AGI skills

Tom Rochette <tom.rochette@coreteks.org>

August 30, 2025 — 861fb9d0

0.1 Context

0.2 Learned in this study

0.3 Things to explore

1 Overview

What are some skills that an AGI based on ChatGPT would need to have?

2 Prompt the operator to disambiguate

- Ideally operator (user) prompting should be minimal. ChatGPT should use its own discretion as much as possible.
- However in certain circumstances there will be too many options to choose from, or some of the options will require a lot of work to complete and may need the input from the operator in order to proceed.

3 Extract instructions/actions from text

- ChatGPT can only interact with its environment through text. As such, the text instructions it generates need to be turned into computer programs that can be executed.

4 Generate code

- For the purpose of our studies we will try to limit ourselves to using the Python programming language as it should be versatile enough to complete a wide variety of tasks. Furthermore ChatGPT is likely to have been trained on a lot of Python code, so it should be able to generate fairly good Python code that does not require a lot of manual tweaking or iterations.
- We will assume that the code generated is not reusable and single use. The reason we make this assumption is because we want to avoid having to have a separate steps that determines when we should be calling this generated code. As such, when we have generated code, we will execute it immediately, and then continue with the next step of the plan.

5 Execute code

- Once ChatGPT has generated python code, it needs to be able to execute it. As we do not want to have to worry about what the script does, we will need to run it in a sandboxed environment.
- For now we're mostly concerned about the generated code trying to delete files on the system. A simple solution to this approach would be to have the file system containing the operating system, generated code and python libraries be read-only. This would prevent the generated code from deleting files, but it would also prevent it from writing files. We can allow it to write files to a temporary working directory.

6 Identify useful behavior that can be reused

- As ChatGPT goes through the motion of solving problems, certain actions and behaviors will be repeated over and over again, the same way a function gets called over and over during program execution. We need to be able to identify those behaviors and extract them into reusable functions.
- One challenge of identifying useful behaviors is determining its beginning and its end. This is similar to defining the responsibilities of a function and how much code should be in the function.
- Another challenge is determining whether the behavior is useful. What should be the criteria to define something as "useful"? Is it the number of times it gets called? Is it the amount of code it contains? Is it the amount of time it takes to execute?
- Given many prompt-response chains, we need to be able to identify the ones that are similar to each other. This is similar to the problem of identifying duplicate functions in a program.
- If we can reduce prompt and responses to unique identifiers (i.e., numbers), then we can use algorithms such as the longest common subsequence to identify similar prompt-response chains.

7 Recognize when a known behavior can be used

- Once we've identified a behavior that can be reused, we need to be able to recognize when it can be used.
- Interestingly enough, the reuse of behavior may appear similar to the prompt-response loop.

8 Recognize when we're in a prompt-response loop

- A prompt-response loop is when we ask ChatGPT something, it responds, we ask it something else, it responds, but not progress is being made. We're stuck in a loop.
- If we're in a prompt-response loop, we need to be able to break out of it.
- At a high level, identifying that we're in a prompt-response loop is difficult. We could ask ChatGPT whether it thinks that the last few prompt-response are somewhat identical, and if so, we could ask it to prompt the operator for help.
- Ideally we would want to avoid having to do prompt-response loop detection, possibly by tracking some form of plan and identifying when we're not making progress towards the next step. This could be done for example, by recording how many prompt-response cycles we've done for a given step of a plan and prompting ChatGPT for a status update.
- A non automated way to prevent infinite prompt-response loops is to have an operator review the operation logs every X steps and identify when the operation is stuck in a prompt-response loop.
- See Preventing repetition

9 Unsorted

- Recognize when the desired behavior/action has been completed
- Read/Write memories
- Browse and interact with the web
- Interact with its environment (e.g., read/write files from the filesystem, execute programs, etc.)

10 See also

- Limitations
- Preventing repetition
- Seed AI
- Setup passive income streams