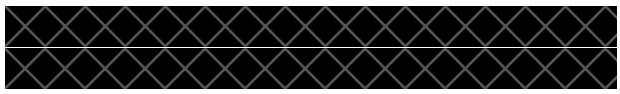
CIS700 HW4



March 3, 2022

1. What wikiHow article did you pick and why?

   We chose the article How to Survive in the Woods.
   The main reason why we chose this article is that the survival story was close to an adventure game in nature. Each part of the article can be seen as a problem or set of problems, which requires the right items, tools, actions and location environment to solve.

2. What portions of the article did you select to translate to PDDL?

   We implemented three problems which are part 2 - building a shelter, part 3 - forage for edible wild plants and part 3 - build a snare if you have string or wire.

3. Give some examples of the actions, types, and predicates you used in your domain.

   We implemented 10 actions other than those provided for the action castle: drop, clear_spider, insert_beam, lean_branches, layer, wash, eat, make_snare, make_horizental_bar, hang_snare.

   Take eat as an example, in our second question - forage for edible wild plants, the npc was initially set to be hungry (which is a predicate) and the goal of the game is not npc hungry. So, in order to achieve this goal, the PDDL needs to find its way to use the action eat, which will trigger the backwards search until it finds a proper plan for this.

   For most of the objects which can be get or drop, we make them the subtype for item, in which way, we could use single get and single drop to handle all these objects, instead of using get_flower or get_beam separately. While each of them has its own types, which will make the special action perform well. For example, in order to hang_snare, we need to have an item of snare type and an item of bar type.

We also included 17 predicates in total to depict the whole picture of the three problems according to the article How to Survive in the Woods. Some of them are, (flat ?loc - location), which indicates whether a location is flat, since we need to find a flat place to build the shelters for the npc, (unwashed ?item - item), which indicates that whether the item is washed, we need to check this, since the npc needs to find something that is edible and clean, so that the npc can eat the item.

4. Explain what goal you selected for your problem, and give the initial state and solution that you created.

4.1 build_shelter

Goal: Shelter is built on the west_plain and the player is also at west_plain.
Initial state:
NPC at camp;
Location map:
(connected camp west path) (connected path east camp) (connected path west cliff) (connected cliff east path) (connected cliff up waterfall) (connected waterfall down cliff) (connected waterfall east mountain) (connected mountain west waterfall) (connected mountain east west_plain) (connected west_plain west mountain) (connected west_plain east east_plain) (connected east_plain west west_plain);
A long and thick branch at mountain;
A bunch of smaller branches, twigs and leaves at east_plain;
Leaves have bug on them;
West_plain is flat and dry;
A tree at west_plain has split in the trunk.
Solution:
NPC goes west from camp to path ->
NPC goes west from path to cliff ->
NPC goes up from cliff to waterfall ->
NPC goes east from waterfall to mountain ->
NPC gets (main) branch at mountain ->
NPC goes east from mountain to west_plain ->
NPC goes east from west_plain to east_plain ->
NPC gets leaves at east_plain ->
NPC gets twigs at east_plain ->
NPC gets (a bunch of small) branches at east_plain ->
NPC goes west from east_plain to west_plain ->
NPC clears spiders on leaves ->

NPC inserts main branch into the split in the trunk of a tree in west_plain to form a beam ->
NPC leans smaller branches on the beam to form a frame at west_plain ->
NPC layers twigs on the frame and leaves inside the shelter; a shelter at west_plain is completed.

4.2 eat_plants

Goal: NPC is no longer hungry.
Initial state:
NPC at camp;
NPC is hungry;
Location map:
(connected camp west path) (connected path east camp) (connected path west cliff) (connected cliff east path) (connected cliff up waterfall) (connected waterfall down cliff) (connected waterfall east mountain) (connected mountain west waterfall) (connected mountain east west_plain) (connected west_plain west mountain) (connected west_plain east east_plain) (connected east_plain west west_plain);
Waterfall has water source;
Flowers at mountain;
Leaves at east_plain;
Both the flowers and leaves are edible and unwashed;
Solution:
NPC goes west from camp to path ->
NPC goes west from path to cliff ->
NPC goes up from cliff to waterfall ->
NPC gets water at waterfall ->
NPC goes east from waterfall to mountain ->
NPC gets flowers in mountain ->
NPC washes flowers using water ->
NPC eats flowers.

4.3 build_snare

Goal: Snare is hung over a footpath.
Initial state:
NPC at camp;
Location map:
(connected camp west path) (connected path east camp) (connected path west cliff) (connected cliff east path) (connected cliff up waterfall) (connected waterfall down cliff) (connected waterfall east mountain) (connected mountain west

waterfall) (connected mountain east west_plain) (connected west_plain west mountain) (connected west_plain east east_plain) (connected east_plain west west_plain) (connected waterfall west footpath) (connected footpath east waterfall);
A branch at mountain;
A wire on the path;
A footpath is made by animal;
Solution:
NPC goes west from camp to path ->
NPC gets wire on the path ->
NPC goes west from path to cliff ->
NPC goes up from cliff to waterfall ->
NPC goes east from waterfall to mountain ->
NPC gets a branch in mountain ->
NPC goes west from mountain to waterfall ->
NPC goes west from waterfall to footpath ->
NPC makes a snare using wire ->
NPC makes a horizontal bar using branch ->
NPC hangs snare over a footpath with a horizontal bar.

5. What limitations of PDDL did you encounter that makes it difficult to precisely convert a wikiHow description into PDDL?

   There are three main limitations. The first one is that PDDL is very verbose. For example, a plain text like "Search for a dry, flat area between 2 trees with splits in their trunks. ", when it is transformed into a PDDL, it becomes 5 predicates. So if we want to convey the meaning of a sentence, it leads to a long series of preconditions. Its feasibility becomes very poor. Secondly, it is very difficult to use PDDL to represent delicate activities. For example, although we could summarize the whole process of building a snare with wire by one single action “make_snare”, what if we want to break it down into specific steps? It is almost impossible to represent something like “make a loop at one end and tie a slip knot, then push the opposite end of the string or wire through the slip knot to form a large circle” with PDDL. Finally, we encountered situations where the plan to reach the goal was very complex and required more actions, in which case it was difficult for the planner to run the results.

6. Could your PDDL be used as an interesting challenge for a text-adventure-style game? If so, how? If not, what would needed to create an interesting challenge?

   We think a complex problem that requires a set of actions, tools, and preconditions like build_shelter might be an interesting challenge in a text-adventure-style game. Because the player needs to go to different places, collect different materials, and need to pre-process the materials, the order in which the action occurs is also critical (for example, we must first have the frame to layer leaves onto it).

   For simple problems like eat_plants and build_snare, limited by the expressiveness of the PDDL, porting them directly to the game would not lead to interesting challenges. However, if we add more states and bifurcations to the game, such as adding situations where players eat poisonous plants, as well as add timestamps and introduce randomness to influence the result of "check if animal has been caught by snare at certain intervals", then they can also become interesting challenges.

7. Discuss how you might use GPT-3 to automatically or semi-automatically convert a wikiHow article to PDDL?

   As mentioned above, the objective of converting a wikiHow article (text-based games) to PDDL requires extracting logical expressions from the given texts. Thus, it could be reshaped into a plan format.

   The initial idea we come up with is to extract the verb that represents the action. For example, we may use a few shot training which consists of the text and the desired actions.

   The following is such an example. (The story is come from the website 10 Lines Short Stories With Moral Lessons for Kids.

   Text: Once there was a dog who wandered the streets night and day in search of food. One day, he found a big juicy bone and he immediately grabbed it between his mouth and took it home.
   Actions: wandered, grabbed
   Text: On his way home, he crossed a river and saw another dog who also had a bone in its mouth. He wanted that bone for himself too. But as he

opened his mouth, the bone he was biting fell into the river and sank. That night, he went home hungry.
Actions: crossed, saw, opened

This is really a tiny example, but it shows that GPT-3 can fairly extract the actions properly, with simply one shot tuning. However, we soon realize that it is enough for converting the GPT-3 to PDDL models, since it has no architecture.

The next step we did is to modify the format by changing the Actions part to, Action1(argument1, argument2, …), Action2(argument1, argument2, …). And the following is the results.

Text: Once there was a dog who wandered the streets night and day in search of food. One day, he found a big juicy bone and he immediately grabbed it between his mouth and took it home.
Actions: search(food), grabbed(juicy bone)
Text: On his way home, he crossed a river and saw another dog who also had a bone in its mouth. He wanted that bone for himself too. But as he opened his mouth, the bone he was biting fell into the river and sank. That night, he went home hungry.
Actions: crossed(river), saw(other dog), opened(mouth), bone(sank)

From above, we can see the actions along with its arguments were successfully extracted.
However, it is not hard to find the problems lays within even this tiny example, which is that the even with the most advanced Davinci model, it can not accurately find the right action if the text is more action-diverse.

Next thing, we think we need to use the GPT-3 to find the sequentility of the plan, such as before, after, or finally, since if we wanted to convert text directly to PDDL, we need also to define the preconditions and effects.

We experimented with several cases, but none of them gives a satisfactory results. It will often mismatch the actions with its possible consequences or preconditions. It might be presumably perform better with a larger fine-tuning datasets, which we currently lack of. But our assumption is that, converting a story directly to the PDDL with the help of the GPT-3 cannot be fully automated. Since, even with the human intelligence, it will still be really difficult to extract the logics, consequences perfectly from a story with rich background stories and diverse actions.