Knowledge Representation and Reasoning with Supply Chain Contracts

Abstract

Knowledge Representation and Reasoning (KRR) is a subfield of AI that deals with the way an agent's knowledge is collected, interpreted, and used to infer more knowledge. Throughout this project, we will seek to implement a foundational understanding of KRR as it pertains to logical agents with an application in supply chain. A logical agent, like other agents, has sensors and actuators to collect information about the environment and perform actions in it. However, unlike other agents they utilize a knowledge base, which is the agent's knowledge bank represented in a formal form, i.e. logical language. Different types of logical languages have varying scope and expressive power. They also support an inference engine which works on top of the knowledge base to check the consistency of the collected knowledge with the knowledge base or extract new ones from the knowledge base according to new percepts. (Russel & Norvig, 2021). There are many practical applications of logical agents related to manufacturing (sensor systems, automated processes) (Metzger & Polakow, 2011), transportation (automated vehicles, smart traffic systems) (Chen & Cheng, 2010), as well as supply chain optimizations (Flynn, et al., 2023) (Li & Chan, 2013)(Dominguez & Cannella, 2020). The interest for this research is supply chain management, and the difficulty for supply chains to adapt to rapid change in demand or circumstances. This research will focus on supply chains and how logical agents can be used to measure and improve their resilience.

Goals

The goal of this project is to perform exploratory research into the ways supply chains can be modeled using knowledge-based agents and identify a process of implementation using declarative language and Answer Set Programming.

Methodology

Through this project I will learn how information about a real-world application can be represented in a knowledge base, and how to design an agent to reason over it. The research will involve the introduction of Answer Set Programming (ASP), which is a declarative programming paradigm based on logic programming (Falkner et al. 2018) and will be used throughout the project for implementation of the supply chain knowledge based. A system of supply chain contracts can be viewed as a Cyber Physical System (CPS) and represented based on the National Institute of Standard and Technology (NIST) Framework. According to this framework, I can include stakeholders' concerns in the agent's knowledge base (Flynn, et al., 2023). We are expecting that the knowledge base can answer specific questions regarding contracts feasibility and offer ways to improve the contracts to enhance resilience in the supply chain.

Timeline

The following is a timeline for the completion of the research during the semester, broken into two-week periods:

1. Logic Programming and ASP Intro
   1.1. (week of 1/28) Familiarize with fundamentals of declarative programming. OBJ Introduce Clingo (ASP). Detailed review of Propositional Logic.
   1.2. (week of 2/4) ASP syntax, differences from Prolog, simple programs
2. Learn to Develop with ASP
   2.1. (week of 2/11) Study introductory ASP examples
   2.2. (week of 2/18) Simplify other implementations and understand differences
3. Form the Basis for Final deliverables
   3.1. (week of 2/25) Describe problem with natural language, Define initial constraints
   3.2. (week of 3/3) Research application of similar problems to descriptive language
4. Representation
   4.1. (week of 3/10) Write implementation
   4.2. (week of 3/17) Write implementation
5. Completion and Testing
   5.1. (week of 3/24) Complete programming and begin checking
   5.2. (week of 3/31) Complete checking and evaluate own performance and progress
6. Project Finalization
   6.1. (week of 4/7) Confirm finalization of programming deliverables
   6.2. (week of 4/14) Collect materials for final presentation and report
7.
   7.1. (week of 4/21) Draft report and presentation
   7.2. (week of 4/28) Finalize and submit report and presentation

*Timeline subject to change throughout progression

Progress Report: Week of 1/21

On Monday (1/22) the project topic was chosen with the help of Dr. A. Izadi. The project's main objective is to familiarize myself with introductory concepts of KRR and to learn and practice the tools necessary to apply these concepts to logical agents reasoning about supply chain contracts. This week, the topic was introduced, and literature was collected regarding some of the necessary concepts including answer set programming, the field of KRR, and an instance of application to supply chain contracts (Flynn et al. 2023).

Last Week Start

At the beginning of the week, I acquired my project topic with the help of Dr. Izadi who advised me that my project topic could help facilitate further research in the same area in the future. My primary interest in computer science is the ways that the real world can be modeled on computers to solve problems with computation, and the answers can lead to changes in the real world that you can see. KRR is a field that deals specifically with the way information from an environment can be formalized in a computer system that makes it possible to solve problems.

Last Week Goal

The goal this week was to collect literature for review and to learn the process of reading a research paper in a useful way. The paper on supply chain contracts between agents (Flynn et al. 2023) was the first document focused on because of the direct relevance to the topic of the project, and the methodology proposes a path toward the same end goal.

Obstacles

The main challenge faced at the start of this project has certainly been the learning curve associated with processing research papers and learning resources. KRR is a widely applicable and researched field of study, and logical agents can become very complex. To learn these concepts, it is important to be able to consume information efficiently and effectively from research articles and conference papers.

I received the criticism that I tend to be too obsessive over details at my first glance of a problem or concept. I like to understand every step as I read them the first time, which works well with many of the easier topics I've encountered. Regarding more complex topics, it tends to inhibit me. I received some advice from peers as well as Dr. Izadi and realized that higher level concepts require an analysis at different levels and must be understood at a more abstract level before understanding the parts. A recent Medium post by Jason Corso, a professor at the University of Michigan, briefly describes method to summarize papers by their problem, approach, claim, evaluation of the claim, and substantiation (PACES method) for extracting useful summary points from conference papers to utilize time more effectively. This proved useful in processing the paper of Flynn et al. as the details of the methodologies could not become clear until I understood what each step was meant to accomplish.

Results

I was able to process a significant amount of information from my initial literature gathering. As a deliverable, I was able to provide a step-by-step summary of the Flynn et al. paper, which will allow me to more competently understand the motivations and methodologies used on this problem.

Plans for This Week (1/28)

This week I have planned to continue my literature review, complete several exercises regarding Logical Agents from the AI: a modern approach (AIMA) text, and deliver my first program using Clingo, an answer set programming tool.

The feasibility of this weekly goal is very high as it is easily measurable. My plan is to become more familiar with the workings of logical agents and collect and demonstrate introductory knowledge of declarative programming to be built on later.

Progress Report: Week of 1/28

The goal for this week was to complete exercises from AIMA (Russel et al. 2022) to round out the foundational understanding of logical agents, as they are the basis for the planned agent-contract system. This included an overview of propositional logic and reasoning. Additionally, the first Clingo program for Answer Set Programming was also introduced.

Challenges

The challenges faced this week included a lengthy refresher on proofs and the seemingly complex syntax of Clingo for ASP. The proof by resolution was a difficult concept to learn quickly because it involves combining propositional clauses in conjunctive normal form (CNF) and applying the inference rule of resolution with a negation of the statement to prove until a contradiction is found. It was difficult to find a systematic way of determining which statements to use to produce the next in the sequence. Additionally, ASP is a declarative approach to programming as opposed to the traditional descriptive (imperative). Naturally, this caused problems while trying to jump into programming.

A systematic way to solve the resolution proofs was found by finding a chain that eliminated multiple literals from a specific clause. The inference rule of resolution allows you to take “(a **or** b) **and** (**not** b **or** c)” and resolve it to ”(a **or** c)”. If you can chain these together until you acquire “(a **or** a)” for example, you can reduce to “a” which means that “a” must evaluate to true given the premises. This allows you to assume true for all “a” in the other clauses. If you can reduce to the empty clause, or false, you have reached the contradiction, otherwise the premises involving your negation is satisfiable and you have not proven the original statement.

As the propositional logic review progressed, the syntax of ASP seemed clearer as well. Lines in Clingo can be understood with elements of propositional logic. Programs are designed by outlining inferences, and constraints of the program seem to be written using clauses that imply a false outcome. If a proposed world, or answer set, satisfies these clauses then there is a contradiction, and the set is deemed not an answer set.

Results

A thorough introduction to propositional logic and logical agents was performed which included identifying statements as valid, unsatisfiable, or neither. This required using inference rules to reduce the statements to CNF and evaluating them by hand. Proof by resolution, as described in the previous section, was also reviewed, and gave some interesting insights into how ASP seems to evaluate incorrect sets. My first Clingo deliverable was a very small ASP program which provided an answer set to an example constraint satisfaction problem (CSP) from the AIMA textbook. Briefly, it involves scheduling 3 robots in 2 hours to complete 5 one-hour tasks. This was not an official example for propositional logic, but an example solved by hand in a previous chapter. I decided to produce Clingo code that outlined the constraints of the problem using declarative language, and the answer set found by hand previously was produced from the program. This is simply an introductory example, and there is much more to ASP and Clingo that will be useful for the project going forward. These two topics coincided quite well, as problem

solving within topic 1 led to new understanding of topic 2. The review of these topics resulted in favorable progress and coincided with the timeline on the proposal.

Next Week

The following week will be focused on exploring additional chapters in the AIMA book relating to Automated Planning and Multi-Agent Decision Making and resolving some of the questions from last week's review of the introduction of logical agents and propositional logic. If additional time is available, a more comprehensive overview of clingo will be performed and will accompany a second ASP demonstration.

Progress Report: Week of 2/4

For the previous week, the goal was to complete review multiagent systems and automated planning to better relate it to the existing methodologies for this problem. Additionally, upon review of previous work, the goal of the research project has shifted to determining a method by which agents or contracts can be evaluated by the restrictiveness of their sets of constraints. This work will be significant as the main goal of analyzing supply chain resilience is to prepare it for drastic changes. Highly restricted agents or constraining contracts can need the most attention in the event of a massive change to the supply chain, and it is best to be aware of these nodes ahead of time.

Challenges

The challenge this week was primarily identifying a more specific goal for the project, and developing a new timeline that ensures the goal is reached by specific dates.

Previous methodologies for analyzing supply chain resilience using logical agents focus on computing series of states and actions of two agents that share a contract, and then questions are proposed to evaluate the resilience of that series, or *trajectory*. These methods suffer from a lack of attention to sudden changes such as COVID-19 with large changes to the supply chain. We have chosen to expand the methodology with the focuses on explicit expression of agent-contract system led to the introduction of concern satisfiability clauses. We seek to add a new query to this system that will be able to quantify to some degree the risk associated with an agent or contract, or the propensity for breaking contracts in the event of a change in the supply chain. By identifying these problem agents/contracts, we will be able to consider these areas of the supply chain in the event of a sudden change.

Developing a new timeline for this project that better outlines the goals was achieved by planning completion of research and write-up of a paper describing the research by March 30th. April will be used for final write-up of the research for the senior project course. This allows for over a month of implementation, a one month of write-up. The expected progress is congruent with the constraints of the new timeline.

Results

This week we continued to build upon our understanding of the agent contract system by researching multiagent systems and planning. Muli-agent systems are introduced in the chapter 18 of Russell & Norvig Textbook in the context of game theory.

Essentially, in this context the multiagent supply chain system can be viewed as a coalition of agents that are working to produce a specified output. To do this we specify contracts as an outline of the constraints between the agents. The textbook also explains that multiagent systems formulate sets of actions that coincide with one another, and sometimes agents will take no action to wait for other actions to be done. In our context, we specify a *trajectory* as a possible series of states and actions of two agents over time. These trajectories are what the previous queries (Q1, Q2, Q3 for feasibility, clause, and concern satisfaction) analyze to evaluate the resilience of the chain. Sections 18.4.1 (Contract net protocol) and 18.4.2 (Scarce resource management with auctions) of the textbook outline some interesting concepts related to our research as well because they deal with a type of contract implementation and resource management, both important topics in the supply chain. The contract net protocol is a concept by which the multiagent system can construct and bid on contracts to perhaps form the type of system we are

working on, and the scarce resource auctions provide an interesting way to resolve resource issues in an economic way. The auctions are particularly interesting as they would inherently provide agents with a new avenue through which to resolve resource issues if it fits within the cost constraints.

The readings this week provided a lot of context and useful information regarding the construction of a multiagent system, as well as constructing and utilizing an action domain to describe the needed actions for each of those agents which are necessary for automated planning. The automated planning provides some insights into planning and some algorithms to explore possible trajectories in supply chain.

Upcoming Work

The bulk of the literature review portion of the project is planned for the next several weeks, and the context of the problem will be sufficiently outlined. To continue the literature review, I will be working to completely review the chapters regarding logical agents, multiagent systems, and automated planning in the Russel & Norvig text, acquiring 3 survey papers to gain more perspective on where this work fits into existing work in the field, and researching several research papers that explore other facets of the problem of supply chain contracts. In the next several weeks, I will also be working to implement a logical agent capable of navigating the Wumpus world, which is outlined in the Russel & Norvig text and is an environment with specific percepts an agent is able to use to reason about its surroundings. More details regarding the specifics of this common example in AI will be covered in my future updates.

Progress Report: Week of 2/11

The current task is to solidify many of the concepts from the reviewed chapters on Logical Agents, Multiagent systems, and Automated planning, as well as learn more about Clingo to be able to implement an agent capable of navigating the example, Wumpus World.

Challenges

The challenge this week was identifying the holes in my understanding of Automated Planning. Automated Planning for logical agents includes determining actions and using them to construct a factored representation of the problem. The disconnection was at how exactly the problem is formed with all the smaller pieces.

I learned that to construct a representation of the problem, you must first design something called an action schema. An action schema is made by outlining *actions* using *preconditions* and *results.* For example, the action of drinking water could have the precondition of water available and thirsty. With an action schema, an agent can eventually decide which actions to take based on the availability of the preconditions in the current state, and whether the goal can be achieved using the results of the action. An action schema alone, there is no problem that can be solved, there are only actions to take. The problem also requires an initial state, and a goal state(s). These states are described as ground atomic fluents. *Ground* means that all the variables of the statements have assigned values, as a state is meant to describe the real world in some way. If I described my position on the Earth as “x” and “y”, it would give you no information as to where I am. *Atomic* means that the statements cannot be logically reduced any further, this is important because it makes it faster to calculate if preconditions are met and what changes are made to the state based on results. *Fluents* are essentially variables that change over time that are relevant to the problem. By breaking down the terms in this way, I was able to understand the formalization of a problem for use with agents.

Results

This additional research into Automated Planning and problem representation will influence the work to come in that the structure for assembling state-action trajectories between multiple agents will use a similar approach. An action language is used to describe what actions are possible and the ground fluents will be used to describe states including initial and goals. Getting to understand how this structure works will assist in the Answer Set Programming implementation because these aspects will need to be defined in Clingo, the ground statements will be used to define the initial and goal states and rules and actions will also be defined with a similar structure.

Upcoming Work

We will continue to expand our knowledge on logical agents and multiagent planning. The goal is to understand how the agent-contract system works to be capable of implementing a system that can assess risk of agents breaking a contract in the future. We will review several survey papers and increase our ability to define problems for agents in Clingo.

Progress Report: Week of 2/18

This week the task was to develop a practical implementation of a logical agent capable of traversing the Wumpus world. This task is significant because it is a steppingstone to representing more complicated logical systems in Clingo in the following weeks.

Wumpus Summary

The Wumpus world is a common example used to teach logical agent reasoning. There is a 4x4 grid, with an agent that begins facing east on the bottom leftmost square. There are multiple pits, one Wumpus, and one piece of gold. Squares adjacent to pits in each cardinal direction have a breeze. Squares adjacent to the Wumpus in each cardinal direction have a stench. The square that contains the gold has glitter. The agent can turn left and right, move forward, shoot, and climb out of the cave. He can only climb out of the initial (1,1) square with the gold, and only has one arrow. If he shoots the Wumpus with the arrow, the Wumpus dies with a loud scream. The scream, stench, breeze, and glitter are all percepts that the agent can use to form a plan. As the agent gains more information through its precepts, it will try to retrieve the gold and climb out safely.

Challenges

This week we faced the challenge of formalizing this problem in a logical language using Clingo. This was difficult as it was the first step we took from learning the concepts of logical agents and automated planning to a practical implementation that is closer to what real world implementations are like.

To formalize the problem, I was able to think back to my research on automated planning from last week. There, we described the resilience in supply chain problem using Planning Domain Description Language (PDDL) by describing an action schema as a group of actions with preconditions, results, and optionally concurrency rules. Considering the action schema in this way makes it easier to think of what makes an action available and assess its viability by looking at its preconditions and results. For instance, the “climb” action in Wumpus world has the preconditions that the agent must have obtained the gold and be in the initial square (1,1). After designing the action schema, ground atomic fluents (explained in the challenges of the previous week’s report) could be constructed as a group of variables that describe the state of the system. Some of these include whether the Wumpus has been killed, and whether the player has the gold. Other than these, the nature of the environment is mostly static, as the Wumpus cannot move. However, as the agent makes moves, he receives percepts. As it is a partially observable environment, the agent never has full knowledge of his surroundings unless he has identified all the squares. To utilize percepts, a percept schema was used to observe percepts and make inferences about the environment using them. For example, if a breeze is perceived, the agent will know that the square it was just on does not have a pit and can mark the other 3 squares around the breeze as potential pits. This combined with a way to determine safe squares that have not been traversed is essential to getting around the environment.

Results

The implementation of the Wumpus world solidified a lot of my knowledge on single agent planning and gave me insight into the way I can apply the structure of initial and goal states with an action schema to describe problems to be solvable. Taking this practical knowledge with me into the next steps will help me formalize the supply chain contracts in the coming weeks.

Upcoming Work

The upcoming weeks will include implementation of the actual multiagent system of agents and contracts. From there, we will be able to identify high risk agents within the system. Additionally, more literature review will be summarized to formulate some more context for this research within the field.

Progress Report: Week of 2/25

This week the task was to complete the logic program USING Clingo that can solve the Wumpus world example and identify methods for assessing risk in an agent-contract system modeling a supply chain. We were able to finish the Wumpus example and collect some documents on current solutions to analyzing risk of the supply chain.

Challenges

This week problem was learning how we can select rules and aggregate them to work for optimization in ASP. Clingo syntax resembles standard propositional logic, with some key differences that help the designer create useful programs. Some of these are choice rules and aggregations. Another challenge was learning to debug a logic program.

The primary function of Answer Set Programming is to generate answer sets of grounded statements that satisfy a logic program. You may outline statements by specifying which statements follow from other statements, and some grounded statements, and an ASP solver will be able to generate sets of grounded statements. It will then check these sets using your constraints and output sets that satisfy the constraints. This is relevant to designing logical agents because you may outline your agent's initial knowledge in logical language, and further knowledge can be inferred by the agent through the solver. Choice rules are a way that you may specify that of the *possible* grounded versions of a logical statement, the solver is meant to choose a specific amount outlined by the designer. In many scheduling tasks, including contract realization and this Wumpus example, the designer is concerned with finding a set of sequential actions that satisfy a set of constraints. The choice rule is how the designer is able to tell the solver that one move is to be selected for each time step. Of all the possible moves, the solver now knows to choose one for each timestep and will return an answer set of that form that satisfies the constraints if possible. Without this understanding, my agent would try to select many moves per time step and incorrectly report a success by completing all of the required actions in one time step.

For debugging, this was a very interesting task. It is not as straightforward to debug logic programs, as it is less intuitive to set things like flags and print statements. At the bottom of the ASP program, you may have lines that look like #show action/2 . As we stated before, the solver will generate many grounded statements that satisfy the constraints of the problem. Think of this as a massive pool of assignments that you must sort through to understand the results of the program. This #show do/2 statement specifies that of all of the statements generated, print out all of the statements labeled "do" that contain 2 predicates. For this problem, we have the form do(M: operation, T: time step). So when we print, it will look like this:

```
do(forward,1) do(turnleft,2) do(forward,3) do(forward,4) do(turnleft,5) do(grab,6) do(turnleft,7) do(forward,8) do(turnr
ight,9) do(forward,10) do(turnleft,11) do(forward,12) do(climb,13)
Optimization: 14
OPTIMUM FOUND

Models       : 6
  Optimum    : yes
Optimization : 14
Calls        : 1
Time         : 5.305s (Solving: 5.10s 1st Model: 3.22s Unsat: 0.02s)
CPU Time     : 4.828s
```

Figure 1.1: Notice how the answers are out of order, but can be understood by the inclusion of the time variable in the do(M,T) statements.

Results

This task took up a fair amount of the remaining time in the project, but helped solidify many ASP principles that will be used while formulating my approach to logical agent reasoning about high-risk nodes in the network.

Upcoming Work

The upcoming weeks will include formulation of the risk assessment in the multiagent system of agents and contracts. Also, survey papers and related work will be reviewed to help with the initial poster presentation.

Progress Report: Weeks of 3/3 and 3/10

The current task was to begin the representation of knowledge for the agents in the agent-contract system. Previously we have learned about formalizing the wording of contracts and constructing problems using initial and goal states with an action domain. This week we were able to construct an action domain for a lumber yard.

Challenges

The motivating example used as a basis for this representation can be found in the Flynn et al. paper. This example includes details for how we can formalize and represent an agent's action domain as a logic program. Clingo is being used to construct the logic program due to its readability and ease of use in describing logical statements consisting of arithmetic, implications, and predicates as well as other logical functions. The challenge here was learning how to explain the domain of the lumber yard using a collection of laws presented in the action language B (Fig. 1.2)

$$\begin{aligned} \textit{Executability condition}: &\quad \textbf{executable } a \textbf{ if } \varphi \\ \textit{Dynamic law}: &\quad a \textbf{ causes } \psi \textbf{ if } \varphi \\ \textit{Static Causal Law}: &\quad \psi \textbf{ if } \varphi \end{aligned}$$

This set of laws combines with a disjoint set of actions and fluents is meant to be able to accurately encode required actions for the agent to be able to fulfill the requirements to be outlined in their contracts.

Results

For this example, we consider two actions needed by the Lumber Yard: *produce* and *deliver.* These two actions are sufficient for demonstrating the representation of actions for an agent as they fulfill abstract duties of the agent, and the process for encoding them can be repeated to explain less abstract actions such as *package* and *ship* if needed for an implementation. We were able to translate the natural language representation of the laws of the action language (Fig 1.2) to Clingo code with some rearrangement of statements and previous knowledge of Clingo syntax.

Upcoming Work

This process will be repeated in the coming days for a Home Builder agent. These two agents will make up a useful example in explaining the process of encoding these logical agents and encoding their action domains. Following that, we will be able to generate and formalize a contract that contains clauses for the agents to fulfill in their action sequences.

Progress Report: Weeks of 3/17

The current task was to begin the representation of knowledge for the agents in the agent-contract system. Previously we were able to construct an action domain for a lumber yard. This week we worked on completing the representation portion of the system by completing the same logic program for the home builder and formalizing the CPS ontology in the form of an ASP program.

Additionally, as the first major section of this project concludes in the representation of the system in logic programs, we have taken some time to review our work so far and take a closer look at the overall story of our project

Challenges

The major challenge this week was taking a step back and analyzing our work. Comparing an approach like this to an ML approach is likely a very common question that could be asked, among other interesting questions that require consideration.

For this question, we need to consider the differences between ML concepts and logic-based approaches. For machine learning approaches, large amounts of data are needed, and would be used to train models to make statistical based conclusions that could help improve resilience. This has seen a decent amount of success, specifically in risk assessment and analysis of resource routing in multi-level supply chain delivery routes. Our approach, however, is based upon logical reasoning. We wish to represent knowledge for the agents in a way that allows for inference and reasoning about the system to gain precise and specific insights rather than statistical ones. We wish to have the system reason about problems with the supply chain's resilience and be able to provide information to mitigate them. This type of work is very relevant and useful to supply chain management for this reason but could also benefit from some ML and specifically natural language processing in the form of automated logic extraction from natural language contract specification documents.

Results

We have been able to explain the process of this section of the work more carefully and can convey the answers to certain popular questions that could be asked of a project like this one. Additionally, the logic programs for the CPS and the second agent are completed.

Upcoming Work

Soon, the contract specification will be generated and formalized, and we will have the representation in the form of our logic programs. We hope in the coming month to be able to reason about these contracts using different queries regarding satisfaction of concerns and requirements, and contract feasibility as a whole

Progress Report: Weeks of 3/24

This week, we worked on reading about ways to reason about the knowledge represented in the system. Implementation will start at the end of this week.

Challenges

The major challenge this week was figuring out how to use the trajectories generated by the agents to make reasoning answers to queries. This is how we make the agents useful by giving them an avenue to make use of their knowledge base.

Using trajectories as a series of states and actions, we can compare them with the trajectories of other agents to see if they are compatible. For example, a trajectory where the lumber yard produces and delivers the lumber in the first 4 weeks, and a trajectory where the homebuilder receives the lumber in the 3rd week and pays, can both satisfy the contract and thus be viable trajectories, but they are not compatible. The time the lumber is delivered is after the homebuilder plans to receive it. The concept of compatible trajectories will help us reason about interactions in the whole system with respect to agent actions. Additionally, the state of the environment must be consistent between all states in both trajectories. This involves comparing fluents that make up the states in both trajectories and making sure that they are the same.

Results

This week was taken to partially recharge and prepare for the remaining weeks. We have a plan now for implementation of reasoning capabilities by focusing on the generation and comparison of trajectories. We then will be able to analyze them and have the agent answer these queries.

Upcoming Work

The logic programs will soon be updated to repond to important user queries such as clause and concern satisfaction. This type of reasoning is possible because of the logical representation that we made in ASP out of the formalizations before.

Progress Report: Week of 3/31

The current task was to begin the representation of knowledge for the agents in the agent-contract system. Previously we were able to construct an action domain for a lumber yard. This week we worked on completing the representation portion of the system by completing the same logic program for the home builder and formalizing the CPS ontology in the form of an ASP program.

Additionally, as the first major section of this project concludes in the representation of the system in logic programs, we have taken some time to review our work so far and take a closer look at the overall story of our project

Challenges

The major challenge this week was identifying how to reason about the knowledge represented in the knowledge base. To do this, we consider the concept of compatible trajectories from earlier. Our reasoning will be based around using our knowledge about the system to generate trajectories and answer queries about them. The goal is to understand the existing queries in order to create additional useful ones later on.

Compatible trajectories, to recap, are trajectories among multiple agents in the system that have coinciding fluents making up the states for each time step. If these fluents are the same, then the trajectories are compatible because they have consistent states and thus can explain the same real-world scenario. We can reason about these trajectories in many ways. The ones we are learning about right now refer to whether these trajectories satisfy all clauses of the contract between the agents, and whether they satisfy all concerns via the CPS. We can also find the time step at which a clause or concern is broken and construct new trajectories from that timestep to gain insights for mitigation purposes.

Results

We are ready to work on the clingo implementation, where we can use the input language to form ASP statements that lead to reasoning over the system.

Upcoming Work

The next week will include working on formulating the clingo program to reason over the previous logic programs. We wish to be able to construct and have the system respond to these queries to support future queries for things like risk assessment. We will also be working on drafting the report to explain the process we have undergone to get to this point.

Progress Report: Week of 4/7

The current task was to learn how to implement a way to generate compatible trajectories to enable agent reasoning about contracts. By generating trajectories for individual agents that are compatible, we can look at them and evaluate whether or not each individual clause of the contract holds, and if any clauses are broken. Judging this, we are also able to evaluate which concerns of each agent are not met by the current trajectories.

Challenges

The major challenge this week was to formulate a way to generate separate trajectories, evaluate if they are compatible, and then check to see if they satisfy the contract clauses. To do this in Answer Set Programming is not straightforward, as it is a declarative programming language which works by producing answer sets to problem domains outlined by the designer.

This week, I was unable to generate trajectories that don't satisfy the contract clauses. The way that I have the contract set up is a set of clingo statements that indicate rules that exclude certain answer sets, for example if a shipment is not received of the proper amount, a clause will be broken, and that answer set will be excluded from the program's output. For reasoning about these trajectories, further consideration must be made to the structure of the entire program. Changes would have to be made such that the trajectories can be computed individually, and then reasoning tasks would be performed by the agents over the joint compatible trajectories.

Results

This week we learned about computing trajectories in clingo and found that the generation technique I have implemented was not as desired. We need to be able to generate trajectories that do not satisfy a contract in order to perform backtracking and to have the agents able to resolve these planning mistakes, but as it stands the agents only generate trajectories that satisfy the contract and provide no useable trajectories otherwise, which would be used for reasoning.

Conclusion of Project and Future Work

Due to time constraints, and the closure of this course, this final report covers additional issues that need more attention. Uncovering a structural flaw in my approach with little time left is disheartening, but I plan to continue work on this project after submission of my final report. Knowledge Representation and Reasoning is an interesting field and I have enjoyed researching different logic-based approaches to improving supply chain resilience. We have additional work planned to add new reasoning capabilities to this framework, and to expand it if need be.

To address the current issues with my implementation, I plan to study the framework more closely and take some time to delve deeper into literature regarding Answer Set Programming. As a new paradigm that I explored for this project, it was very interesting to use and gave me many new insights into problem solving and programming in general. Though, at the same time it proved to be very difficult to adapt my mind to think using the declarative approach as opposed to the object-oriented or procedural approaches I am used to. By furthering my foundational knowledge on ASP, and studying the framework more closely, I believe I will be able to reconfigure the structure of my approach to be better suited for these reasoning tasks.

Azadeh Izadi 4/18/2024

References

Chen, B., & Cheng, H. H. (2010). A review of the applications of agent technology in traffic and Transportation Systems. *IEEE Transactions on Intelligent Transportation Systems*, *11*(2), 485–497. https://doi.org/10.1109/tits.2010.2048313

Dominguez, R., & Cannella, S. (2020). Insights on multi-agent systems applications for Supply Chain Management. *Sustainability*, *12*(5), 1935. https://doi.org/10.3390/su12051935

Falkner, A., Friedrich, G., Schekotihin, K., Taupe, R., & Teppan, E. C. (2018). Industrial applications of answer set programming. *KI - Künstliche Intelligenz*, *32*(2–3), 165–176. https://doi.org/10.1007/s13218-018-0548-6

Flynn, D., Nadeau, C., Shantz, J., Balduccini, M., Son, T. C., & Griffor, E. R. (2023). Formalizing and reasoning about supply chain contracts between agents. *Practical Aspects of Declarative Languages*, 144–160. https://doi.org/10.1007/978-3-031-24841-2_10

Li, J., & Chan, F. T. S. (2013). An agent-based model of supply chains with dynamic structures. *Applied Mathematical Modelling*, *37*(7), 5403–5413. https://doi.org/10.1016/j.apm.2012.10.054

Metzger, M., & Polakow, G. (2011). A survey on applications of Agent Technology in Industrial Process Control. *IEEE Transactions on Industrial Informatics*, *7*(4), 570–581. https://doi.org/10.1109/tii.2011.2166781

Russell, S., Norvig, P., Chang, M.-W., Devlin, J., Dragan, A., Forsyth, D., Goodfellow, I., Malik, J. M., Mansinghka, V., Pearl, J., & Wooldridge, M. (2022). Chapter 7: Logical Agents. In *Artificial intelligence a modern approach*. Pearson.