Obtaining Plan Traces from Natural Language Conversations in a Healthcare Chatbot

Bianca Precebes da Silva, Maria Viviane de Menezes[1], Lívia Almada Cruz[1], Bruno da Silva Pinho, Luis Fernando Oliveira Sousa

[1]Universidade Federal do Ceará (UFC)
Quixadá – CE – Brazil

bprecebes@gmail.com, {vivianemenezes,livia.almada}@ufc.br

brunopinho123@gmail.com, fernando.osousa@alu.ufc.br

Abstract. *Automated Planning is a subarea of Artificial Intelligence (AI) that studies the deliberative process of choosing actions for an agent to achieve its goals. A planner is a problem-solving algorithm that takes as input a high-level description of the agent and its environment (planning domain) and produces a sequence of actions (plan) that moves the agent from an initial state to a goal state. Acquiring a planning domain can be performed using algorithms that learn the domain description from a kind of specific data (plan traces). Obtaining such plan traces is difficult and a viable strategy is to extract it from Natural Language information. This work aims to build a database of plan traces from Natural Language dialogue data. These data represent conversations between real users and a healthcare chatbot.*

1. Introduction

Planning is an essential aspect of human behavior, enabling to select and organize actions to achieve specific goals. Automated Planning is a subarea of Artificial Intelligence (AI) dedicated to the computational modeling and execution of planning processes [Ghallab et al. 2004]. A planning domain is defined in terms of predicates and actions, which represent the properties of the environment and the capabilities of the agent, respectively. A planning problem is instantiated in a domain by defining an initial state and a desired goal. A solution to a planning problem is a plan: a sequence of actions that transforms the initial state into a state in which the goal is satisfied [Russell and Norvig 2009].

Classical Planning operates under the assumption of a deterministic environment, in which the outcomes of an agent's actions are not subject to uncertainty [Pereira and Barros 2007]. In real-world scenarios, this assumption often does not hold, as actions may produce non-deterministic effects due to the inherent dynamic nature of the environment. A solution to a non-deterministic planning problem is called a policy: a mapping from states to actions that prescribes which action to take in each possible state to achieve the goal, considering the uncertainty in outcomes. Planning domains and problems are typically specified using PDDL (Planning Domain Definition Language) [Haslum et al. 2019], a declarative language widely adopted by the planning community.

The acquisition of planning domain models is a complex task, as it demands specialized knowledge of the domain being modeled. In the literature of Knowledge Engineering for Planning [Vallati and Kitchin 2020, kep 2024], domain acquisition paradigms

are typically categorized as either manual or automated. In automated approaches, the fundamental data used to infer the underlying structure of the domain is referred to as *plan traces*. These traces are generally composed of two main components: (a) the plan, which corresponds to the sequence of executed actions; and (b) the state list, which captures the sequence of states traversed during the execution; these traces are difficult to obtain. `NLtoPDDL` [Miglani and Yorke-Smith 2020] is a automatic approach for acquiring planning domains that obtain plan traces from unstructured data in Natural Language.

Dialogue systems (or chatbots) may range from simple question–answer exchanges to more complex, goal-oriented conversations [Botea et al. 2019], such as support systems used in healthcare. During information processing, user utterances (i.e., responses at each interaction step) are classified into entities (domain-relevant variables) and intents (categories corresponding to expressions identified in the text). In addition to the general components of Natural Language understanding and generation, the *dialogue manager* is a core component of these systems. It is responsible for identifying the current dialogue state and selecting the next action to be executed [Teixeira and Dragoni 2022].

Plantão Coronavírus is a chatbot designed to provide guidance to the population during the COVID-19 pandemic. It enabled citizens to access reliable information about the virus and report symptoms, which were then evaluated by the system to determine whether redirection to appropriate healthcare professionals, such as physicians, nurses or psychologists. The chatbot played a relevant role during the most critical phases of the pandemic (2020 and 2021). Previous work [Pinho 2024] introduced an AI Planning-based approach to manage the chatbot's behavior, adopting a *manual planning domain acquisition* strategy derived from a limited set of dialogue samples.

In this work, we aim to extract plan traces from dialogues, which are essential for the automatic acquisition of planning domains. We leverage a large dataset of approximately 7,000 dialogues and apply Machine Learning algorithms to identify the set of actions in the planning domain, as well as the corresponding paths reflected in real user interactions. The main objective is to construct a structured database of plan traces from unstructured Natural Language dialogues collected during real interactions with a healthcare chatbot. To achieve this, we follow a multi-step process: (i) extracting entities from user utterances; (ii) deriving user predicates based on the identified entities and intents, which constitute the state descriptions (part (b) of the plan trace); and (iii) extracting action sequences from the dialogues, corresponding to the plans (part (a) of the plan trace). We evaluated the quality of the resulting plan traces by comparing the induced paths with those defined by the chatbot's original management policy.

This paper is organized as follows. Section 2 presents the theoretical foundation. Section 3 discusses related work. Section 4 describes the methodology adopted to extract plan traces from dialogues, including the data processing and Machine Learning techniques employed. Section 5 presents and analyzes the results obtained. Finally, Section 6 concludes the paper and outlines directions for future work.

2. Theoretical Foundations

2.1. Automated Planning

Automated Planning is a subrea of AI concerned with the representation and generation of a sequence of actions that enables an intelligent agent to achieve its goals

[Russell and Norvig 2009]. The classical planning approach assumes that the environment evolves in a deterministic way. Thus, the actions executed by the agent are not influenced by external environmental factors [Pereira and Barros 2007]. In this context, a classical planning domain can be represented by a state-transition model. In this model, a state is a specific configuration of the environment labeled by a set of propositions, and actions trigger transitions that cause the environment to evolve from one state to another. Thus, a specific problem within the domain is defined by the addition of an initial state and a goal. A *plan* is the sequence of actions performed to achieve the agent's goal.

PDDL (Planning Domain Definition Language) [McDermott et al. 1998] is a language widely adopted as the standard for specifying planning domains and problems. In this language, each action is represented by preconditions and effects, which define the state of the agent and the environment before and after the action is executed, respectively. The preconditions of actions are sets of propositional atoms that define the propositions that must be true for an action to be executed in a given state. The effects consist of the outcome of the execution, with their respective propositions added or removed, which alter the current state of the environment [Ghallab et al. 2004]. In PDDL, action schemas are used, which are representations with variables that are instantiated by propositional atoms. Given a planning problem, a *planner* performs a search, generating a state transition graph on demand from the action language, aiming to find a path that leads from the initial state to a goal state [Pereira and Barros 2007].

In real-world problems, the agent's actions exhibit *uncertain effects*, meaning the actions can be *non-deterministic* [Pereira and Barros 2007]. A *non-deterministic planning domain* is a model that accounts for uncertainties in nature and the environment [Pereira and Barros 2007]. Each action has a set of possible effects representing different configurations. A solution to a non-deterministic planning problem is characterized in terms of the possible executions of a plan, defined as a *policy* [Cimatti et al. 2003].

2.2. Planning Domain Acquisition

The domain acquisition process is a complex activity, primarily because it requires extensive knowledge about the problems to be modeled and their respective characteristics [Segura-Muros et al. 2021]. In this context, obtaining a domain model for real-world planning problems requires collaborations between Subject Matter Experts (SMEs) and Knowledge Engineers (KEs). However, this results in a time-consuming process that is prone to errors and difficult to understand. According to [Miglani and Yorke-Smith 2020], the main paradigms for acquiring planning domain models can be summarized into three types of approaches: (i) the manual approach, (ii) the automatic approach from structured data (plan traces), and (iii) the automatic approach based on unstructured data (from Natural Language).

The *manual approach* involves the active participation of subject matter experts, as it requires specific knowledge about the domains that can only be provided by these professionals. Thus, they are responsible for the entire knowledge engineering process, dealing with the acquisition, formulation, validation, and maintenance of planning for the production of the domain model [Shah et al. 2014]. The main issue with this approach is the potential errors introduced through manual coding, as the knowledge of real-world domains is often limited by human perspective, and the success of planning systems entirely depends on the skills of the expert defining the model [Arora et al. 2018].

Figure 1. Example of plan trace in the blocks world domain.

(a) Plan

Start	End	Action
s9	s6	unstack C A
s6	s7	stack B C
s7	s11	stack A B

(b) List of states

State	Predicates
s9	(on A table) $\wedge$ (on C A) $\wedge$ (on B table) $\wedge$ (clear B) $\wedge$ (clear C)
s6	(on A table) $\wedge$ (on B table) $\wedge$ (on C table) $\wedge$ (clear A) $\wedge$ (clear B) $\wedge$ (clear C)
s7	(on A table) $\wedge$ (on C table) $\wedge$ (on B C) $\wedge$ (clear A) $\wedge$ (clear B)
s11	(on C table) $\wedge$ (on B C) $\wedge$ (on A B) $\wedge$ (clear A)

The *automatic approach from structured data* learns the domain model through plan histories, also referred to as *plan traces*. With previously defined information about intermediate states, a *plan trace* consists of a sequence of actions and intercalated states [Miglani and Yorke-Smith 2020]. Each action is associated with two states: one representing the world at the beginning of the action (pre-state) and another representing the world after the execution of the action (post-state). It is possible to have partial information regarding the pre-state and post-state.

Figure 1 illustrates a *plan trace* of a plan in the blocksworld domain [Nilsson 1980]. In the *plan trace*: (a) shows the sequence of executed actions, with the *Start* and *End* columns indicating the names of the states associated before and after the application of a specific action; (b) displays the set of intermediate states associated with each action in the plan during its execution.

2.3. Natural Language Processing (NLP)

Natural Language Processing (NLP) is a subarea of AI aimed at enabling machines to understand, interpret, and manipulate human language [Jurafsky and Martin 2021]. According to [IBM 2024], among the tasks of NLP, Named Entity Recognition (NER) stands out. The NER task is to identify named entities present in texts and classify them into predefined categories, such as person, location, or organization. For example, considering the sentence "My name is John and I study at the Cool School" traditional NER labels identify the words "John" and "Cool School" as person and organization, respectively.

State-of-the-art NER systems employ Machine Learning algorithms [Rauber 2005] that are pre-trained using language models. The use of such models, such as OpenAI GPT [Radford et al. 2018] and BERT [Devlin et al. 2019], significantly improves the performance of many NLP tasks and also reduces the amount of labeled data needed for learning. spaCy [AI 2024] is one of the most popular libraries for NLP and has the capability to perform NER.

2.4. Dialogue Systems

Dialogue systems, commonly known as chatbots, are intelligent agents focused on interacting with users. They are designed to interpret questions and respond to them, reinforc-

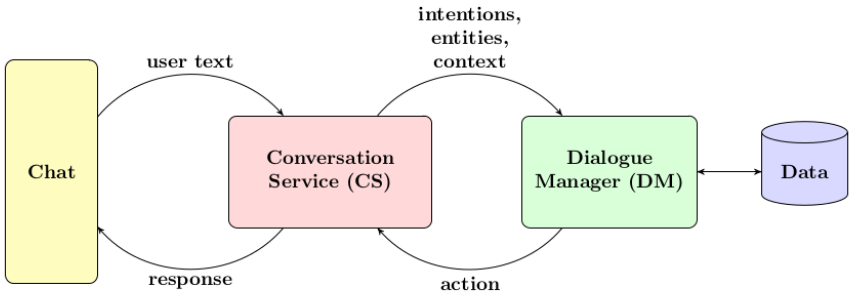
ing the illusion that the conversation is taking place between two human beings.

The construction of dialogue is typically designed with three main components [Teixeira and Dragoni 2022]: (i) the component responsible for Natural Language Understanding (NLU), which receives user input and is able to interpret it; (ii) the *dialogue manager*, which perceives the current state of the dialogue and decides the next action to be executed; and (iii) the Natural Language Generation (NLG) component, which converts the action into a response that is understandable by the end user.

In a dialogue, the user utterance refers to the text typed by the user in the system's interface. For processing this information, the utterances are classified into intents, which define the categories of expressions used by the user. For example, a statement like "I would like to book a flight to Bucharest" can be classified under the "FLIGHT-BOOKING-REQUEST" intent. Additionally, entities are defined as the representative domain variables present in the user's utterances, which can either be general (places, dates, people) or specific to the domain in question. In the example statement, "Bucharest" could represent an assignment to the entity "$PLACE". These pieces of information (intents and entities), along with the dialogue history, help build the context of the conversation, which contains relevant data up to a specific point in the dialogue [Botea et al. 2019].

The Figure 2 shows the general architecture of a dialogue system. The Chat module in the diagram is responsible for receiving the user's utterance and establishing communication. The utterance is then sent to the Conversation Service (CS) module, where the components responsible for understanding (NLU) and generating (NLG) the Natural Language data process the sentences into categories and perform the necessary translations for understanding by both the machine and the user. It is within the CS that the user's utterances are classified into intents and the entities are recognized according to the dialogue characteristics, thereby extracting the context of the conversation.

Figure 2. Dialogue system architecture [McTear 2020].



The Dialogue Manager (DM) module is the central component of a dialogue agent, as it handles the intents, entities, and context in a way that controls the flow of the conversation and produces response messages to the user [McTear 2020]. Computationally, a dialogue can be interpreted as a series of communicative actions used to achieve specific goals. Thus, each utterance can be treated as an action by the dialogue manager, which addresses the problem of identifying the current status of the conversation in order to decide the next action to be executed in a given state [Teixeira and Dragoni 2022]. After the action is chosen, the DM sends it to the Conversation Service (CS), where text translation

and response dispatch to the Chat module occur.

The management of dialogue systems can be classified into two categories: (i) those that use dialogue trees and (ii) those that use Machine Learning [Botea et al. 2019]. Dialogue trees structure the conversation into a complex series of branches that depend on the user's responses. As the configurations become more complex, the tree grows quickly, with many similar dialogue interactions that are difficult to track. Moreover, systems using dialogue trees rely on the costly manual description of all possible conversation paths. On the other hand, systems that use Machine Learning for management employ techniques to calculate the next response using examples as a starting point, allowing for the rapid development of complex dialogue from training data. This characteristic, however, may lead to limitations regarding the quality of the data and the capacity of the simulators used in the process. Furthermore, the approach is not fully capable of providing explanations for the paths taken in the resulting decisions [Teixeira and Dragoni 2022].

Considering dialogue as a path to achieving a specific goal through the selection of actions, the Automated Planning approach can be a promising alternative for dialogue management tasks [Teixeira and Dragoni 2022]. By anticipating the outcome of each of these actions, a planner can identify the states that can be reached and choose an efficient path that leads to the dialogue's goal [Teixeira et al. 2019]. The planner assesses the conversation's context and decides which action should be taken. Dialogue systems must be designed in a way that enables them to explain their reasoning, in order to justify their choices to users and comply with ethical standards [DEEP-DIAL 2021].

3. Related Work

The NLtoPDDL approach [Miglani and Yorke-Smith 2020] aims to perform the automatic acquisition of planning domains from unstructured data, specifically Natural Language texts. The inputs used are process manuals that present data in the form of instructional lines. The process of domain acquisition is divided into two subproblems: (i) the extraction of action sequences for subsequent plan extraction from unstructured data, and (ii) the automated acquisition of the planning domain using the structured data obtained in the previous phase. The algorithm associates words in the text with labels such as *actions* and *states*, seeking to extract words that may represent actions or action arguments. The [Miglani and Yorke-Smith 2020] approach is similar to the one proposed in this work regarding the automatic acquisition of planning domains from unstructured data. However, the authors perform the complete domain acquisition process, unlike this research, where we focus on the first stage of the study, which is the acquisition of structured data in the form of plan traces from real texts in Natural Language, using data analysis tools for classification and information extraction.

In the work of [Segura-Muros et al. 2021], the authors propose a planning domain acquisition process based on Machine Learning techniques from structured data, named PlanMiner. The approach enables the acquisition of a domain with numerical predicates and logical-arithmetic relations between predicates expressed in PDDL, using plan traces with partially known states as input. PlanMiner addresses the domain learning problem by describing it as a classification task. In order to model an action, it is essential to identify its preconditions and effects. Based on this, the authors argue that the classification approach is suitable, as these properties correspond to the configurations of pre-states

(states where the action is applied) and post-states (resulting states after the action execution). The learning process in PlanMiner is divided into the following phases: (i) Dataset extraction: A set of plan traces is received as input and used to generate a collection of datasets; (ii) Discovery of new information: PlanMiner applies regression techniques [Manly and Alberto 2016] to infer new knowledge and enrich the datasets; (iii) Acquisition of classification models: The generated datasets are used as input for a classification algorithm; and (iv) Generation of the planning domain: The classification models are processed to produce a set of action models. The final output of PlanMiner is a PDDL domain formed by the combination of the learned action models.

Considering the [Segura-Muros et al. 2021] approach for domain acquisition, while the authors use as input the *previously structured data in the form of plan traces*, without identifying how they were obtained, the present work focuses on using unstructured data in Natural Language from dialogues to obtain the plan traces, as a prior step to domain acquisition. The difference between the studied domains should also be mentioned, as the PlanMiner authors focus on benchmark domains in the planning area and the NLtoPDDL approach focuses on diverse applications, while this work aims to study domains that assist in the task of dialogue management.

4. Methodology

The aim of this work is to extract plan traces (structured data) from dialogue data, which are inherently unstructured and expressed in Natural Language. These inputs correspond to conversations from a healthcare chatbot. By structuring this data, it becomes possible to automatically acquire the chatbot's planning domain, which will be used in the context of dialogue management. The information handled is confidential; therefore, in order to protect user privacy, the examples presented in this paper are fictional and were created only for illustrative purposes, although they are representative of the real data collected.

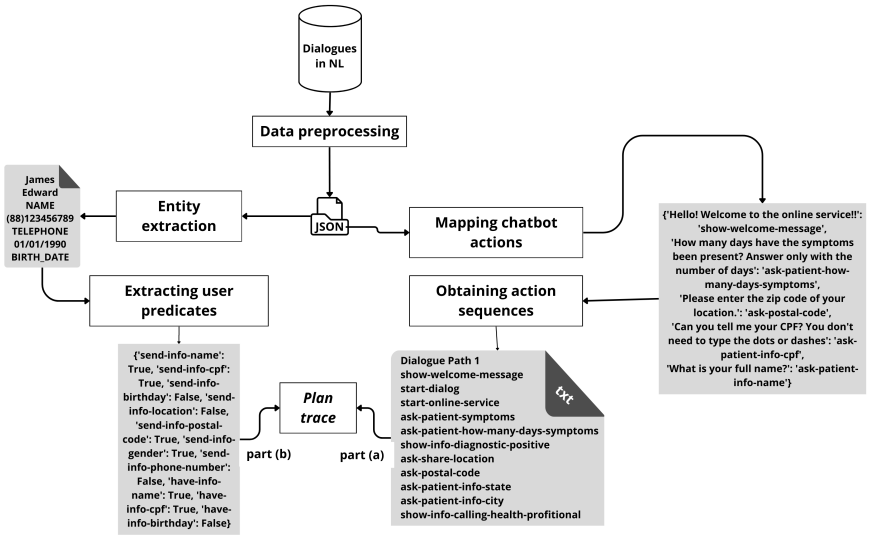
The conversation initially takes place between the user and the chatbot, but as it progresses, it may transition to a dialogue between the user and a healthcare professional. In this context, the planning problem addressed in this work is the same across all dialogues: the initial state is the start of the conversation and the goal is successfully concluded the interaction.

Figure 3 illustrates the architecture of the system used to extract plan traces from dialogue data in Natural Language. Google Colaboratory[1] tool was chosen as the development environment and the Python programming language was used to implement the algorithms for accessing the dataset. During the preprocessing stage, columns deemed unnecessary for the research objectives were removed, as well as emoticons and excessive spacing between sentences. Each line in the dataset undergoes this process, resulting in properly formatted dialogues that are saved in a JSON file for later use in the process.

In order to extract the *entities* represented in the user statements, the spaCy[2] library was used, specifically the Portuguese language model *pt_core_news_lg*. The spaCy NER model, when applied to Portuguese language, recognizes entities related to person, location, organizations and miscellaneous entities. To fit the scope of this work, entities

[1] `https://colab.google/`
[2] `https://spacy.io/`

Figure 3. Process of obtaining plan traces from dialogues.



related to organizations and miscellaneous categories were disregarded, and the spaCy model was updated with entities present in the dialogue data, such as postal code (CEP), phone number, CPF, gender and date of birth, totaling 7 entities. This update was carried out using regular expressions to ensure the correct format of the data, a validation function for data such as CPF, and the PhraseMatcher[3] class from spaCy, which recognizes matching patterns in the text.

User predicates are obtained by analyzing the entities and intents present in the user statements, such that for each intent and entity, there is a related predicate. For example, the user statement "I am a woman" corresponds to the intent "send-info-gender", and the text "woman" is an entity "have-info-gender". These predicates are true when the entity and intent are found in the dialogue flow. Therefore, the entities defined in the previous stage, which include name, location, postal code (CEP), phone number, CPF, gender, and date of birth, were mapped and translated into corresponding user predicates. In the extraction process, this mapping is performed using Python dictionary. Thus, each key in the dictionary represents a user predicate, with its corresponding boolean value.

In order to obtain the action sequences, as the dialogue is processed each line corresponding to the chatbot's messages undergoes a verification process. This process determines which action corresponds to the message sent, based on the previously established mapping. As a result, the path taken by the dialogue is returned, outlining all the actions executed by the chatbot throughout the conversation flow. Each dialogue goes through this extraction process, resulting in a text file containing the action sequences.

5. Results

The implementations were carried out on Google Compute Engine back-end notebooks using Python 3, running on an Intel(R) Xeon(R) CPU 2.20GHz with 12.7GB of RAM and 107.7GB of disk space. This configurations was used in the notebooks related to data preprocessing and the extraction of action sequences. For user predicate extraction,

[3] `https://spacy.io/api/phrasematcher`

we use a Google Compute Engine back-end notebook with Python 3, configured with a TPU with 8 cores, 334.6*GB* of RAM, and 225.3*GB* of disk space. The implementations developed and the files used in this research are available in the supplementary material[4]. The dataset contains **7,136** dialogues and a sample of **1,125** dialogues underwent the preprocessing stage. The properly formatted dataset was saved in a JSON file.

Based on the recognition of entities present in the user's statements, 14 predicates were mapped, with each of the 7 entities covering predicates with the prefixes `HAVE` (derived from the entity) and `SEND` (from the intent). Initially, the predicates are assigned a false boolean value, indicating the start of a new dialogue. Each line from the patient's conversation was analyzed, and if the algorithms detected entities in the text, the values of the user predicates were updated, resulting in a contextual record at each interaction. Table 1 shows predicates values, indicating the state of the dialogue.

Table 1. Example of extracting user predicates from each analyzed utterance.

User statement	**Predicate values**
01: no entities found	'send-info-name': False, 'send-info-cpf': False, 'send-info-birthday': False, 'send-info-location': False, 'send-info-postal-code': False 'have-info-name': False, 'have-info-cpf': False, 'have-info-birthday': False, 'have-info-location': False, 'have-info-postal-code': False
02: postal code entity found	'send-info-name': False, 'send-info-cpf': False, 'send-info-birthday': False, 'send-info-location': False, **'send-info-postal-code': True** 'have-info-name': False, 'have-info-cpf': False, 'have-info-birthday': False, 'have-info-location': False, **'have-info-postal-code': True**

Each list of predicates characterizes a possible state in the state list of the plan trace (part (b)), representing a specific configuration of the environment at each user interaction through propositions. Considering the 7 entities and their boolean values, there is an expected total of 2^7 possible predicate configurations, i.e., 128 states. After analyzing the processed dialogues and using a script to identify the number of unique user predicate dictionaries, a list of 45 states was obtained. Including the initial state in which all predicates have a false boolean value, a total of **46 states** was established.

Table 2 shows the action mapping that was carried out, represented as a Python dictionary, where the key is the string corresponding to the action name and the value is the string referring to the message sent by the chatbot. Each line from the chatbot's messages was analyzed[5], and based on the mapping, the corresponding action for each text was identified and added to the dialogue path.

This process resulted in the extraction of **1,125** dialogue paths, with an execution time of 9 seconds. The sequence of actions represents part (a) of the plan trace, characterizing the plans obtained from each dialogue. However, the intermediate states were

[4] `https://zenodo.org/records/15391970`

[5] The texts were translated into English; but the original messages are in Portuguese.

Table 2. Example of action mapping performed based on chatbot messages.

Chatbot Message	Action
Hello! Welcome to the online service!!	`show-welcome-message`
What do you need now? (Type the number of the desired option) 1. Online assistance about the Coronavirus 2. Information about the Coronavirus 3. Mental healthcare 4. Others	`start-dialog`
Are you experiencing any of these symptoms? [...] Type the numbers of the symptoms separated by commas (e.g., 2, 5), or type 7 if you are not experiencing any of the symptoms listed above.	`ask-patient-symptoms`
Your service has been completed.	`finish-service`

not mapped. To evaluate the resulting sequence of actions, we compared all the chatbot management policy paths (structured data). The goal was to map which policy paths are actually followed in real conversations. This mapping was important since the domain used to generate that policy was manually constructed based on a small set of real dialogues (around 30 dialogues). The original domain has 89 predicates and 60 actions[6], which produces a policy with **457** distinct action sequences that guide the agent from the initial state to a goal state. Some other important results:

- In the action analysis, **21 new actions** were mapped, representing scenarios not covered by the policy;
- 779 dialogues (69.24%) do not correspond to a complete path in the policy, i.e., from the initial state to the goal. These dialogues are characterized by incomplete interactions, such as interrupted conversations or disorganized sequences.
- In 474 dialogues, users made some type of mistake in their statements (e.g., incorrect data formats or selecting nonexistent menu options), resulting in loops. These are typical situations in dialogue system interactions, reflecting the nondeterministic nature of actions.

Supplementary material[7] shows figures that illustrate examples of action sequences obtained in this study and their respective paths within the policy. The dashed lines represent edges that do not exist in the policy, indicating possible paths that had not been previously mapped. The analysis of this information allows for the following observations: (i) discovery of new actions that had not been previously mapped; (ii) the dialogues include new interactions with the chatbot after the user has already been assisted by a healthcare professional, which are not represented in the policy; (iii) the dialogues encompass follow-up conversations from a monitoring program that were not covered by the policy, although they share some common actions and; (iv) the dialogues also include shorter interactions where only the CPF or name are requested for identification.

[6] `https://editor.planning.domains/#read_session=zkrf2y0dSL`
[7] `https://zenodo.org/records/15391970`

6. Conclusion

Plan traces are essential for the automatic acquisition of planning domains, as it corresponds to a structured representation of real-world data that is necessary for learning predicates and actions. The goal of this work is to obtain plan traces from dialogue data collected from a healthcare chatbot, which consists of unstructured data in Natural Language. Thus, the main contributions of this research are: (i) Development of a dataset consisting of 1,125 plan traces, necessary for the automatic acquisition of the planning domains which can be used to manager the dialogue; (ii) Coverage analysis to determine how well the real dialogues are represented by real policy used to manager the chatbot, which was obtained from a manually created domain based on a small number of real dialogues; and (iii) Discovery of new actions that were not mapped in the original policy.

As future work, we intend to integrate the traces produced in this study with those obtained from the policy, in order to build a comprehensive dataset suitable for the automatic acquisition the planning domain used to manager the dialogue. This domain, automatically generated from the plan traces, will be compared with the manually acquired domain. Furthermore, considering the efficiency of Large Language Models (LLMs) in Knowledge Extraction, we intend to analyze the insertion of new techniques using new models to efficiently compare the different acquisition processes.

References

(2024). Knowledge engineering for planning and scheduling (keps) workshop – icaps 2024. `https://icaps24.icaps-conference.org/program/workshops/keps/`. Accessed: 2025-05-09.

AI, E. (2024). spacy: Industrial-strength natural language processing in python.

Arora, A., Fiorino, H., Pellier, D., Métivier, M., and Pesty, S. (2018). A review of learning planning action models. *The Knowledge Engineering Review*, 33.

Botea, A., Muise, C., Agarwal, S., Alkan, O., Bajgar, O., Daly, E., Kishimoto, A., Lastras, L. A., Marinescu, R., Ondrej, J., Pedemonte, P., and Vodolán, M. (2019). Generating dialogue agents via automated planning. *CoRR*, abs/1902.00771.

Cimatti, A., Pistore, M., Roveri, M., and Traverso, P. (2003). Weak, strong, and strong cyclic planning via symbolic model checking. *Artificial Intelligence*, 147(1):35–84. Planning with Uncertainty and Incomplete Information.

DEEP-DIAL (2021). The fourth workshop on reasoning and learning for human-machine dialogues.

Devlin, J., Chang, M.-W., Lee, K., and Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Ghallab, M., Nau, D., and Traverso, P. (2004). *Automated Planning: Theory and Practice*. Elsevier Science.

Haslum, P., Lipovetzky, N., Magazzeni, D., Muise, C., Brachman, R., Rossi, F., and Stone, P. (2019). *An introduction to the planning domain definition language*, volume 13. Springer.

IBM (2024). O que é processamento de linguagem natural (pln)?

Jurafsky, D. and Martin, J. H. (2021). *Speech and Language Processing*. Prentice Hall, 3rd edition.

Manly, B. F. and Alberto, J. A. N. (2016). *Multivariate statistical methods: a primer*. Chapman and Hall/CRC.

McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M., Weld, D., and Wilkins, D. (1998). Pddl - the planning domain definition language.

McTear, M. (2020). *Conversational AI: Dialogue Systems, Conversational Agents, and Chatbots*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Miglani, S. and Yorke-Smith, N. (2020). NLtoPDDL: One-shot learning of PDDL models from natural language process manuals. In *Proceedings of the ICAPS Workshop on Knowledge Engineering for Planning and Scheduling (KEPS)*. ICAPS.

Nilsson, N. J. (1980). *Principles of artificial intelligence*. Tioga Publishing Co.

Pereira, S. d. L. and Barros, L. N. d. (2007). *Planejamento sob incerteza para metas de alcançabilidade estendidas.* 2007. Doutorado em ciência da computação, Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo.

Pinho, B. S. (2024). Planejamento não-determinístico para o gerenciamento de agentes de diálogos orientados à meta. Master's thesis, Universidade Federal do Ceará.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. (2018). Improving language understanding with unsupervised learning. *Technical Report*.

Rauber, T. W. (2005). Redes neurais artificiais. 29.

Russell, S. and Norvig, P. (2009). *Artificial Intelligence: A Modern Approach*. Prentice Hall Press, USA, 3rd edition.

Segura-Muros, J. Á., Pérez, R., and Fernández-Olivares, J. (2021). Discovering relational and numerical expressions from plan traces for learning action models. *Applied Intelligence*, 51(11):7973–7989.

Shah, M. M. S., Chrpa, L., Jimoh, F., Kitchin, D. E., McCluskey, T. L., Parkinson, S., and Vallati, M. (2014). Knowledge engineering tools in planning : State-ofthe-art and future challenges.

Teixeira, M. S. and Dragoni, M. (2022). A review of plan-based approaches for dialogue management. *Cogn. Comput.*, 14(3):1019–1038.

Teixeira, M. S., Dragoni, M., and Eccher, C. (2019). A planning strategy for dialogue management in healthcare. In *SWH 2019 - Second International Workshop on Semantic Web Meets Health Data Management*, pages 42–50.

Vallati, M. and Kitchin, D. (2020). *Knowledge Engineering Tools and Techniques for AI Planning*. Springer.