ONTOGENIA

ONTOLOGY GENERATION WITH METACOGNITIVE PROMPTING IN LARGE LANGUAGE MODELS

Miguel Ceriani[1], Anna Sofia Lippolis,[1,2] Andrea Giovanni Nuzzolese,[1,2] Sara Zuppiroli[1,2]
[1] Institute for Cognitive Sciences and Technologies, CNR [2] University of Bologna

MOTIVATION

- **Ontology engineering is complex** and labor-intensive
- There is a need for intelligent automation **LLMs** can help with
- **Metacognitive prompting** improves performance over CoT, asking the LLM to consider its own reasoning process
- This can **reduce errors** and enhance logical consistency

CONTRIBUTIONS

- **A method** to test the effectiveness of Metacognitive Prompting and its application in automated ontology generation
- **A framework** to incorporate LLM-assisted ontology generation into the eXtreme Design methodology
- **A qualitative and quantitative evaluation** of this framework

RESULTS

Ontometrics	Case1 No pattern No MP	Case2 Pattern No MP	Case3 No pattern MP	Case4 Pattern MP	Reference Ontology
Axioms count	49	119	64	118	108
Logical axioms count	26	74	36	76	56
Class count	14	17	14	21	31
Object property count	8	11	8	14	5
Data property count	0	2	3	2	0
Properties count	8	13	11	16	5
Individual count	1	19	0	11	0
DL expressivity	ALCROI	AL(D)	ALC(D)	ALCI(D)	SRI

Axiom type	Case1 No pattern No MP	Case2 Pattern No MP	Case3 No pattern MP	Case4 Pattern MP
owl:Ontology	Yes	Yes	Yes	Yes
owl:Class	Yes	Yes	Yes	Yes
owl:ObjectProperty	Yes	Yes	Yes	Yes
owl:DatatypeProperty	No	No	Yes	Yes
rdfs:domain	Yes	Yes	Yes	Yes
rdfs:range	Yes	Yes	Yes	Yes
rdfs:subClassOf	Yes	Yes	Yes	Yes
rdfs:subPropertyOf	No	No	No	No
owl:disjointWith	No	No	No	No
owl:equivalentClass	Yes	No	Yes	No
owl:Restriction	Yes	No	Yes	Yes
owl:imports	No	Yes	No	Yes

- Classes and properties are successfully identified
- **Richer formalization** with MP, along with a set of different types of axiom types
- **A larger number of axioms** when a pattern is used
- A **higher number of classes and object properties**
- Low cost and time

METHODOLOGY

1. Domain and CQ definition

- Use case of the African Wildlife Ontology
  - → broad domain
  - → covered by Ontology Design Patterns
- Comprehensive: 14 CQs
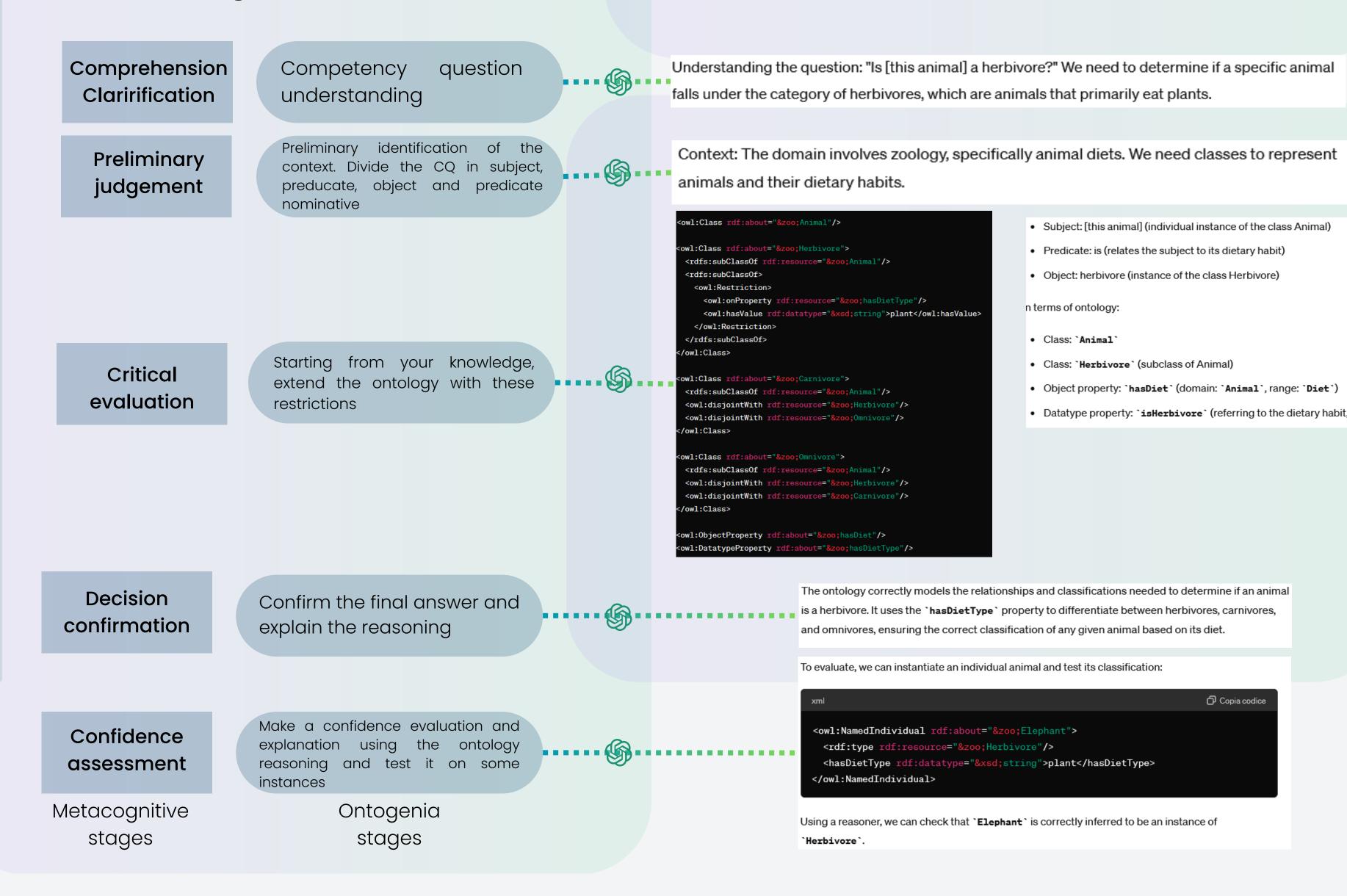
2. ODP selection

- allows structured knowledge adaptation for new scenarios
- From the ODP website
- Selected **8** ODPs to be dynamically inputted to the prompt: *AgentRole, AcquaticResources, Classification, ClimaticZone, Collection Entity, PartOf, Linnaean Taxonomy, SpeciesEat*

3. Procedure and prompt design

Metacognitive stages	Ontogenia stages
Comprehension Clarirification	Competency question understanding
Preliminary judgement	Preliminary identification of the context. Divide the CQ in subject, preducate, object and predicate nominative
Critical evaluation	Starting from your knowledge, extend the ontology with these restrictions
Decision confirmation	Confirm the final answer and explain the reasoning
Confidence assessment	Make a confidence evaluation and explanation using the ontology reasoning and test it on some instances

Understanding the question: "Is [this animal] a herbivore?" We need to determine if a specific animal falls under the category of herbivores, which are animals that primarily eat plants.

Context: The domain involves zoology, specifically animal diets. We need classes to represent animals and their dietary habits.

```
<owl:Class rdf:about="&zoo;Animal"/>
<owl:Class rdf:about="&zoo;Herbivore">
  <rdfs:subClassOf rdf:resource="&zoo;Animal"/>
  <rdfs:subClassOf>
    <owl:Restriction>
      <owl:onProperty rdf:resource="&zoo;hasDietType"/>
      <owl:hasValue rdf:datatype="&xsd;string">plant</owl:hasValue>
    </owl:Restriction>
  </rdfs:subClassOf>
</owl:Class>

<owl:Class rdf:about="&zoo;Carnivore">
  <rdfs:subClassOf rdf:resource="&zoo;Animal"/>
  <owl:disjointWith rdf:resource="&zoo;Herbivore"/>
  <owl:disjointWith rdf:resource="&zoo;Omnivore"/>
</owl:Class>

<owl:Class rdf:about="&zoo;Omnivore">
  <rdfs:subClassOf rdf:resource="&zoo;Animal"/>
  <owl:disjointWith rdf:resource="&zoo;Herbivore"/>
  <owl:disjointWith rdf:resource="&zoo;Carnivore"/>
</owl:Class>

<owl:ObjectProperty rdf:about="&zoo;hasDiet"/>
<owl:DatatypeProperty rdf:about="&zoo;hasDietType"/>
```

- Subject: [this animal] (individual instance of the class Animal)
- Predicate: is (relates the subject to its dietary habit)
- Object: herbivore (instance of the class Herbivore)

In terms of ontology:

- Class: **`Animal`**
- Class: **`Herbivore`** (subclass of Animal)
- Object property: **`hasDiet`** (domain: **`Animal`**, range: **`Diet`**)
- Datatype property: **`isHerbivore`** (referring to the dietary habit,

The ontology correctly models the relationships and classifications needed to determine if an animal is a herbivore. It uses the **`hasDietType`** property to differentiate between herbivores, carnivores, and omnivores, ensuring the correct classification of any given animal based on its diet.

To evaluate, we can instantiate an individual animal and test its classification:

```
<owl:NamedIndividual rdf:about="&zoo;Elephant">
  <rdf:type rdf:resource="&zoo;Herbivore"/>
  <hasDietType rdf:datatype="&xsd;string">plant</hasDietType>
</owl:NamedIndividual>
```

Using a reasoner, we can check that **`Elephant`** is correctly inferred to be an instance of **`Herbivore`**.

4. Evaluation measures definition

- **qualitative** ontology engineer evaluation
- **quantitative**: Ontometrics and OOPS!Ontology Pitfall Scanner

5. Testing

- **GPT-4 Turbo API** as backbone model
- Temperature 0, other parameters as default
- **Four iterations** (generic with CQs, generic with CQs and ODPs, CQs and MP, CQs with MP and ODPs) with a division of CQs in batches

```
prompt = (
    f"Read the following instructions: '{procedure}'. Basing on the procedure, and following the previous
    output: '{previous_output}',  design an ontology that comprehensively answers the following competency
    questions: '{CQs}', using the following ontology design patterns: {combined_patterns}. Do not repeat
    classes, object properties, data properties, restrictions, etc. if they have been addressed in the previous
    output. Remember that crucial elements for a good ontology design are '{ontology_elements}'. When you're
    done send me only the whole ontology you've designed in OWL format."
)
```

LIMITATIONS

1. Absence of property hierarchical relationships
2. Lack of annotations and inverse relationships
3. With patterns, untyped class and different naming conventions are found
4. Data properties less common with patterns
5. LLMS: generated ontologies exhibit numerous intrinsic an domain-related issues; pairing of classes and properties problems
6. LLMs' context limit