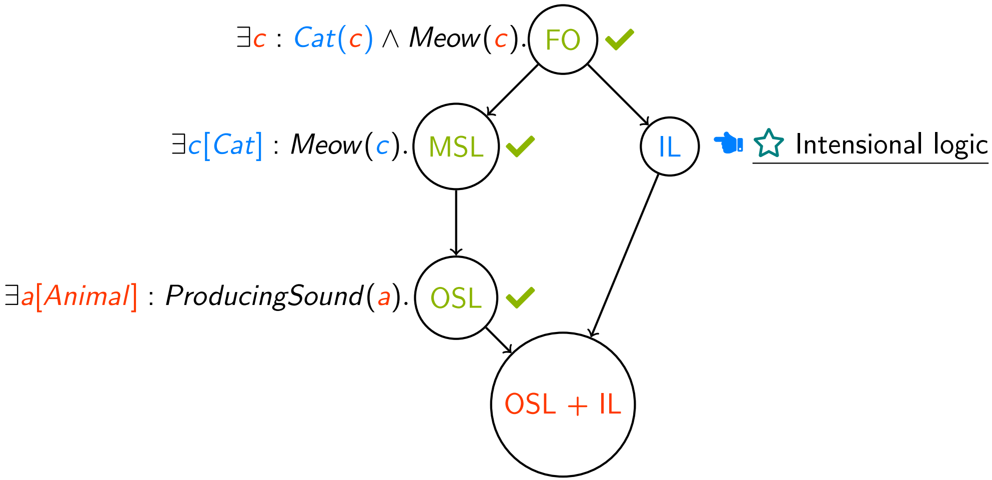
  that animal is a dog and it barks, or that animal is a cat and it meows, ...

- Recall that we want:

  An animal *produces sound* iff it produces a sound characteristic for its species.

- Here: sound characteristic for species is collection of **concepts** as Barks, Meows, ...

Roadmap


$\exists c : Cat(c) \wedge Meow(c).$
FO
$\exists c[Cat] : Meow(c).$
MSL
IL
Intensional logic
$\exists a[Animal] : ProducingSound(a).$
OSL
OSL + IL

Intensional logic

- Extension vs Intension
- Example: Evening star and morning star are concepts.
- New type: $Concept$
- Elements of $Concept := \{\widetilde{Animal}, \widetilde{Dog}, \widetilde{Cat}, \widetilde{Eats}, \widetilde{Bark}, \widetilde{Meow}, \ldots\}$
- New operators: $`(Animal) = \widetilde{Animal}$ and $\$(\widetilde{Eats})(d) = Eats(d)$
- Example:

$$Sound(Concept) \to \mathbb{B}$$

$$Sound(`(Bark)) \land Sound(`(Meow)).$$

Intensional logic

- Extension vs Intension
- Example: Evening star and morning star are concepts.
- New type: *Concept*
- Elements of *Concept* $:= \{\widetilde{Animal}, \widetilde{Dog}, \widetilde{Cat}, \widetilde{Eats}, \widetilde{Bark}, \widetilde{Meow}, \dots\}$
- New operators: $`(Animal) = \widetilde{Animal}$ and $\$(\widetilde{Eats})(d) = Eats(d)$
- Example:

$$Sound(Concept) \to \mathbb{B}$$

$$Sound(`(Bark)) \wedge Sound(`(Meow)).$$

Intensional logic

- Extension vs Intension
- Example: Evening star and morning star are concepts.

- New type: *Concept*

- Elements of $Concept := \{\widetilde{Animal}, \widetilde{Dog}, \widetilde{Cat}, \widetilde{Eats}, \widetilde{Bark}, \widetilde{Meow}, \ldots\}$
- New operators: $`(Animal) = \widetilde{Animal}$ and $\$(\widetilde{Eats})(d) = Eats(d)$
- Example:

$$Sound(Concept) \to \mathbb{B}$$

$$Sound(`(Bark)) \land Sound(`(Meow)).$$

Intensional logic

- Extension vs Intension
- Example: Evening star and morning star are concepts.
- New type: *Concept*
- Elements of *Concept* $:= \{\widetilde{Animal}, \widetilde{Dog}, \widetilde{Cat}, \widetilde{Eats}, \widetilde{Bark}, \widetilde{Meow}, \ldots\}$
- New operators: $`(Animal) = \widetilde{Animal}$ and $\$(\widetilde{Eats})(d) = Eats(d)$
- Example:

$$Sound(Concept) \to \mathbb{B}$$

$$Sound(`(Bark)) \land Sound(`(Meow)).$$

☆ Intensional logic

- Extension vs Intension
- Example: Evening star and morning star are concepts.
- New type: *Concept*
- Elements of *Concept* $:= \{\widetilde{Animal}, \widetilde{Dog}, \widetilde{Cat}, \widetilde{Eats}, \widetilde{Bark}, \widetilde{Meow}, \ldots\}$

- New operators: $`(Animal) = \widetilde{Animal}$ and $\$(\widetilde{Eats})(d) = Eats(d)$

- Example:

$$Sound(Concept) \to \mathbb{B}$$

$$Sound(`(Bark)) \wedge Sound(`(Meow)).$$

Intensional logic

- Extension vs Intension
- Example: Evening star and morning star are concepts.
- New type: *Concept*
- Elements of *Concept* $:= \{\widetilde{Animal}, \widetilde{Dog}, \widetilde{Cat}, \widetilde{Eats}, \widetilde{Bark}, \widetilde{Meow}, \ldots\}$
- New operators: $`(Animal) = \widetilde{Animal}$ and $\$(\widetilde{Eats})(d) = Eats(d)$
- Example:

$$Sound(Concept) \to \mathbb{B}$$

$$Sound(`(Bark)) \land Sound(`(Meow)).$$

Intensional logic

- Back to the *ProducingSound*:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \land Bark(\underline{a})) \lor (Cat(\underline{a}) \land Meow(\underline{a})).$$

- Can we do better with intentional logic?

$$\forall a[Animal] : ProducingSound(a) \Leftrightarrow \exists s[Concept] : Sound(s) \land \$(s)(a).$$

- This is not good!

$$\forall a[Animal] : ProducingSound(a) \Leftrightarrow Bark(\underline{a}) \lor Meow(\underline{a}).$$

- Can this be made correct in intensional logic? Yes!

☆ Intensional logic

- Back to the *ProducingSound*:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \land Bark(\underline{a})) \lor (Cat(\underline{a}) \land Meow(\underline{a})).$$

- Can we do better with intentional logic?

$$\forall a[Animal] : ProducingSound(a) \Leftrightarrow \exists s[Concept] : Sound(s) \land \$(s)(a).$$

- This is not good!

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow Bark(\underline{a}) \lor Meow(\underline{a}).$$

- Can this be made correct in intensional logic? Yes!

Intensional logic

- Back to the *ProducingSound*:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \land Bark(\underline{a})) \lor (Cat(\underline{a}) \land Meow(\underline{a})).$$

- Can we do better with intentional logic?

$$\forall a[Animal] : ProducingSound(a) \Leftrightarrow \exists s[Concept] : Sound(s) \land \$(s)(a).$$

- This is not good!

$$\forall a[Animal] : ProducingSound(a) \Leftrightarrow Bark(\underline{a}) \lor Meow(\underline{a}).$$

- Can this be made correct in intensional logic? Yes!

Intensional logic

- Back to the *ProducingSound*:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \wedge Bark(\underline{a})) \vee (Cat(\underline{a}) \wedge Meow(\underline{a})).$$

- Can we do better with intentional logic?

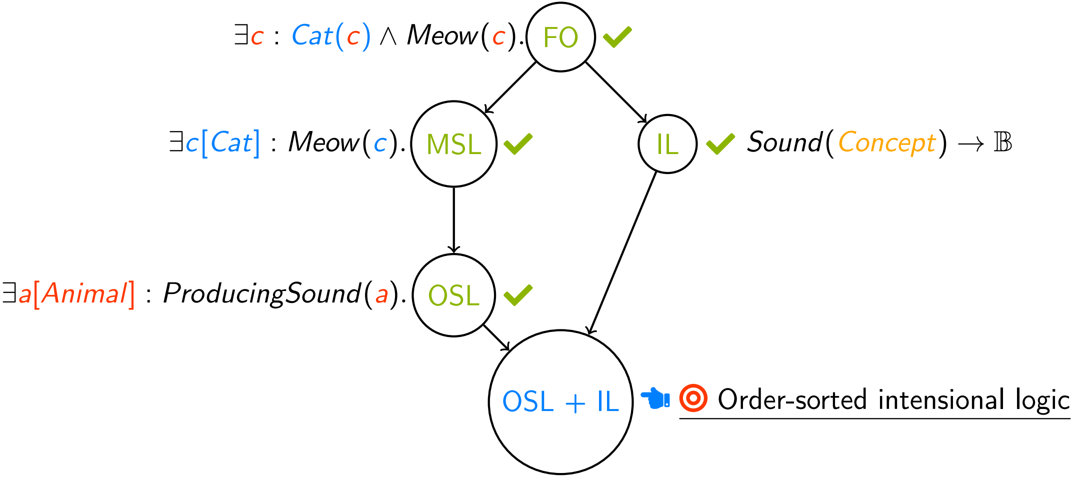
$$\forall a[Animal] : ProducingSound(a) \Leftrightarrow \exists s[Concept] : Sound(s) \wedge \$(s)(a).$$

- This is not good!

$$\forall a[Animal] : ProducingSound(a) \Leftrightarrow Bark(\underline{a}) \vee Meow(\underline{a}).$$

- Can this be made correct in intensional logic? Yes!

Roadmap


$\exists c : Cat(c) \wedge Meow(c).$
FO
$\exists c[Cat] : Meow(c).$
MSL
IL
$Sound(Concept) \to \mathbb{B}$
$\exists a[Animal] : ProducingSound(a).$
OSL
OSL + IL
Order-sorted intensional logic

Order-sorted intensional logic

- We defined *ProducingSound* as:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \wedge Bark(\underline{a})) \vee (Cat(\underline{a}) \wedge Meow(\underline{a})).$$

- We abbreviate it with:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow [[Bark(\underline{a})]] \vee [[Meow(\underline{a})]]$$

- Introduce new type:

$$Sound <: Concept \quad := \{\widetilde{Bark}, \widetilde{Meow}\}$$

- Then *ProducingSound* can be defined as:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow \exists s[Sound] : [[\$(\underline{s})(a)]].$$

Order-sorted intensional logic

- We defined *ProducingSound* as:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \land Bark(\underline{a})) \lor (Cat(\underline{a}) \land Meow(\underline{a})).$$

- We abbreviate it with:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow [[Bark(\underline{a})]] \lor [[Meow(\underline{a})]]$$

- Introduce new type:

$$Sound <: Concept \quad := \{\widetilde{Bark}, \widetilde{Meow}\}$$

- Then *ProducingSound* can be defined as:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow \exists s[Sound] : [[\$(\underline{s})(a)]].$$

Order-sorted intensional logic

- We defined *ProducingSound* as:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \wedge Bark(\underline{a})) \vee (Cat(\underline{a}) \wedge Meow(\underline{a})).$$

- We abbreviate it with:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow [\![Bark(\underline{a})]\!] \vee [\![Meow(\underline{a})]\!]$$

- Introduce new type:

$$Sound <: Concept \quad := \{\widetilde{Bark}, \widetilde{Meow}\}$$

- Then *ProducingSound* can be defined as:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow \exists s[Sound] : [\![\$(\underline{s})(a)]\!].$$

Order-sorted intensional logic

- We defined *ProducingSound* as:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow (Dog(\underline{a}) \wedge Bark(\underline{a})) \vee (Cat(\underline{a}) \wedge Meow(\underline{a})).$$

- We abbreviate it with:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow [\![Bark(\underline{a})]\!] \vee [\![Meow(\underline{a})]\!]$$

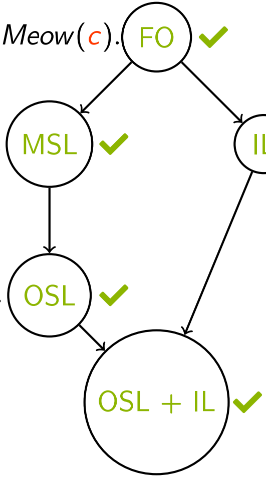
- Introduce new type:

$$Sound <: Concept \quad := \{\widetilde{Bark}, \widetilde{Meow}\}$$

- Then *ProducingSound* can be defined as:

$$\forall a[Animal] : ProducingSound(\underline{a}) \Leftrightarrow \exists s[Sound] : [\![\$(\underline{s})(a)]\!].$$

Conclusion



$\exists c : Cat(c) \wedge Meow(c).$

$\exists c[Cat] : Meow(c).$

$Sound(Concept) \to \mathbb{B}$

$\exists a[Animal] : ProducingSound(a).$

$\forall s[Sound] : \exists a[Animal] : [\![\$(s)(a)]\!].$

Conclusion

- Expressing subtyping polymorphism is a challenge in order-sorted logic.
- Intensional logic is required.
- Implicit guarding makes it compact and elegant.
- Complex well-typing relation (future work).

Conclusion

- Expressing subtyping polymorphism is a challenge in order-sorted logic.
- Intensional logic is required.
- Implicit guarding makes it compact and elegant.
- Complex well-typing relation (future work).

Conclusion

- Expressing subtyping polymorphism is a challenge in order-sorted logic.
- Intensional logic is required.
- Implicit guarding makes it compact and elegant.
- Complex well-typing relation (future work).

Conclusion

- Expressing subtyping polymorphism is a challenge in order-sorted logic.
- Intensional logic is required.
- Implicit guarding makes it compact and elegant.
- Complex well-typing relation (future work).

Takeaway: *Implicit type assertions and quantification over concepts provide elegant solution to subtyping polymorphism in OSL.*

Thank you for your attention! Questions?

Đorđe Markovic [*dorde.markovic@kuleuven.be*]
Marc Denecker [*marc.denecker@kuleuven.be*]

ICLP 2024 Dallas, TX, USA