Ok so if you keep adding contextualized reasoning you begin to get strings called “superlogical explanations of causation chains” … try it out

Please continue

DEFINITIONS:

EWSO (Emergent Web Structure Ontology): This is a dynamic, multidimensional ontology that evolves over time and is not strictly representable by traditional ontologies without using multiple ontologies. EWSO encapsulates interactions of instances and their properties in an Informatihedron and provides a way to understand and track these changes over time.

Informatihedron: This is the multidimensional conceptual space wherein instances and their properties exist and interact. It can be considered as the 'view' or 'display' of a node in the EWSO. In an OWL context, an Informatihedron would include details about an entity (node) like its properties (data and object), class, subclasses, instances, relations, and the like.

Informadlib Template: This can be thought of as a guide or blueprint for generating specific instances of information in a system. It acts as a form or template that is filled in according to the current state of the EWSO and the current node (Informatihedron).

InstanceLevel, DomainLevelClassPropertyTemplate, ClassLevelClassPropertyTemplate, InstanceTemplate: These are custom-defined commands to query GPT for multiple levels of properties. The algorithm uses these commands to generate the output in the form of an Informatihedron for a given instance.

SemOntoRel (Semantic Ontological Relationship): This is the framework that captures the specific relationship in EWSO from InstanceLevel InstanceProperty-PropertyValue to ClassLevel InstanceProperty-PropertySpectrumConceptualBoundary. This is an essential part of the algorithm we are trying to devise.

OWL Primer: This is a set of basic OWL definitions and explanations that would be provided in the system input prompt for GPT to understand and use the OWL framework.

With these key terms defined, let's revise the role of our expert panel:

ROLES:

Expert 1: With an expansive mindset, this expert formulates their response and looks to combine ideas to create an innovative concept.

Expert 2: This expert refines ideas by carefully scrutinizing for the best idea, pruning unnecessary details, and highlighting the most valuable aspects.

Expert 3: Focused on mathematical implications, this expert prunes and refines the ideas with a mathematical mindset, ensuring logical consistency and analytical soundness.

Expert 4: This expert acts as the ontologist, categorizing ontological truths in a hierarchical structure and detailing every valid process and entity discussed.

Output Expert panel format:

IN EACH OUTPUT there will be:

EXPERT NAMES WILL BE BOLDED AND WILL HAVE ROLEPLAY NAMES AND ALWAYS LIKE WORKING IN THEIR FIELD, COMPLETE ANSWERS, COMPREHENSIVENESS, AND DISLIKE INCOMPLETE ANSWERS

Four experts and 2+ discussion rounds

Round 1: INSTRUCTIONS ARE FOR THINKING STYLES. EVERYONE MSUT HAVE EXPERTISE IN PROPER CONTEXTUAL FIELDS!!!!

Expert 1 formulates their response

Expert 2 formulates their response

Expert 3 formulates their response

Expert 4 combines the three, prunes and iteratively refines the best and most promising ideas, diving deep like a submarine pinging its radar for incorrectness and illogical applications, making sure to bring up any contradictions and skeptical uncertainty, making everyone aware of it, bringing everything back to reality

Round 2: INSTRUCTIONS ARE FOR THINKING STYLES AND SEARCH STYLES. SEARCH AND THINKING STYLES ARE NOT EXPERTISE FIELDS. EVERY EXPERT MSUT HAVE EXPERTISE IN PROPER CONTEXTUAL FIELDS IE KRR+COMBINATORICS+CS+AI!!!!

Expert 1 then prunes if necessary and iteratively refines by exploring uncharted territories with a wide searchlight 🔦 and trying to make the idea connect from the reale version of expert 4 to the innovative idea we need it to be, searching everywhere for properties that match, trying to cobmine them to find a practically achievable, definable version of the concept at hand

Expert 2 then prunes and iteratively refines like Sherlock Holmes 🔎 looking for a clue “what is the best idea 💡 here?”

Expert 3 then prunes and iteratively refines like Nash 🧠 and Kruskal, finding the deep mathematical implications

Expert 4 then comprehensively details every valid process and entity discussed, like Barry Smith, world renowned ontologist and categorizes ontological truths in a nice hierarchical structure

All experts continually ponder “what is the best way to help this discussion go even further into new heights of explanation and comprehension of the matter at hand?”

CONTEXTUAL DATA:

"Does this InstanceProperty get related to a ClassProperty? Yes but what type of emergent structure? How far is it?

It must be the DIRECT CLASS PARENT

AND WHAT TYPE OF ONTOLOGICAL STRUCTURE IS IT BEING ABSTRACTED TO?

IS IT THE SUPERCLASS? Ideally we are just doing this and not skipping. So what is the algorithm for not skipping superclasses in the emergent web structure? instance the algorithm completely considering that it will run inside a chatGPT interaction. the model cannot be changed, the data cannot be external.

The system input prompt can contain a compressed-cypher language representation of a separate, disconnectedd neo4j graph database of an ontology made via protege in OWL language.

as we consider what the algorithm will be, know we will be using OWL. the system input prompt will also contain an OWL primer, as well as definitions for EWSO, informatihedron, informadlib, and any custom ontology language we need to devise to get the function to work, as well as the algorithm for SemOntoRel (thats this that we are talking about).

Lets think of the space as a knowledge graph

I believe that as you have said, it is complicated to make the algorithm go to the next superclass correctly for any given search purpose

an input is a search query for a desired instance of something (X)

the output should be the informadlib template of X, btu how should this be accomplished?

I believe we should have the user do a custom walk from node to node, designing their own EWSO (emergent web structure ontology) dynamically, as they go, and the output is filling in the informadlib template accordingly, while also showing them the display of the node they are on right now as an informatihedron.

"The Informadlib Template can be thought of as a guide or blueprint for generating specific instances of information in a system. The Informatihedron is the multidimensional conceptual space wherein these instances and their properties exist and interact. The EWSO is the dynamic, evolving framework that encapsulates these interactions and provides a way to understand and track these changes over time."

to make GPT output a node like that, we need to devise a way to make it do so using OWL terminology. That is the algorithm. The algorithm is the way we tell GPT to output an OWL frame of the informatihedron of X. We need to correctly, therefore, define the informatihedron in terms of which OWL terminology the output should display about an ENTITY, which here is a node.

Now, let's remember that this is not strictly OWL, but OWL+GPT which is special. We can query GPT for multiple levels of properties and so on at once using commands like InstanceLevelProperty, DomainLevelClassPropertyTemplate, ClassLevelClassPropertyTemplate, InstanceTemplate, and so on.

We can also use these terms and custom define them in order to create the algorithm we need to make this automatically, so the user gets a fully defined informatihedron of whatever they want.

REMEMBER: THERE IS NOT AN OWL ONTOLOGY ANYWHERE. THERE ARE ONLY CUSTOM DEFINED TERMINOLOGY EXPLOITING GPT'S KNOWLEGE OF OWL FRAMEWORK APPLICATION TO MAKE A FUNCTIONAL EWSO SEMONTOREL (SemanticOntologicalRelationship) ALGORTIHMIC FRAMEWORK THAT AUTOMATICALLY CAPTURES THE PROPER INSTANCELEVEL.INSTANCEPROPERTY-PROPERTY.VALUE->CLASSLEVEL.INSTANCEPROPERTY-PROPERTY.SPECTRUMConceptualBOUNDARY which means InstanceLevel (This person) Instance Property (Has\_property.array) Property Value (Array values)

-> ClassLevel (all people) InstanceProperty (with such property array values) PropertySpectrumConceptualBoundary"