What wikiHow article did you pick and why?

I selected the "How to Open a Coconut" article. I chose this because it appears to be a pretty simple task, but involves many different actions as well as the use of different tools, which I thought would make creating a PDDL interesting. It is also conveniently broken down into subtasks, such as draining the coconut, opening the coconut through smashing, or opening the coconut with the help of an oven or microwave. Furthermore, the task is something that we would perceive as a 'streamlined' task where there is a certain sequence of actions to be taken to complete the task. I wanted to see how PDDL would define such problems and if it would cause issues for planners to plan out the right sequence of actions.

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

I chose to translate draining the coconut, cracking it open by wrapping it in a towel and hammering it, or heating it up in the oven/microwave to make it easier to hammer it open. These correspond to Methods 1, 2, and 3 in the article.

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

Some examples of actions include position (stacking item 1 on top of item 2), poke (use hammer to hammer at a screwdriver to poke a hole in coconut), takeout (take out the screwdriver from the coconut), and drain (drain the coconut's water into a container). Some types include player, item, the different types of tools, and the coconut. I made use of a lot of predicates, including on (item 1 on item 2), pressed (another item is on this item), ontop (this item is on top of another item), has_hole (coconut has hole), has_water (coconut has water), has_meat (coconut has meat), is_open (whether the coconut has been cracked open), and others such as whether the oven is preheated, or if the coconut has been heated.

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

I laid out 3 goals according to the three methods: draining the coconut, cracking it open with a towel and hammer, and heating it up before cracking it open. The initial states for the three respsective problems are:

Problem 1 (has_water glass):
(at coconut table)
(has_water coconut)
(at screwdriver table)
(at hammer table)
(at glass table)
(at self table)

Problem 2 (at meat table):
(connected table down ground)
(connected ground up table)
(at coconut table)
(has_hole coconut)
(has_meat coconut)
(at hammer table)
(at knife table)
(at towel table)
(at self table)

Problem 3 (at meat table & heated coconut):
(connected table down ground)
(connected ground up table)
(connected table right oven)
(connected oven left table)
(connected table left microwave)
(connected microwave right table)
(at coconut table)
(has_hole coconut)
(has_meat coconut)
(at hammer table)
(at knife table)
(at towel table)
(at self table)

The last problem that I defined and attempted was the full sequences from start to end (extracting both water and meat from the coconut). However, this introduced too many branches for BFS to search and took too long. The full sequence is:

1. position screwdriver coconut self table
2. position hammer screwdriver self table
3. poke coconut screwdriver hammer self table
4. remove hammer screwdriver self table
5. takeout screwdriver coconut self table
6. position coconut glass self table

7. drain coconut glass self table
8. remove coconut glass self table
9. wrap coconut towel self table
10. position hammer towel self table
11. strike coconut hammer towel self table
12. remove towel coconut self table
13. position knife coconut self table
14. separatemeat coconut knife self table

This results in the glass filled with coconut water and the coconut meat on the table. With being able to drain the coconut and get the meat as separate goals, I believe this last big problem is also solvable.

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

A particular limitation I encountered was the difficulty in properly defining the uses of items. I could either add a lot of attributes (predicates) to individual items or I could create lots of subcategories under "item". I went with the latter option and found that I essentially just created a 'type' for each tool I had (hammer, screwdriver, knife, container). The generality of 'item' as a type makes it difficult for a planner to do what a human would logically do, since lots of attributes need to be defined (and lots of preconditions checked) for each action to verify logicality. Also, to use PDDL effectively, I had to break down and describe each step precisely, adding detailed individual actions, such as lining up the screwdriver on the coconut, hammering the screwdriver in, and then removing the screwdriver. This is opposed to an action that says "poke hole in coconut". There are also a limited number of tools, and each tool has its own specific functionality, so it felt that the PDDL action definitions were quite restrictive and there wasn't much exploration needed in planning since there were limited choices at each step.

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?

I do see potential in developing my PDDL into an interesting challenge for a text-adventure-style game with goals of competing tasks in the kitchen. However, a lot more locations and items would need to be added. Perhaps the final goal is to cook up a dish that uses the coconut as an ingredient, and my entire PDDL would be a sub-problem in that game, which is a puzzle that the player would need to solve.

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

There are many aspects where GPT-3 could come in handy. For example, we can ask it to summarize the article by listing out the main goals and the specific steps required to achieve each goal. We can further ask it to provide a list of tools/items that we would need. While GPT can potentially also come up with a list of locations, it is often non-intuitive to think about how locations can be laid out, thus it is better to do that manually. For example, in the example I used, a microwave/oven could easily become an item in the kitchen location, depending on the scope of the game/problem. Ultimately, GPT-3 can provide a very strong foundation from which to construct the PDDL by listing out goals, steps, and items needed. With some manual revision, the steps can be turned into a sequence of actions and the goals a series of predicates.