Knowledge Representation: Knowledge Bases

Wojciech Jaworski

Institute of Informatics
The University of Warsaw

Database vs Knowledge Base

- Database: data collection kept according to certain rules
- Knowledge Base: formal representation of data from a certain discipline, which allows inferring.
- Database's structure (set of entities and relations) is specified by its diagram. Knowledge base's structure is ontology.
- In database all information is kept in structures equivalent to the EC logic.
- Entities' characteristics are described by triggers, procedures kept in base, and application, which uses database.
- In knowledge base, there is one language with which one can represent structures, data, and theory.
- We extract knowledge from knowledge base by inferring.
- Inferring procedures are separated from knowledge.

Expert Systems

- Applications which use knowledge bases to work.
- Due to the separation of inferring from knowledge, expert systems can explain and give reasons for what they are doing.
- Expert systems' frame: expert system without knowledge.
- CLIPS (C Language Integrated Production System)

Table of Contents

Project CYC (enCYClopedia)

- Cyc is an artificial intelligence project that attempts to assemble a comprehensive ontology and knowledge base of everyday common sense knowledge, with the goal of enabling AI applications to perform human-like reasoning.
- Components:
  - knowledge representation's language - CycL;
  - ontology whose domain is all of human consensus reality (terms, classes, and relations);
  - theory of the world (set of statements, facts, rules, heuristics, etc. which links ontological terms);
  - inferring system.
- Right now Cyc contains:
  - almost 500.000 terms;
  - out of that 26.000 are relations;
  - about 5.000.000 facts (statements) about those terms.

CYC's

- Thing, Intangible Thing, Individual, Relations, Sets, Collections, Paths, Logic, Math
- Space; Spatial Paths; Borders, Geometry; Movement; Artifacts; Events & Scripts; Time
- Physical Objects; Materials, Parts & Statics; State Change Dynamics; Physical Agents; Plans & Goals; Actors & Actions; Agents
- Weather; Natural Geography; Chemistry; Physical Artifacts; Astronomy; Earth & Solar System
- Organization; Organizational Actions; Types of Organizations; Organizational Plans; Law; Political Geography; Nations, Governments & Geopolitics; Business; Military Organizations
- Living Things; Ecology; Plants & Animals; Social Behavior
- Sentient Beings; Emotion, Perception & Belief; Human Anatomy & Physiology; Products & Devices; Conceptual Works; Literature & Works of Art; Human Behavior & Actions; Vehicles, Buildings & Weapons; Mechanical & Electrical Devices; Software; Language; Social Relations & Culture
- Human Activities; Business & Commerce; Politics & Warfare; Sports, Recreation & Entertainment; Professions & Occupations; Purchasing & Shopping; Social Activities; Transportation & Logistics; Travel & Communication; Everyday Living
- Domain-Specific Knowledge (Chemistry, Computer Security, Healthcare, ...)
- Domain-Specific Facts and Data

CycL Language

- Knowledge representation's language used by CYC.
- First order predicate calculus extended for identities, presumptions, skolem normal forms, and elements of second order logic, such as:
  - quantification over predicates;
  - metalanguage predicates.
- CycL uses *circumscription*, it assumes that unique names exist, and, when it is suitable, uses the closed world assumption.
- CycL syntax is based on LISP.
- *Circumscription* — logic in which for a given theory we narrow ourselves to its minimal models (similar to CWA).

Inference Engine)

- Inference Engine allows answering to user questions.
- Classic methods of reasoning are not scalable to a size of knowledge bases such as Cyc.
- Inference engine uses classic reasoning rules such as modus ponens, modus tollens, universal, and existential quantification.
- Their special case are reasoning mechanism such as inheritance, or automated classification.
- Cyc does greedy search in the proof space is steered by heuristics.
- Cyc contains also reasoning modules dedicated for certain classes of reasoning handling, for example defining inclusion/disjoint of collections, reasoning about identity, time, or mathematical inferring.

Microtheories — Knowledge Base Structure

- Cyc's knowledge is divided into thousands of "microtheories".
- Every one of them is a set of statements, which concern the same set of terms.
- Microtheories focus on a certain discipline of knowledge, have a certain level of detail, concern a certain time interval, etc.
- Microtheory mechanism allows to represent statements, which are, in general, in contradiction to each other.
- Microtheories allows to focus the reasoning process. That improves efficiency of the system.
- Splitting knowledge into microtheories is done accordingly to how it is represented in brain.

Natural Language Processing Subsystem

- Components:
- Lexicon
  - words from English language are attributed to constants e.g. `#$Light-TheWord`,
  - using CycL formulas, those constants are attributed to grammatical categories, terms represented by a word e.g. `#$LightEnergy`, or `#$LightingDevice`;
- Syntactic Parser
  - works using phrase-structure grammar;
  - returns all possible parse trees;
- Syntactic Interpreter
  - turns parse trees into CycL formulas;
  - works according to the composition rule;
  - templates, which are filled with terms responsible for their arguments, are attributed to verbs.

Natural Language Processing Subsystem

- Knowledge base in natural language processing is a model, which is extended with information extracted from texts.
- Cyc's knowledge allows settling semantical ambitiousness.
- Right now, Cyc is capable to interpret only a part of English language.
- The goal is to allow it to communicate with an user in natural language.

Semantic Integration Bus

- Tool for integrating Cyc with external data sources, such as databases, spreadsheets, web pages, text documents.
- External information is provided in knowledge base as virtual facts. That allows to use them in reasoning.
- Cyc interprets database records as implicit CycL formulas. In a similar manner, it process text fields.
- Integration is done semiautomatic.

Learning and Development

- The goal is to make Cyc able to learn at its own.
- Right now, it can learn new terms from Wikipedia.
- New statements are constantly added to knowledge base, by manual and automated methods.
- Cyc adds by itself a large amount of facts. It is an effect of reasoning.
- Right now, Cyc contains about 10% of knowledge required to let it develop by itself.

Distributions

- OpenCyc (`opencyc.org`) is a simplified base provided with Open Source license.
- OpenCyc is available as an application, ontology in OWL format, and as a semantic web `sw.opencyc.org`.
- ResearchCyc (`researchcyc.cyc.org`) is a full-scale Cyc, distributed under commercial and research licenses.

Distributed Artificial Intelligence

- Dividing knowledge into microtheories allows to reason in distributed environment.
- It resembles humans cooperation and specialization:
  - common man contains knowledge, which allows him/her to communicate with others
  - but no one posses all knowledge,
  - instead, we have experts, whom we might ask for a help.
- One might implement this phenomenon:
  - every agent posses a general knowledge base about basic facts
  - and deep knowledge in a disciple, in which it specializes
- If during its reasoning process an agent concludes that a problem exceeds its knowledge, then it might communicate with an expert agent.
- Common ontology is required for allowing such communication.

Platform for constructing applications which support analyses making

- Asking questions using natural language (not in SQL).
- Finding answers in a result of reasoning (not from pattern finding), with reason giving.
- General knowledge base supported with disciple knowledge.
- Examples:
  - Analyzing medical data concerning population, diseases, diagnoses, etc.
    - ★ *Which patients had a heart attack less than 2 weeks prior to a coronary artery bypass graft (CABG) between 2000 and 2005?*
  - Financial analyses: defining productivity of corporations, market trends, size of events' influence on market...
    - ★ *What type of derivative, whose underlyer is equity in African corporations, is most heavily invested in by GS corporate customers in Japan?*
  - Modeling cultural patterns and norms in various cultures. Predicting how members of a particular society will react to a certain behavior.

CycSecure: Network Intrusion Protection

- Predicts attack, advises how to repair
- Knowledge base about security holes and their influence on web operation.
- Web model, on which simulations can be made
- Simulating cyber-attacks

Table of Contents

Wordnet (wordnet.princeton.edu)

- Semantic Network contains words (nouns, verbs, adverbs, and adjectives) grouped according to synonymic relation (synsets).
- 117 000 synsets (terms).
- Words with multiple meanings belongs to multiple synsets.
- Every synset has a short description and a sentence describing its usage.
- Synsets are linked to each other by relation of being more general/detailed (hyperonymy, hyponymy, subsumption, relation *is a*), which generates a hierarchy of terms.
- Wordnet distinguishes terms from individuals and has relation of belonging to a term.
- Relation of being a part is introduced (meronymy). It is inherited by terms more specific.
- There exists also relations linking words with a common root.
- Słowosieć (plWordnet, /plwordnet.pwr.wroc.pl) is a Polish version of Wordnet. It contains 73839 synsets.

GeoNames

- Open geographical data base `www.geonames.org`.
- Over 10 million geographic names.
- 7,5 million of unique places.
- Names are divided for 645 categories e.g. (second order administration region, bay, lake district, village, gas pipeline, etc.)
- Every name is mated with its aliases and different language versions: New York, Bandaraya New York,Big Apple,Cathair Nua Eabhraic,Eabhraig Nuadh,Efrog Newydd,Evrek Nowydh,Nowy Jork,...
- Information about administrative affiliation, geographic coordinates, inhabitants, meters above sea level.

Freebase

- Freebase is a large collaborative knowledge base consisting of metadata composed mainly by its community members.
- It is an online collection of structured data harvested from many sources, including individual, user-submitted wiki contributions.
- Freebase aims to create a global resource which allows people (and machines) to access common information more effectively.
- Unlike the W3C approach to the semantic web, which starts with controlled ontologies, Freebase adopts a folksonomy approach, in which people can add new categories (much like tags), in a messy sprawl of potentially overlapping assertions.
- 46,295,148 Topics — entities (25 November 2014)
- 2,677,948,113 Facts (25 November 2014)

Graphs

- Freebase data is stored in a data structure called a graph.
- A graph is composed on nodes connected by edges.
- In Freebase, the nodes are defined using /type/object and edges are defined using /type/link.
- By storing the data as a graph, Freebase can quickly traverse arbitrary connections between topics and easily add new schema without having to change structure of the data.

Topics

- Freebase has over 39 million topics about real-world entities like people, places and things.
- Since Freebase data is represented a graph, these topics correspond to the nodes in the graph.
- Examples of the types of topics found in Freebase:
  - Physical entities, e.g., Bob Dylan, the Louvre Museum, the Saturn planet,
  - Artistic/media creations, e.g., The Dark Knight (film), Hotel California (song),
  - Classifications, e.g., noble gas, Chordate,
  - Abstract concepts, e.g., love,
  - Schools of thoughts or artistic movements, e.g., Impressionism.
- Some topics are notable because they hold a lot of data (e.g., Wal-Mart), and some are notable because they link to many other topics, potentially in different domains of information.
- For example, abstract topics like love, poverty, chivalry, etc. don't have many properties associated with them but they appear often as book subjects, film subjects, etc. making them more notable.

Types

- Any given topic can be seen for many different perspectives for example:
  - Bob Dylan was a song writer, singer, performer, book author, and film actor;
  - Leonardo da Vinci was a painter, a sculptor, an anatomist, an architect, an engineer, ...;
  - Love is a book subject, film subject, play subject, poetry subject, ...;
  - Any city is a location, potentially a tourist destination, and an employer of civil servants.
- In order to capture this multi-faceted nature of many topics, the concept of types is introduced in Freebase.
- Topics in Freebase can have any number of types assigned to them.
- The topic about Bob Dylan is assigned several types: the song writer type, the music composer type, the music artist (singer) type, the book author type, etc.

Properties

- Each type carries a different set of properties germane to that type. For example,
  - The music artist type contains a property that lists all the albums that Bob Dylan has produced as well as all the music instruments he was known to play;
  - The book author type contains a property that lists all the books Bob Dylan has written or edited, as well as his writing school of thoughts or movement;
  - The company type contains many property for listing a company's founders, board members, parent company, divisions, employees, products, year-by-year revenue and profit records, etc.
- Thus, a type can be thought of as a conceptual container of properties that are most commonly needed for describing a particular aspect of information.

Table of Contents

CyCL Syntax

- Constants are preceded by `#$`, e.g. `#$Dog`.
- Predicates are in brackets, first a name of a predicate, then its arguments: `(#$isa #$Burek #$Dog)`
- Constants representing functions contain in its name `Fn`, e.g. `#$BirthFn`.
- Terms are in brackets, first a name of a function, then its arguments: `(#$BirthFn #$Burek)`
- Conjunctions: `#$and`, `#$or`, `#$not`, `#$implies`:

```
(#$or (#$isa #$Burek #$Dog)
      (#$not (#$isa #$Burek #$Dog)))
```

- Quantifiers: `#$forAll`, `#$thereExists`:

```
(#$forAll ?X (#$implies (#$isa ?X #$Dog)
                        (#$not (#$isa ?X #$Cat))))
```

- Variables are preceded with `?`, e.g. `?X`.

Collections and Individuals

- `#$Collection` — term, collection
- `#$Individual` — individual, object, instance of concept
- `(#$isa X Y)` means:
  `X` is the object belonging to the term `Y`;
  `X` is the instance of the collection `Y`.
- `(#$genls X Y)` means that
  the term `Y` is more general than the term `X`;
  every instance of the collection `X` is also an instance of the collection `Y`.
- `#$genls` is partially ordered (reflexive, antisymmetric, and transitive)
- `(#$disjointWith X Y)` — terms `X` i `Y` are disjoint.

OpenCyc Interface

Index Frame

- It is used for showing information about a selected variable from a knowledge base
- It contains a list of operations which can be done on a selected variable, as well as a list of related assertion types.
- By clicking on a green button located next to a name of relation we ask for possible values of one of its arguments, e.g. `(#$isa #$RoadVehicle ?ARG2).`
- By clicking on a red button, we get a hierarchy of terms.

Assertion Display Frame

- Contains assertions concerning a selected constant.
- Its content depends from an option selected in Index Frame
- To see detailed information about a selected assertion, one has to click a button.
- Meanings of buttons:
  - white — monotonically true
  - yellow — true by assertion
  - blue — Forward Rule
  - violet — Backward Rule
  - red — negation
  - green — inferred

Arity and Argument Types

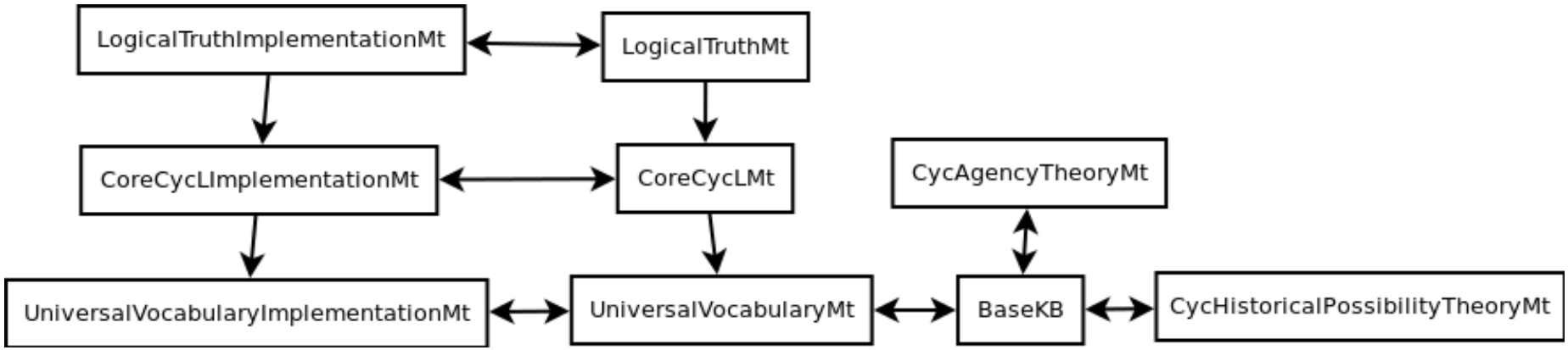
- `#$arity` - predicate which determines number of function's or predicate's arguments: `(#$arity #$BirthFn 1) (#$arity #$arity 2)`
- `#$argIsa`, `(#$argIsa #$performedBy 1 #$Action)` means that a first argument `#$performedBy` has to be an instance `#$Action`.
- `#$argGenl (#$argGenl #$penaltyForInfraction 2 #$Event)` means that a second argument `#$penaltyForInfraction` has to be a subcollection `#$Event`.
- `#$arg[N]Isa`, `#$arg[N]Genl`
- `#$resultIsa` indicate a collection to which belongs function's result.
- `#$resultGenl` indicate a collection, which subcollection is function's result.
- `#$UnaryFunction`,`#$BinaryPredicate`, `#$BinaryFunction`

Collections of Collections

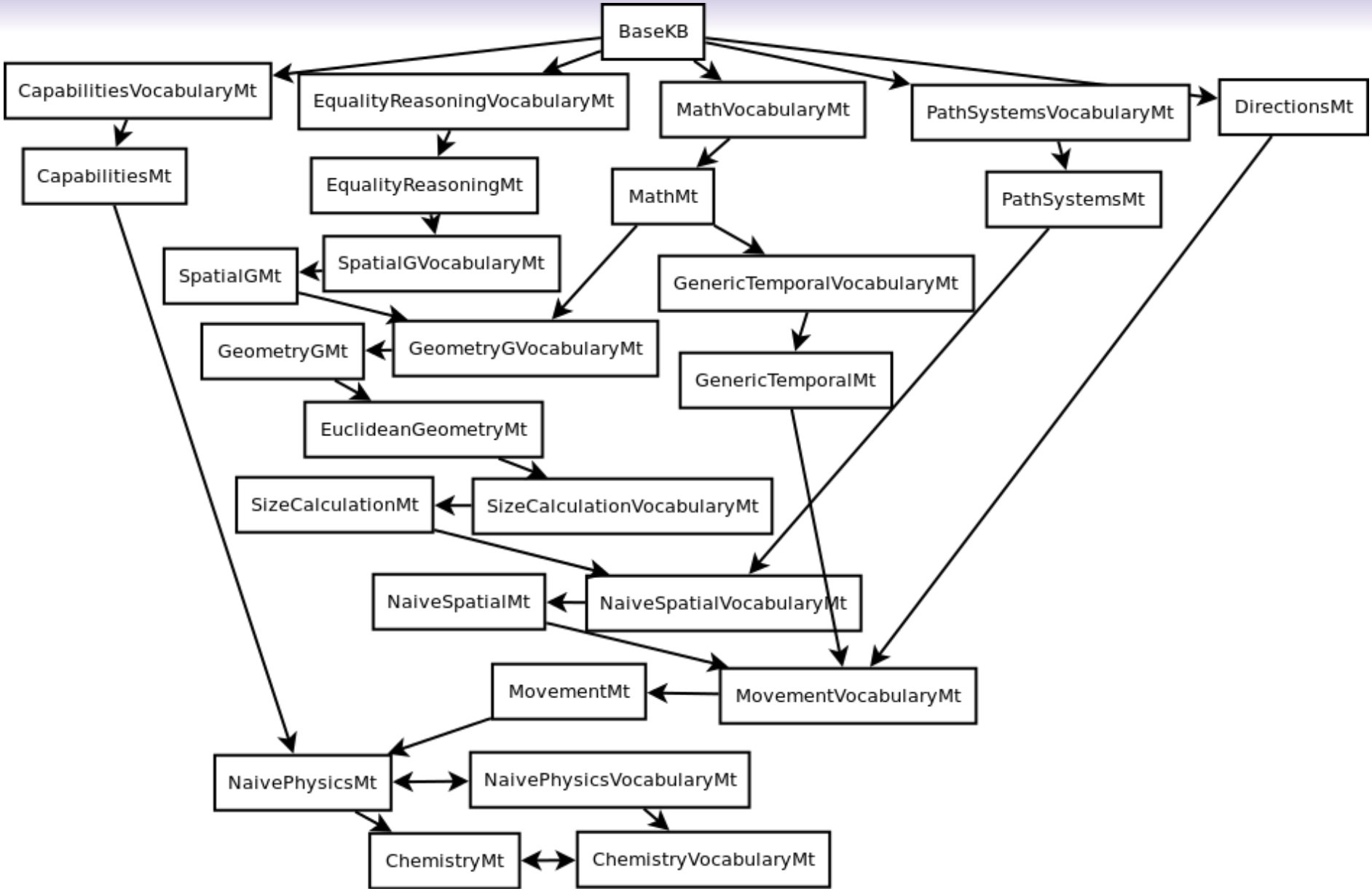
- Collections, which are instances, are called types.
- `COLTYPE`, in predicate `(#$typeGenls COLTYPE COL)` implies type (collection of collections), while `COL` denotes collection.
- `(#$typeGenls COLTYPE COL)` means that every instance `COLTYPE` has `COL` as its generalization.
- For example: `(#$typeGenls #$CreditCardTypeByBrand #$CreditCard)` means that every type of credit card is a subcollection of collections of all credit cards.
- Attention: it does not mean that every collection `CreditCard` is an instance of `CreditCardTypeByBrand`.
- `#$IndividualDenotingFunction` functions returning individuals `(#$resultIsa ?FUNCTION #$Individual)`
- `#$CollectionDenotingFunction` functions returning collections `(#$resultIsa ?FUNCTION #$Collection)`

Microtheories

- Facts are true only in range of microtheories.
- `(#$ist MT FORMLA)` means that the Cyc formula `FORMLA` is true in the microtheory `MT`.
- `(#$genlMt MT-1 MT-2)` means that every assertion which is true in `MT-2` is also true in `MT-1`.
- Relation `#$genlMt` imposes a partial order upon microtheories.
- Most general microtheories:



Microtheories that builds chemistry



Reification

- A non-atomic term (NAT) is a denoting term like a constant. NATs are formed by applying a denotational function to a denoting term, ex. `#$GovernmentFn`
- Some functions return concepts that we want to "reify" and keep in the KB. When a new NAT is created using a reifiable function, that new term is itself reified (kept around separately in the KB) and becomes a Non- Atomic Reified Term, or NART,
- Other functions return concepts that we don't want to store separately in the KB, ex. `#$TimesFn`
- When a new NAT is created using an unrefiable functions, it does not get created as a persistent term in the KB. Unrefiable functions result in Non- Atomic Unreified Terms, such as (#$TimesFn 3 7).