Automating Leibniz's Theory of Concepts

Jesse Alama
Paul E. Oppenheimer
Edward N. Zalta

Center for the Study of Language and Information

Stanford University

alama@logic.at, {paul.oppenheimer,zalta}@stanford.edu
http://countersatisfiable.org/
http://peoppenheimer.org/
http://mally.stanford.edu/zalta.html

August 2015
CADE-25
Berlin

1. Intro
2. Background Theory
3. Representation
4. Fundamental Theorems
5. Workflow
6. Observations
7. Bibliography

Leibniz's Technical Work in Philosophy and his Vision

- In mathematics, Leibniz (independently of Newton) developed mathematical analysis
- Leibniz also proposed the goals: Characteristica Universalis and Calculus Ratiocinator.
- In Philosophy, he developed a theory of concepts:
  - Calculus of Concepts
  - Concept Containment Theory of Truth
  - Modal Metaphysics of Complete Individual Concepts
- Goal: Represent Leibniz's theory in a way that achieves his goal: use a formal language and derive the theorems with automated reasoning tools.
- Interest: Both the Kripke and the Lewis semantics of quantified modal logic is reconstructed; each theory is preserved.

Leibniz's Calculus of Concepts

- Let x, y, z range over concepts, taken as primitive.
- Let $x \oplus y$ ('the sum of x and y) be an operation on concepts.
- Let $x \preceq y$ ('x is included in y') be a relation on concepts, defined as:
  - $x \preceq y \equiv \exists z(x \oplus z = y)$
- Leibniz's axioms: idempotence, commutativity, and associativity of $\oplus$
- Structure: concepts are organized in a semi-lattice.
- Question: can we analyze $\oplus$ and $\preceq$?

Leibniz's Containment Theory of Truth

- Leibniz thought *predication* is a relation among concepts.
- To say 'Alexander is king' is to say:
  - The concept Alexander contains the concept king.
  - $\boldsymbol{c}_a \succeq \boldsymbol{c}_K$

  where 'the concept Alexander' is in some sense complete (contains all of Alexanders properties), 'the concept king' contains all the properties implies by being a king, and concept containment ($\succeq$) is the converse of concept inclusion ($\preceq$).
- Problem (raised by Arnauld): Hasn't Leibniz analyzed a contingent claim in terms of a necessary relation?
- Question: Can we analyze $\boldsymbol{c}_a$, $\boldsymbol{c}_K$, and $\succeq$?

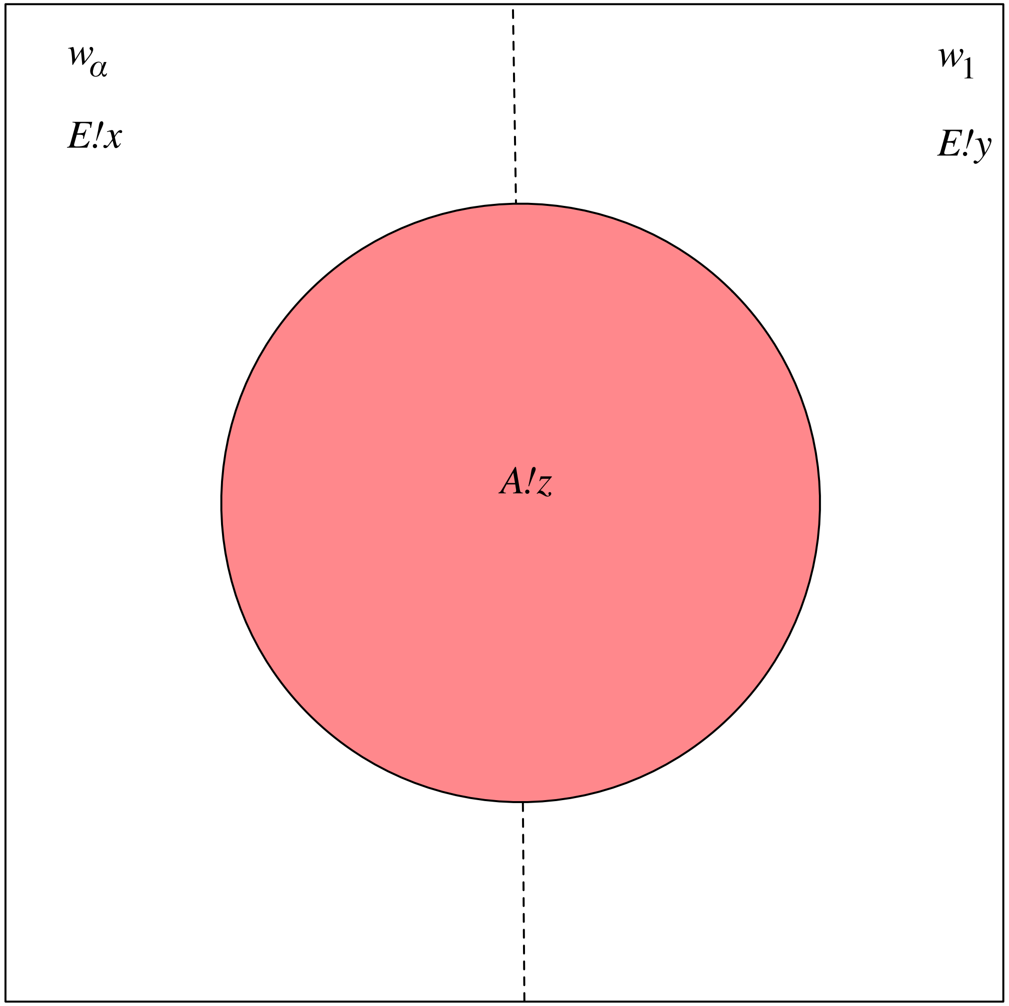
Leibniz's Modal Metaphysics of Concepts

- Consider the claim 'Alexander is king but might not have been'.
- For Leibniz, the mind of God contains many concepts of Alexander: $c_{a_1}, c_{a_2}, c_{a_3}, \ldots$
- Counterparts: each c_{a_i} is a a possible version of Alexander.
- Generally: for every thing in the world, the mind of God has many complete individual concepts for that thing.
- The complete individual concepts are partitioned into *compossible* concepts: each cell *could* be actualized into a possible world.
- God then ruled out those cells where people had no free will, and then actualized the one cell of compossible concepts that had the least amount of evil.
- Questions: How do we precisely analyze complete individual concepts without invoking God? Should we use Kripke semantics or Lewis's counterpart theory?

The Theory of Abstract Objects

- Use 2°, QML, with definite descriptions (rigid) and λ-terms (relational). One primitive predicate: concreteness ($E!$).
- Interpret in a fixed-domain semantics that validates BF ($\Diamond\exists\alpha\varphi \rightarrow \exists\alpha\Diamond\varphi$) and CBF ($\exists\alpha\Diamond\varphi \rightarrow \Diamond\exists\alpha\varphi$).
- Add one new mode of predication: xF (*x encodes F*)
  Cf. usual atomic formulas $F^n x_1 \ldots x_n$ ($x_1, \ldots, x_n$ *exemplify* F^n)
- Distinguish ordinary ($O!$) and abstract ($A!$) objects:
  - $O!x =_{df} \Diamond E!x$
  - $A!x =_{df} \neg\Diamond E!x$
- Axiom: Ordinary objects don't encode properties:
  $O!x \rightarrow \neg\exists F xF$
- Comprehension Schema: $\exists x(A!x \,\&\, \forall F(xF \equiv \varphi))$
- $x = y \ =_{df} \ ((O!x \,\&\, O!y \,\&\, \Box\forall F(Fx \equiv Fy)) \vee (A!x \,\&\, A!y \,\&\, \Box\forall F(xF \equiv yF)))$

A Frame with Two Possible Worlds



The Domain of Abstract Objects: A Plenitude

- $\exists!x(A!x \,\&\, \forall F(xF \equiv Fy))$
- $\exists!x(A!x \,\&\, \forall F(xF \equiv Fy \,\&\, Fz))$
- The Triangle:
  $\exists!x(A!x \,\&\, \forall F(xF \equiv T \Rightarrow F))$
- The Class of Gs:
  $\exists!x(A!x \,\&\, \forall F(xF \equiv \Box\forall y(Fy \equiv Gy)))$
- The Actual World:
  $\exists!x(A!x \,\&\, \forall F(xF \equiv \exists p(p \,\&\, F = [\lambda y\, p])))$
- The Number of Gs:
  $\exists!x(A!x \,\&\, \forall F(xF \equiv F \approx G))$
- The Null Set of ZF (‘$\emptyset_{ZF}$’):
  $\exists!x(A!x \,\&\, \forall F(xF \equiv \mathit{ZF}[\lambda y\, F\emptyset_{ZF}]))$,
  where ‘ZF’ denotes Zermelo-Fraenkel Set Theory

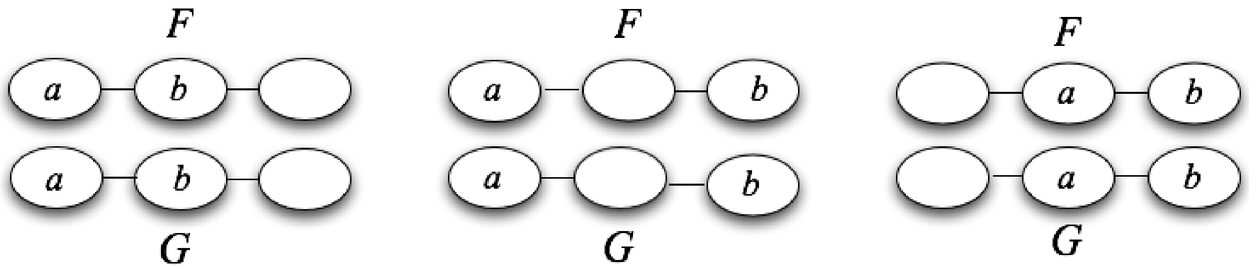
The Theory of Relations

- β-conversion, $[\lambda x_1 \dots x_n\, \varphi]y_1 \dots y_n \equiv \varphi^{y_1,\dots,y_n}_{x_1,\dots,x_n}$
- Second-order Comprehension: $\exists F^n \Box \forall x_1 \dots \forall x_n (F^n x_1 \dots x_n \equiv \varphi)$

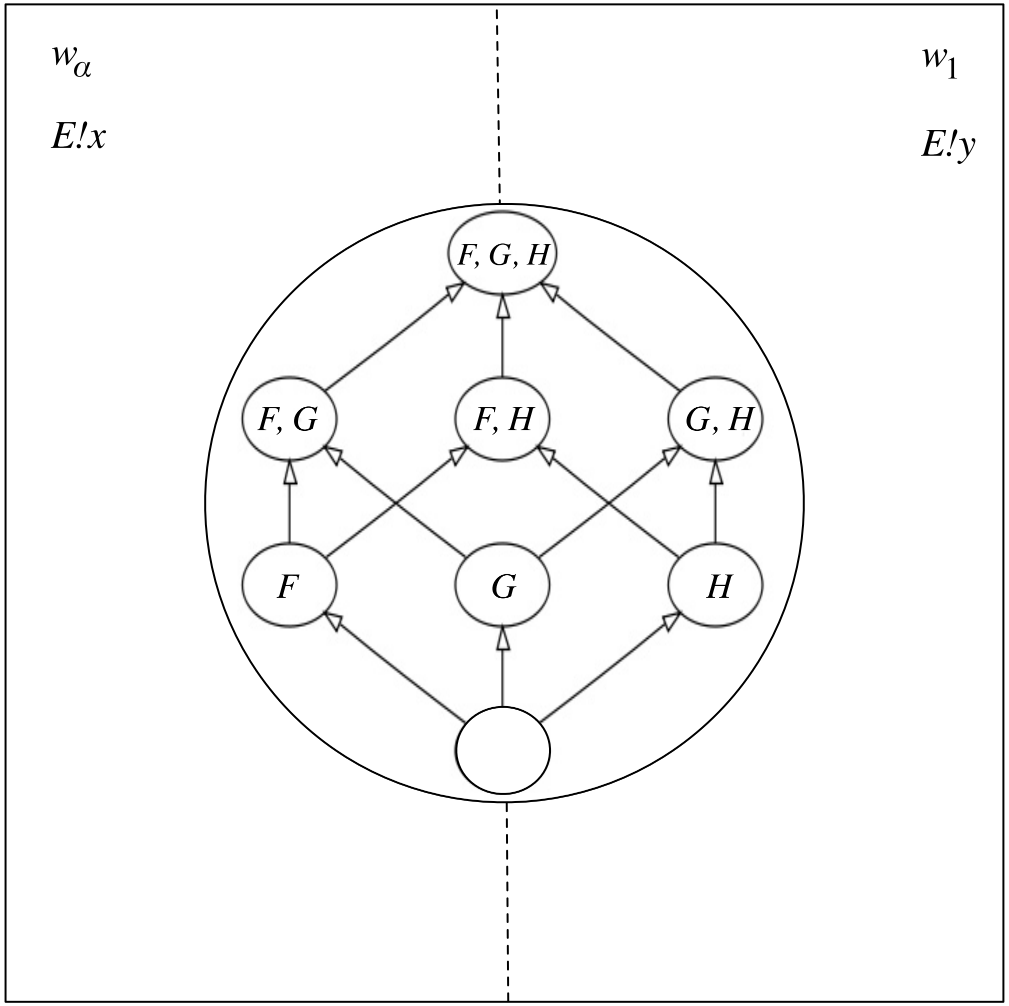
  $\exists F \Box \forall x (Fx \equiv \varphi)$ $\exists p \Box (p \equiv \varphi)$

  where φ has no free Fs (or ps) and no encoding subformulas.
- Identity for Properties: $F^1 = G^1 \;=_{df}\; \Box \forall x (xF^1 \equiv xG^1)$
- Identity for Propositions: $p = q \;=_{df}\; [\lambda y\, p] = [\lambda y\, q]$
- Identity for Relations: $F^n = G^n \;\equiv_{df}$ (where $n > 1$)

  $$(\forall x_1) \dots (\forall x_{n-1})([\lambda y\, F^n y x_1 \dots x_{n-1}] = [\lambda y\, G^n y x_1 \dots x_{n-1}]\ \&$$
  $$[\lambda y\, F^n x_1 y x_2 \dots x_{n-1}] = [\lambda y\, G^n x_1 y x_2 \dots x_{n-1}]\ \& \dots \&$$
  $$[\lambda y\, F^n x_1 \dots x_{n-1} y] = [\lambda y\, G^n x_1 \dots x_{n-1} y])$$
- Picture: To show $F^3 = G^3$, pick arbitrary a, b



The Domain of Abstract Objects



Leibniz's Calculus of Concepts Analyzed

- Concepts analyzed as abstract objects that encode properties:
  - $Concept(x) =_{df} A!x$
- Analysis of concept summation:
  - $x \oplus y =_{df} \iota z(Concept(z) \,\&\, \forall F(zF \equiv xF \vee yF))$
- Analysis of concept inclusion:
  - $x \leq y =_{df} \forall F(xF \rightarrow yF)$
- The principles governing semi-lattices fall out as theorems.
- cf. Zalta 2000

Leibniz's Containment Theory of Truth Analyzed

- The concept of ordinary individual u:
  - $\boldsymbol{c}_u \; =_{df} \; \iota x(C!x \,\&\, \forall F(xF \equiv Fu))$

  So the concept of Alexander is $\boldsymbol{c}_a$.
- The concept of property G:
  - $F \Rightarrow G \; =_{df} \; \Box\forall x(Fx \to Gx)$
  - $\boldsymbol{c}_G \; =_{df} \; \iota x(Concept(x) \,\&\, \forall F(xF \equiv G \Rightarrow F))$

  So the concept of *being a king* is $\boldsymbol{c}_K$.
- *Concept containment* is the converse of *concept inclusion*:
  - $x \succeq y \; =_{df} \; y \preceq x$

  That is, $x \succeq y$ if and only if $\forall F(yF \to xF)$.
- Analysis of 'Alexander is king':
  - The concept of Alexander contains the concept of being a king
  - $\boldsymbol{c}_a \succeq \boldsymbol{c}_K$

  $\boldsymbol{c}_a \succeq \boldsymbol{c}_K$ is derivable from the contingent premise Ka.

  This addresses Arnauld's objection.

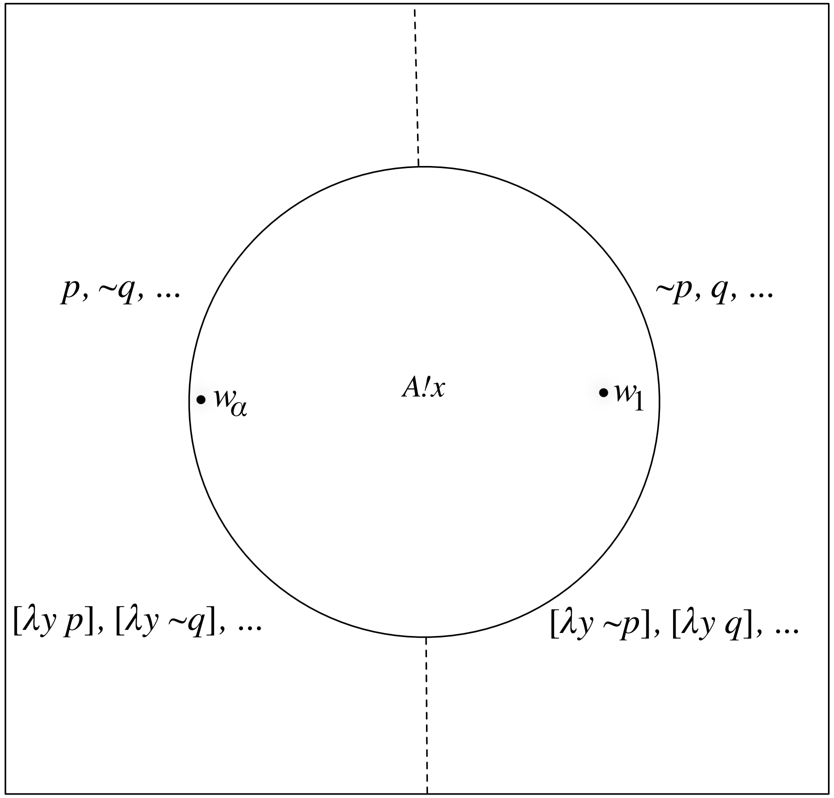
World Theory (Zalta 1993; Fitelson & Zalta 2007

$$Situation(x) =_{df} A!x \,\&\, \forall F(xF \rightarrow \exists p(F = [\lambda y\, p]))$$

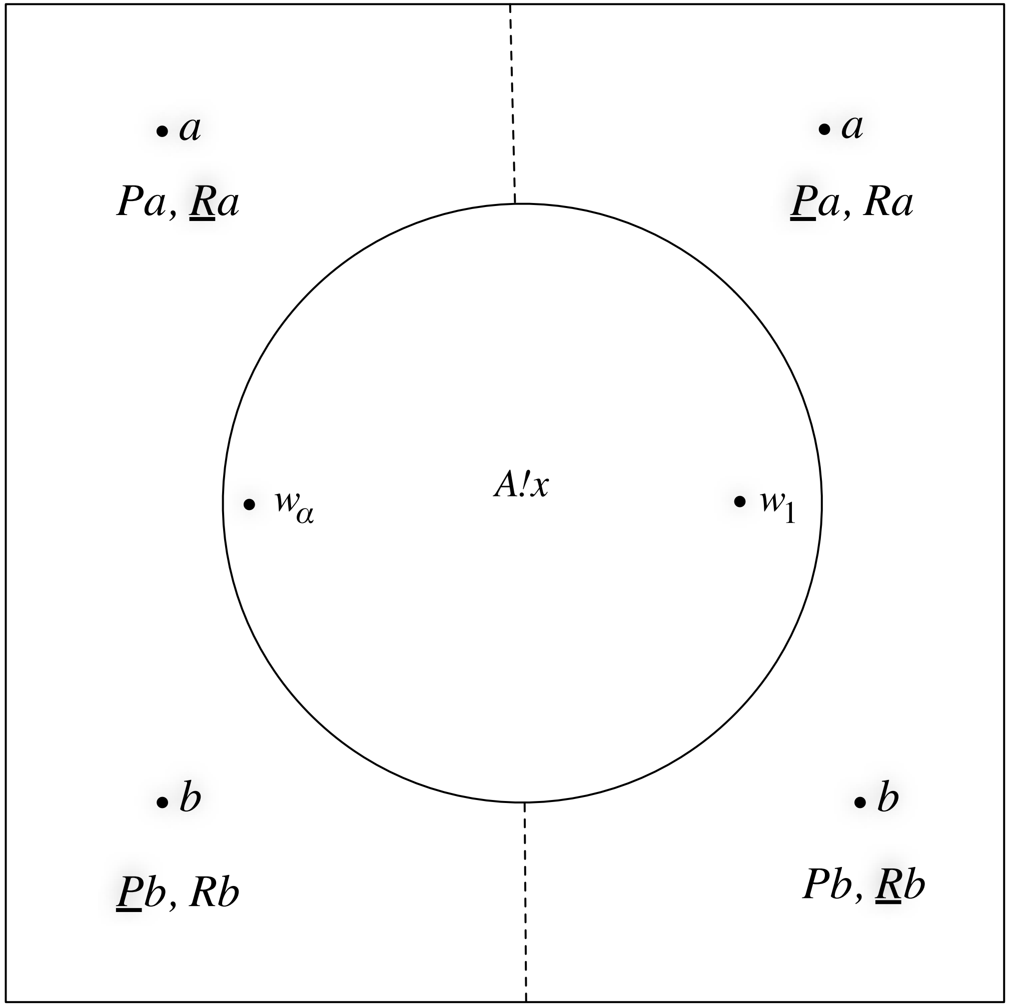
$$x \models p =_{df} Situation(x) \,\&\, x[\lambda y\, p]$$

$$World(x) =_{df} Situation(x) \,\&\, \Diamond \forall p((x \models p) \equiv p)$$

Kripke: $\Box p \equiv \forall w(w \models p)$ Lewis: $\Diamond p \equiv \exists p(w \models p)$

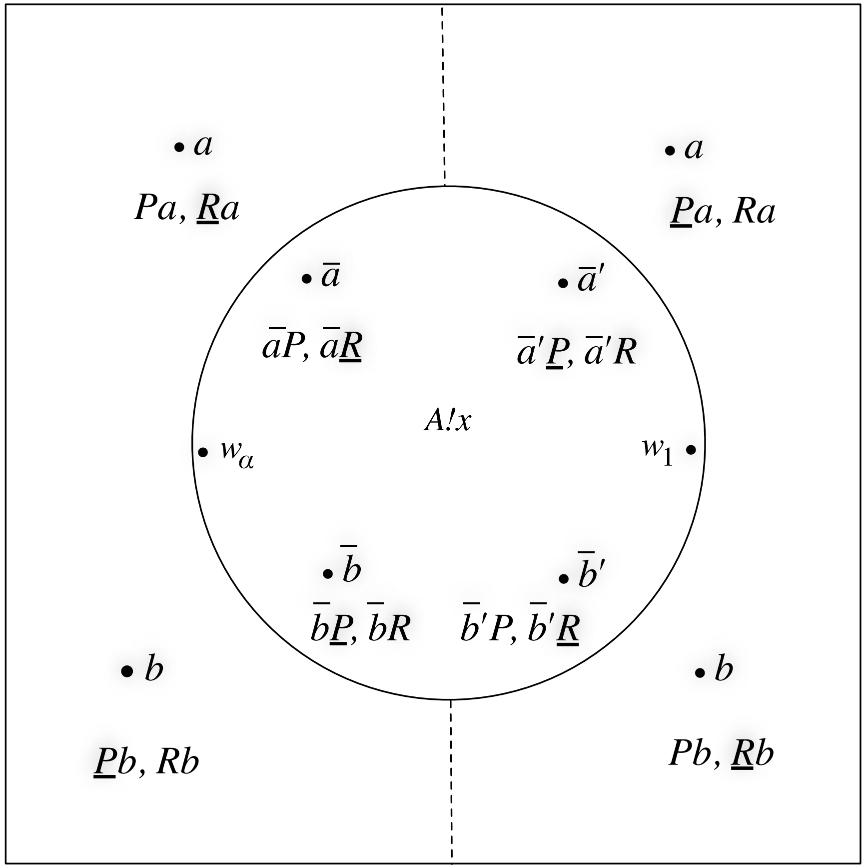


A Model with 2 Ordinary Objects, 4 Properties



Same Model with Abstract Objects

$$\overline{a} = \iota x(A!x \,\&\, \forall F(xF \equiv w_\alpha \models Fa)) \qquad \overline{a}' = \iota x(A!x \,\&\, \forall F(xF \equiv w_1 \models Fa))$$
$$\overline{b} = \iota x(A!x \,\&\, \forall F(xF \equiv w_\alpha \models Fb)) \qquad \overline{b}' = \iota x(A!x \,\&\, \forall F(xF \equiv w_1 \models Fb))$$



Leibniz's Modal Metaphysics Analyzed: I

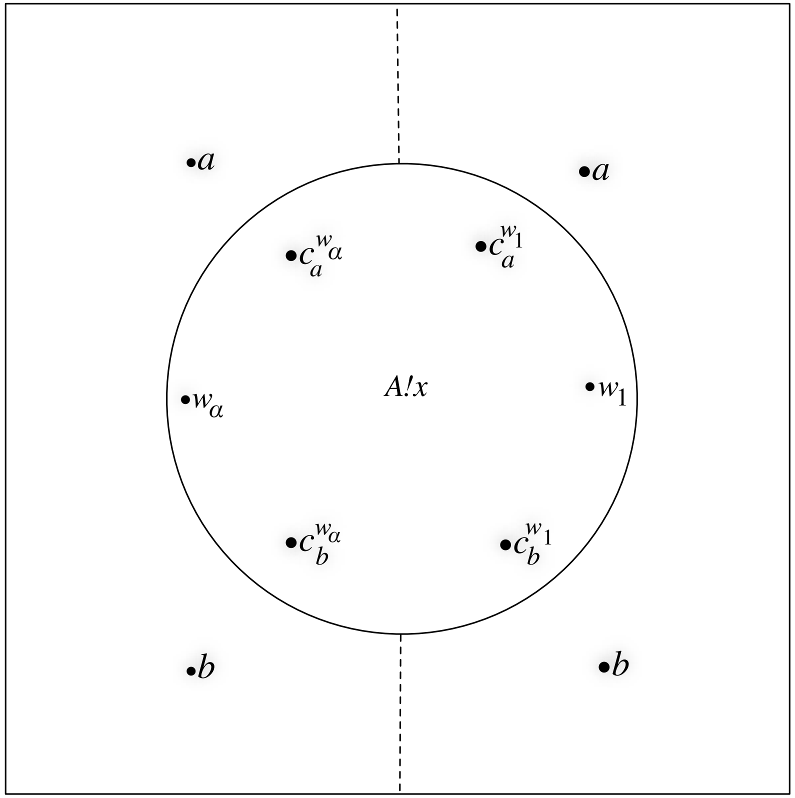
$$\mathbf{c}_u^w =_{df} \imath x(Concept(x) \;\&\; \forall F(xF \equiv w \models Fu))$$

$$RealizesAt(u, x, w) =_{df} \forall F((w \models Fu) \equiv xF)$$

$$AppearsAt(x, w) =_{df} \exists u RealizesAt(u, x, w)$$

$$IndividualConcept(x) =_{df} \exists w AppearsAt(x, w)$$

$$\vdash IndividualConcept(x) \rightarrow \exists! w AppearsAt(x, w)$$



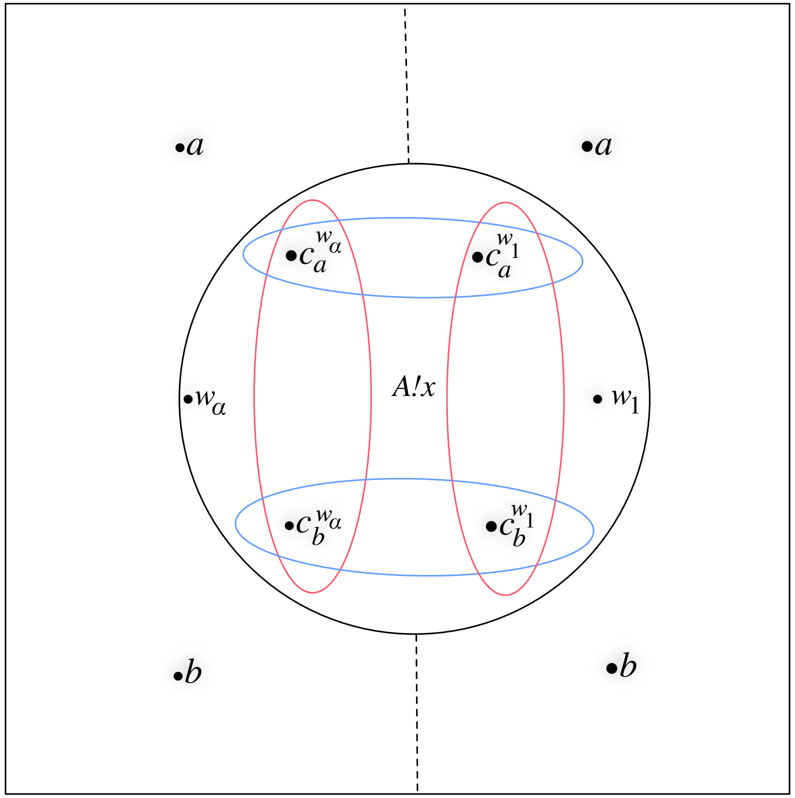
Leibniz's Modal Metaphysics Analyzed: II

$Compossible(c_1, c_2) \ =_{df} \ \exists w(Appears(c_1, w) \ \& \ Appears(c_2, w))$

$\vdash$ Compossibility partitions the individual concepts.

$Counterparts(c, c') \ =_{df} \ \exists u \exists w_1 \exists w_2 (c = \mathbf{c}_u^{w_1} \ \& \ c' = \mathbf{c}_u^{w_2})$

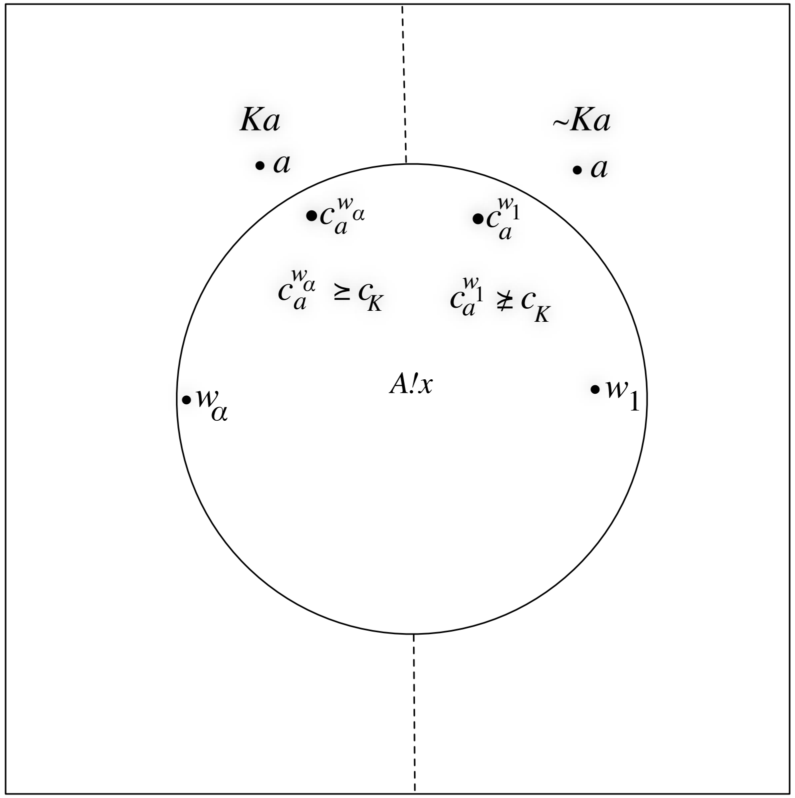
$\vdash$ Counterparthood partitions the individual concepts.



Leibniz's Modal Metaphysics Analyzed: III

Theorem 40a:

$(Fu \,\&\, \Diamond\neg Fu) \rightarrow [\boldsymbol{c}_u \succeq \boldsymbol{c}_F \,\&$
$\exists c(Counterparts(c, \boldsymbol{c}_u) \,\&\, c \not\succeq \boldsymbol{c}_F \,\&\, \exists w(w \neq \boldsymbol{w}_\alpha \,\&\, Appears(c, w)))]$



Representing Second-Order Syntax

- 4 Sorts for the language:
  `object(X)`, `property(F)`, `proposition(P)`, `point(D)`
- Distinguished point: `d`; Distinguished property: `e`
- Fx becomes `ex1_wrt(F,X,d)` and subject to:
  ```
  (! [F,X,D] : (ex1_wrt(F,X,D) =>
  (property(F) & object(X) & point(D)))).
  ```
- $\Box Fx$ becomes:
  ```
  (! [D] : (point(D) => ex1_wrt(F,X,D)))
  ```
- $O!x \ =_{df} \ \Diamond E!x$ becomes:
  ```
  (! [X,D] : ((object(X) & point(D)) => (ex1_wrt(o,X,D) <=>
     (? [D2] : (point(D2) & ex1_wrt(e,X,D2))))))).
  ```
- $A!x \ =_{df} \ \neg\Diamond E!x$ becomes:
  ```
  (! [X,D] : ((object(X) & point(D)) => (ex1_wrt(a,X,D) <=>
     ~(? [D2] : (point(D2) & ex1_wrt(e,X,D2))))))).
  ```

Representing the Second Mode of Prediction

- xF becomes `enc_wrt(X,F,D)`
- Sorting principle:

```
fof(sort_enc_wrt,type,
 (! [X,F,D] : (enc_wrt(X,F,D) =>
  (object(X) & property(F) & point(D))))).
```

Representing Identity Claims: I

- Identity for ordinary objects:

```
fof(o_equal_wrt,definition,
 (! [X,Y,D] : ((object(X) & object(Y) & point(D)) =>
  (o_equal_wrt(X,Y,D) <=>
   (ex1_wrt(o,X,D) & ex1_wrt(o,Y,D) &
    (! [D2] : (point(D2) => (! [F] : (property(F) =>
     (ex1_wrt(F,X,D2) <=> ex1_wrt(F,Y,D2)))))))))))).
```

- Analogous definition of identity for abstract objects, but they encode the same properties at all points.

Representing Identity Claims: II

- General Identity:

```
fof(object_equal_wrt,definition,
(! [X,Y,D] : ((object(X) & object(Y) & point(D)) =>
 (object_equal_wrt(X,Y,D) <=>
  (o_equal_wrt(X,Y,D) | a_equal_wrt(X,Y,D)))))).
```

- Tie defined identity into system identity:

```
fof(object_equal_wrt_implies_identity,theorem,
 (! [X,Y] : ((object(X) & object(Y)) =>
  (? [D] : (point(D) & object_equal_wrt(X,Y,D)) =>
   X = Y)))).
```

Representing Definite Descriptions

- $\boldsymbol{c}_u \;=_{df}\; \imath x(C!x \;\&\; \forall F(xF \equiv Fu))$
- ```
  fof(concept_of_individual_wrt,definition,
  (! [X,U,D] : ((object(X) & object(U) & point(D)) =>
   (concept_of_individual_wrt(X,U,D) <=>
    (ex1_wrt(c,X,D) & ex1_wrt(o,U,D) &
     (! [F] : (property(F) =>
      (enc_wrt(X,F,D) <=> ex1_wrt(F,U,D))))))))).
  ```
- Define: `is_the_concept_of_individual_wrt(X,U,D)`
- $\boldsymbol{c}_G \;=_{df}\; \imath x(Concept(x) \;\&\; \forall F(xF \equiv G \Rightarrow F))$
- Define: `is_the_concept_of_wrt(Y,F,D)`
  ```

Representing λ-Expressions

- Vacuously bound variable: $[\lambda z\, p]$
- Instance of λ-Conversion: $\Box([\lambda z\, p]x \equiv p)$,
- This becomes:

```
fof(existence_vac,axiom,
 (! [P] : (proposition(P) =>
  (? [Q] : (property(Q) & is_being_such_that(Q,P)))))).

fof(truth_wrt_vac,axiom,
 (! [P,Q] : ((proposition(P) & property(Q)) =>
  (is_being_such_that(Q,P) =>
   (! [D,X] : ((point(D) & object(X)) =>
    (ex1_wrt(Q,X,D) <=> true_wrt(P,D)))))))).
```

Fundamental Theorem

Theorem 40a (Zalta 2000):

$(Fu \ \& \ \Diamond\neg Fu) \rightarrow [\boldsymbol{c}_u \succeq \boldsymbol{c}_F \ \&$
$\exists c(Counterparts(c, \boldsymbol{c}_u) \ \& \ c \not\succeq \boldsymbol{c}_F \ \& \ \exists w(w \neq \boldsymbol{w}_\alpha \ \& \ Appears(c, w)))]$

Representation of the Fundamental Theorem

```
fof(theorem_40a,conjecture,
 (! [U,F] : ((object(U) & property(F)) =>
  ((ex1_wrt(o,U,d) & ex1_wrt(F,U,d) &
   (? [D] : (point(D) & ~ex1_wrt(F,U,D)))) =>
    (? [X,Y] : (object(X) & object(Y) &
     ex1_wrt(c,X,d) & ex1_wrt(c,Y,d) &
     is_the_concept_of_individual_wrt(X,U,d) &
     is_the_concept_of_wrt(Y,F,d) & contains_wrt(X,Y,d) &
      (? [Z] : (object(Z) & ex1_wrt(c,Z,d) &
      counterparts_wrt(Z,X,d) & ~contains_wrt(Z,Y,d) &
       (? [A,W] : (object(A) & object(W) &
        is_the_actual_world_wrt(A,d) & world_wrt(W,d) &
         ~equal_wrt(W,A,d) & appears_in_wrt(Z,W,d)))))))))))).
```

A Dual Fundamental Theorem

Theorem 40b [24]:

$$(\neg Fu \ \& \ \Diamond Fu) \rightarrow [\boldsymbol{c}_u \not\succeq \boldsymbol{c}_F \ \& \ \exists c(Counterparts(c, \boldsymbol{c}_u) \ \& \ c \succeq \boldsymbol{c}_F \ \& \ \exists w(w \neq \boldsymbol{w}_\alpha \ \& \ Appears(c, w)))]$$

Website with results

- http://mally.stanford.edu/cm/concepts/
- http://mally.stanford.edu/cm/concepts/main-theorems.html

How We Proved the Theorems

- To prove a conjecture with (deeply) defined terms, we kept adding, as premises, the definitions of, and axioms governing, the defined terms.
- We tried to avoid *hapax legomena*, i.e., terms that occur only once in the file, though often this is not the cause of the failure to discover a proof.
- Demo: symbol-summary theorem40a.p
- Often the proofs were discovered after throwing in everything but the kitchen sink. Then we faced the problem of pruning.
- Jesse Alama's tipi tools:
  `https://github.com/jessealama/tipi`
  `http://arxiv.org/abs/1204.0901`
- Demo: tipi minimize theorem29.orig.p

Dependencies

- Theorems can be graphed in terms of dependencies
- A change to the representation of one formula in one theorem has to be correspondingly made in all theorems that use the formula.
- Moreover, all those theorems containing the representation have to be rechecked for provability.
- Thus, any change to a prior axiom, definition, or theorem, requires checking all the theorems that depended on that changed formula. This is a variation on version control.
- Our solution:
  - each problem file for a theorem is annotated with the axioms, definitions, sorts, theorems, and lemmas that are used in its derivation.
  - create and maintain a makefile to generate a master file
  - regenerate problems from this master file
  - `http://mally.stanford.edu/cm/concepts/leibniz.html`

Representation Issues

- Eliminate Restricted Variables from Definitions:
  `(! [X1,...,Xn]: ((sort1(X1) & ... & sortn(Xn)) => (Definiendum(X1,...,Xn) <=> ...X1...Xn...))).`
- Formulate and prove only non-modal versions of theorems, unless the modal version is required for a proof elsewhere.
- In object theory, not every theorem is a necessary truth: we assume, and allow reasoning with, contingent premises, e.g., the Russell principle governing rigid definite descriptions. So the Rule of Necessitation can't be applied to arbitrary theorems.
- Example (Theorem 38): $Fu \equiv \boldsymbol{c}_u \succeq \boldsymbol{c}_K$
- Formulate instances of comprehension as they are needed:
  - instances of object comprehension as needed
  - instances of β-Conversion as needed

A Customized Theorem Prover for Object Theory?

- Long term goal: a customized theorem prover that recognizes classical object-theoretic syntax
- This would recognize which formulas have encoding subformulas: these are allowed in λ-expressions, on pain of paradox.
- Reasoning with contingent premises: recognize any dependencies on contingent axioms and theorems.
- But see Oppenheimer and Zalta 2011; object theory:
  - contains formulas that neither are terms themselves nor can be converted to terms by λ-abstraction, and therefore,
  - involves reasoning that seems to be capturable only in the logic of relational rather than functional type theory.
- Whether that causes a problem in adapting and customizing current automated reasoning technologies remains to be seen.

Bibliography: Primary Literature

- Leibniz, Gottfried Wilhelm:
  - C *Opuscules et fragments inédits de Leibniz*, L. Couturat (ed.), Paris, 1903.
  - CA *Correspondence with Arnauld*, translated by H. T. Mason, Manchester: Manchester University Press, 1967.
  - G *Die philosophischen Schriften von Gottfried Wilhelm Leibniz*, Volumes i – vii, C. I. Gerhardt (ed.), Berlin, 1875–90.
  - GI *Generales Inquisitiones de Analysi Notionum et Veritatum*, F. Schupp (ed.), Hamburg, 1982.
  - LP *Logical Papers*, G. H. R. Parkinson (ed. and trans.), Oxford: Clarendon, 1966.
  - PW *Philosophical Writings*, G. H. R. Parkinson (ed.), M. Morris and G. Parkinson (trans.), London: Dent & Sons, 1973.
  - T *Theodicy*, translated by E. Huggard, New Haven: Yale University Press, 1952.

Bibliography: Secondary Literature

- Fitelson, B., and E. Zalta, 2007, 'Steps Toward a Computational Metaphysics', *Journal of Philosophical Logic*, 36(2): 227–247.
- Kripke, S., 1963, 'Semantical Considerations on Modal Logic', *Acta Philosophica Fennica*, 16: 83–94.
- Lewis, D., 1968, 'Counterpart Theory and Quantified Modal Logic', *Journal of Philosophy*, 68: 113–126.
- Linsky, B., and E. Zalta, 1994, 'In Defense of the Simplest Quantified Modal Logic', *Philosophical Perspectives*, 8: 431–458.
- Oppenheimer, P. and E. Zalta, 2011, 'Relations versus Functions at the Foundations of Logic: Type-theoretic considerations', *Journal of Logic and Computation*, 21: 351–374.
- Zalta, E., 2000, 'A (Leibnizian) Theory of Concepts', *Philosophiegeschichte und logische Analyse/Logical Analysis and History of Philosophy*, 3: 137–183.
- Zalta, E., 1993, 'Twenty-Five Basic Theorems in Situation and World Theory', *Journal of Philosophical Logic*, 22: 385–428.