

An Ontology Formalization of Relation Type Hierarchy in Conceptual Structure Theory

Philip Nguyen, PhD, Principal Technical Specialist,
Department of Justice, Government of South Australia

Ken Kaneiwa, PhD, Senior Researcher,
National Institute of ICT, Kyoto, Japan

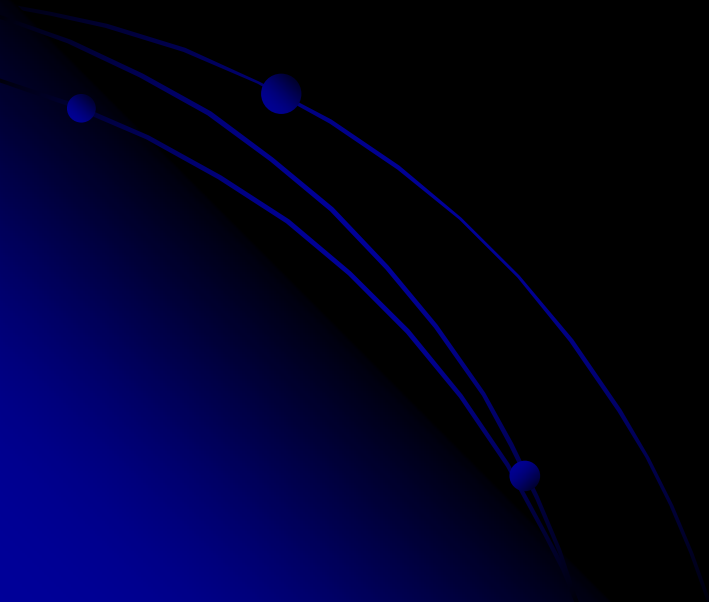
Dan Corbett, PhD, Principal Scientist,
Schafer Corp., Arlington, Va., USA

Minh-Quang Nguyen, PhD, Researcher,
Institut National de Recherche Scientifique, Montreal, Canada



Topics

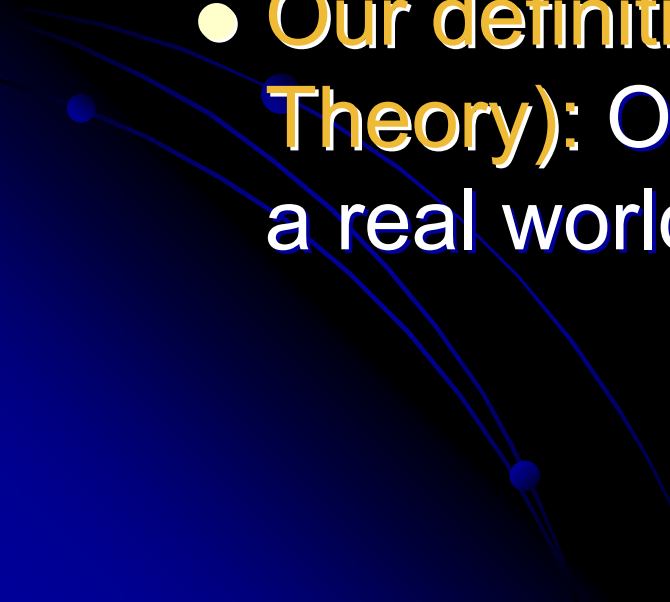
- Aims
- Ontology Formalization with Relation Type Hierarchy
- Potential Applications & Future Work



Aims

- **Ontology Formalization**
 - Focus on Relation Types
 - Maintain semantic linkages between concept type and relation type hierarchies through axiomatic semantics
- **Potential Application: Automatic Reasoning**
 - Query-Answering System
 - Semantic Web

Ontology Definition

- **Aristotle:** Ontology = anything that may be known about something in the world
 - **T. Gruber:** Ontology = a specification of a conceptualization
 - **Our definition (Conceptual Structure Theory):** Ontology = a mapping between a real world and an abstract world
- 

Ontology Formalization

Real World



Individuals I

Abstract World



T



ConceptTypes

RelationTypes

$conf$

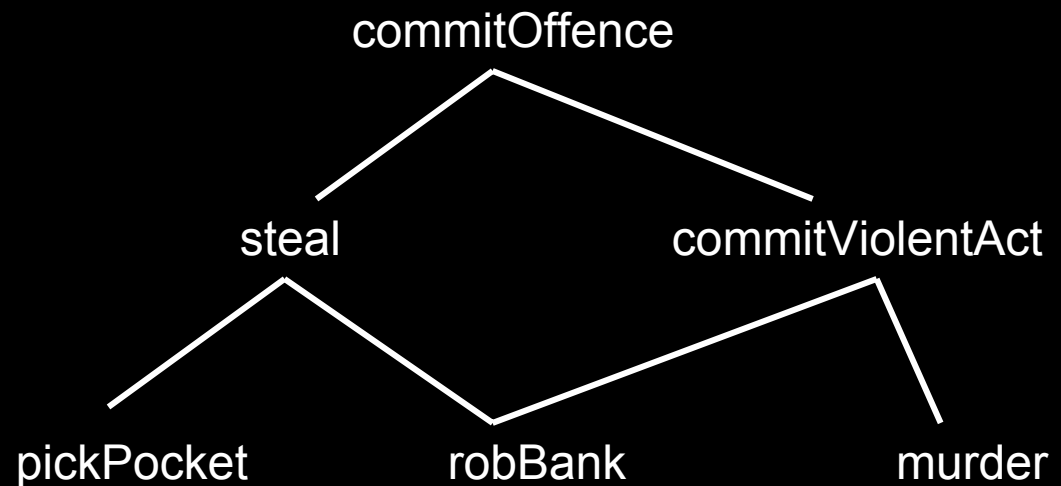
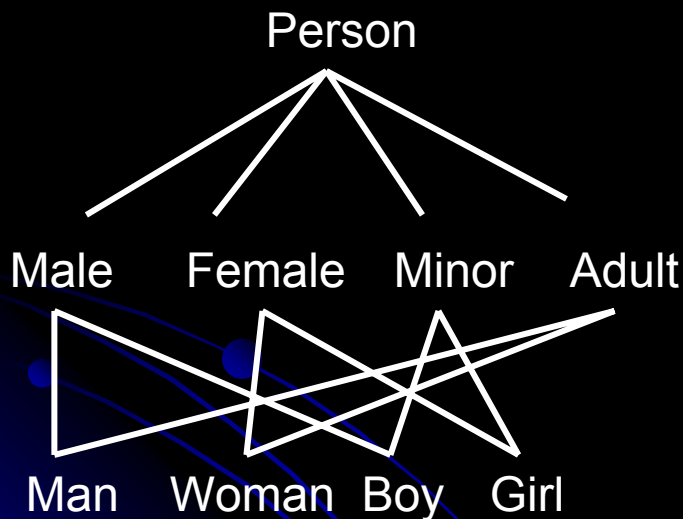
B

$$K = (T, I, \leq, conf, B)$$

D. Corbett, "Reasoning and Unification over Conceptual Graphs", Kluwer Academic Publishers, 2003

P. Nguyen and D. Corbett. "A Basic Mathematical Framework for Conceptual Graphs," IEEE Transactions on Knowledge and Data Engineering, vol. 18, no. 2, pp. 261-271, February, 2006

Concept & Relation Type Hierarchies (legal ontology)



- Formalization of a new idea as a new concept type or a new relation type is often arbitrary (usually domain and/or application dependent).
- Some relation types could be transformed into concept types (reducing the number of basic relation types).

Concept Types & Relation Types

$$K = (T, I, \leq, conf, B)$$

- T : hierarchies of concept & relation types
(ordered by the relation \leq)
- Concept types:
 $\text{Man} \leq \text{Person} \leq \text{LivingEntity}$
- Relation types:
 $\text{isChildOf}(\text{Person}, \text{Woman}, \text{Man})$
 $\text{isSonOf}(\text{Man}, \text{Woman}, \text{Man})$
 $\text{isSonOf} \leq \text{isChildOf}$

Individuals & Type Conformance

$$K = (T, I, \leq, \mathit{conf}, B)$$

- I : set of individuals (in the real world)
- conf : conformance relation

$$\mathit{conf} : I_c \rightarrow T_c$$

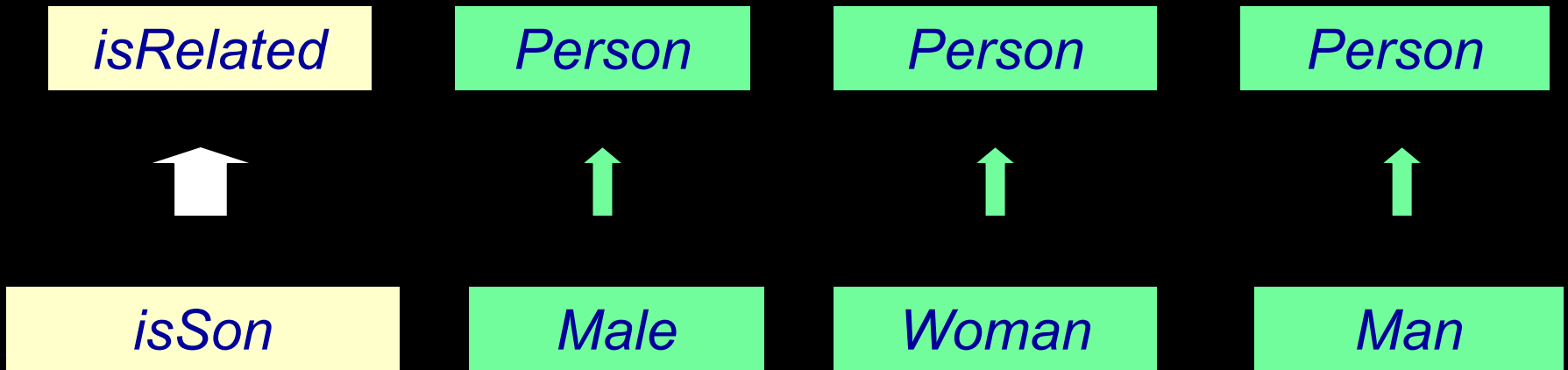
e.g.

$$\mathit{conf}(\text{"Peter"}) = \text{Man}$$

Peter is a man, a person and a living being, i.e.

Man = infimum (person, living being, ...)

Relations & Arguments (subsumption)



Individuals & Type Conformance

$$K = (T, I, \leq, \mathit{conf}, B)$$

- I : set of individuals & relations between them (in the real world)
- conf : conformance relation

$$\mathit{conf} : I_R \rightarrow T_R$$

e.g.

$r =$

isSon (Peter, Mary, John)

$\mathit{conf}(r) =$

isSon (Male, Woman, Man)

Relation Usage Pattern & Subsumption

$$K = (T, I, \leq, \mathbf{conf}, B)$$

$$B : T_R \rightarrow \tau(T_C)$$

B (relation) = tuple (ordered list) of concepts

e.g.

B (*isSon*) = [Male, Woman, Man]

B (*isRelated*) = [Person, Person, Person]

isSon \leq *isRelated* \Leftrightarrow

- o *isSon* is semantically included in *isRelated*
- o *their arguments also related in respective order:*
 - **Male \leq Person**
 - **Woman \leq Person**
 - **Man \leq Person**

Relations & Usage Pattern (notation)

$B(isSon) = [Male, Woman, Man] \Leftrightarrow$
 $isSon(Male, Woman, Man)$

Male, Woman, Man are arguments of *isSon*



Argument Completion (type inheritance)

commitOffence

*Offender, OffenceVictim, OffenceAct, OffenceInstrument,
OffenceMotive*



steal

Thief

*steal**

*Thief, TheftVictim, OffenceAct: <stealing>,
OffenceInstrument, StolenObject*

Type arguments go down, but not instance arguments

John steals from Mary \Rightarrow John commits an offence against Mary
(but the reverse is not true)

Argument Completion (type inheritance)

- steal (Thief)
- commitOffence (Offender, OffenceVictim, OffenceAct, OffenceInstrument, OffenceMotive)
- steal \leq commitOffence
- steal*(Thief, TheftVictim, OffenceAct: <stealing>, OffenceInstrument, StolenObject)

Argument Completion (instance generalization)

steal

Thief



pickPocket

Larcenist, Victim, StolenAmount

John picks \$5.00 from Mary's pocket \Rightarrow John steals \$5.00 from Mary
(but the reverse is not true)

\Rightarrow Instance arguments go up

pickPocket (Larcenist: John, Victim: Mary, StolenAmount: \$5.00) \Rightarrow
*steal**(Thief: John, Victim: Mary, StolenObject: <money, \$5>)

Argument Completion (instance generalization)

- pickPocket (Larcenist, Victim, StolenAmount)
- steal (Thief)
- pickPocket \leq steal
- steal*(Thief, Victim, StolenObject)

John picks \$5.00 from Mary's pocket



John steals \$5.00 from Mary

(but the reverse is not true)

Property Propagation (axiomatic semantics)

- **Type Inheritance:** For any type, its arguments and properties are inherited by all of its instances, and by all of its subtypes.
- **Instance Generalization:** For any instance of a type and for any supertype of that type, one can build another instance of that supertype such that the arguments and properties of the first instance also hold true for the second instance.
- **Summary:** Type arguments and properties go down while instance arguments and properties go up

Query-Answering System

(legal reasoning)

Facts:

- Any offender would have a record with Police.
- Children in a dysfunctional family are more likely to offend.
- Children in a family whose parents are often absent are monitored by a welfare agency (for possible assistance).
- John's parents are in jail.

Questions:

- Is John being monitored by a welfare agency?
- Does John have a Police record?

Query-Answering System (legal reasoning)

isInDysfunctionalFamily

Person, Offence: <moreLikely> <hasPoliceRecord>



hasAbsentParents

Person, MonitoringWelfareAgency



hasParentsInJail

Person: MARK

hasParentsInJail (Person: MARK, MonitoringWelfareAgency, Offence:
<moreLikely><hasPoliceRecord>)*

Query-Answering System

(legal reasoning)

Knowledge Base (Ontology & Database):

- hasParentsInJail (Person)
- hasAbsentParents (Person, MonitoringWelfareAgency)
- isInDysfunctionalFamily (Person, Offence: <moreLikely>)
- Offence: <hasPoliceRecord>
- hasParentsInJail (Person: MARK)
- hasParentsInJail < hasAbsentParents < isInDysfunctionalFamily

Answer:

- hasParentsInJail*(Person: MARK, MonitoringWelfareAgency, Offence: <moreLikely><hasPoliceRecord>)

Future Work

(Predicate of Predicates)

Meta-relation:

- causes (collapses (Bank: Lehman Brothers), crashes (StockMarket: World))
- crashes (StockMarket: World) = follows (crashes (StockMarket: America), crashes (StockMarket: Europe), crashes (StockMarket: Asia))

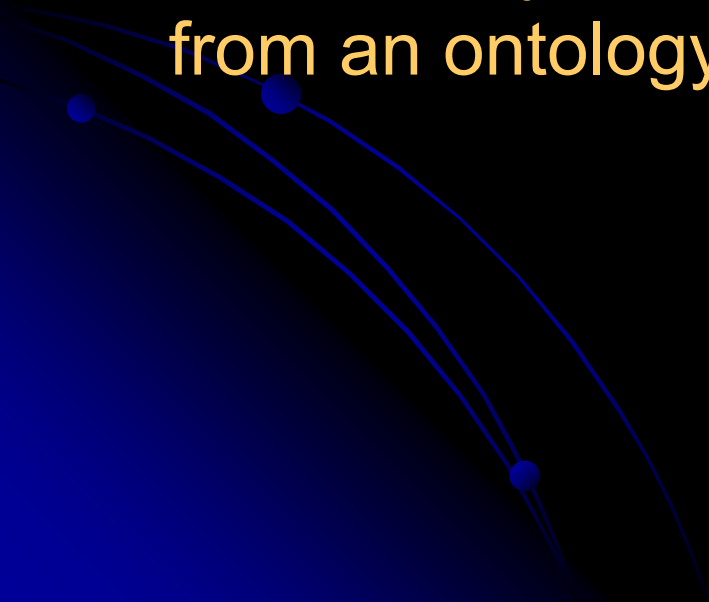


causes (


- collapses (Bank: Lehman Brothers),
- follows (
 - crashes (StockMarket: America),
 - crashes (StockMarket: Europe),
 - crashes (StockMarket: Asia)))

Future Work

(Denotational Semantics)

- From axiomatic semantics to denotational semantics
 - Recursively define an ontology with n individuals from an ontology with $(n-1)$ individuals.
- 

Conclusion

- Relation Type Hierarchy with semantic linkage to concept type hierarchy.
 - Axiomatic semantics with inference rules for propagation of arguments and properties.
 - Application: Automated reasoning, e.g., Query-answering system for legal reasoning.
- 

Thank You!

