G6 is a system with tools, workflows and agents. These are the 3 primitives. Everything can be derived from this (llm, llm/code hybrid, code). Workflows are tools which call agents.

We never just call and LLM - everything has agency. We have curated highly specialised RAG pipelines with agents optimised for real world problem solving, open ended problem solving which represent the minimal set of specialists capable of building and developing their own AI systems in any domain at a human or above human expert level.

Add tools as a code repo with all the available functions and something for searching and finding which one is useful.

Deep research - crawl + searXNG - also need upskill so use the teacher to craft a curriculum for AI to learn and commit to memory for autonomous training ing in some domain by generating synthetic questions from known data and then learning from errors with boosting.

Memento MDP tool for learning.

Learning, programming, governance and reflexivity tools as previously drafted- check what is in cursor repo on github??

Memory is episodic, semantic, relational or procedural.

Tools are as per original tool spec in system design.

Workflows include the hierarchical decomp with goal tree, the solver loop, the 8 primitives, linguistic precision duality, safety/ethics, constrained optimisation,

Is there a dependency on the chat question does it reference something like a previous message, shared context or other external info if so then need to get this before responding.

Calls to the specialist agents should be atomic I.e. conceivably answerable in a single LLM pass - otherwise they should be broken down.

Note that A2A and workflows + tool calling is absolutely necessary to get above human performance.

Build workflows before tooling.

Once the coding agent is up and running then we can start thinking about using this to write the system code.

Synthetic alignment with teacher forcing by using llm to predict results of running code to learn models of how to program.

Distillation of language natural language algos into code/smolagents.

Divergence/convergence workflow for creativity.

Workflows are design patterns implemented as higher order functions e.g. a chain of thought can take a list of prompts as input and auto gen the chain of thought. Or reflexivity can take constraints or success criteria needed to stop reflexive looping etc. They can be called as tools and are intermediate between fully agentic and fully programmed code. There is another axis which is the level to which control flow is delegated to the AI vs being specified by the programmer. Dependent on level of flexibility needed for a given process.

From examples, to patterns/models, to rules, to algorithms, to programs along the axis of being more heavily specified. Dependent on the degree to which a process is 'understood'.

1. Intelligence is compositional and the building blocks are finite.

2. There exist abstract axes along which all knowledge or processes can be defined and thus transformed into the framework of AI agents.

3. Scaling inference compute such that queries are atomic + providing RAG with information retrieval + tool calling for reasoning/inference is enough to achieve superhuman performance without retraining models.

4. We still need scaffolding to allow for search in AI as the primary means of knowledge discovery where the nodes are algorithms/workflows together with the execution of the algorithm with a view towards some higher goal that we are trying to achieve and a path is a series of algorithms together with their executions which leads to some final goal being accomplished and backtracking allows us to see exhibit an algorithmic trace that can be traversed to achieve that goal. This is a means of exploring the problem space - <https://www.geeksforgeeks.org/machine-learning/search-algorithms-in-ai/>

5. AI systems can train themselves autonomously if they are given the ability to understand their value function (as constraints on a solution to a problem) and have the ability to accurately evaluate the quality and accuracy of their output.

6. The system becomes much more powerful if you give it more observability (the ability to see itself, analyse itself and modify behaviour based on the self - that is, so it doesn't get stuck in a loop and so it learns from mistakes).

7. Memory = retrieval + storage of information.

8. Reasoning is best not done by an LLM - all reasoning should be done by first formalising something through examples and then executing code and the  reviewing the result of code where possible (STEM).

9. Learning is the process of translating memory and reasoning into actionable insight.

10. The language of reasoning is diagrams/flowcharts and it's formalisation  is category theory.

11. Workflows are tools which structure problem solving.

12. There are a finite number of tools which should be deployed from most to least specific which is applicable to the problem at hand. Where multiple tools do the same thing or approximately the same thing, we start can choose between the tools based on their relative quality, time and cost re LLM calls or compute.

ML for AGI is program synthesis not neural network based. With this insight and the above design considerations a system can learn explicit and not implicit representations and thus bypass the neural scaling laws or efficient compute frontier.