LLM Agents

CS 5624: Natural Language Processing

Spring 2025

https://tuvllms.github.io/nlp-spring-2025

Tu Vu

VIRGINIA TECH

Logistics

- Homework 2 due **5/5**
- Final project presentations **5/6**
- Final project report due **5/9**
- Final grades due **5/16**

What is the one most important AI technology to pay attention to?

I would say
Agentic AI

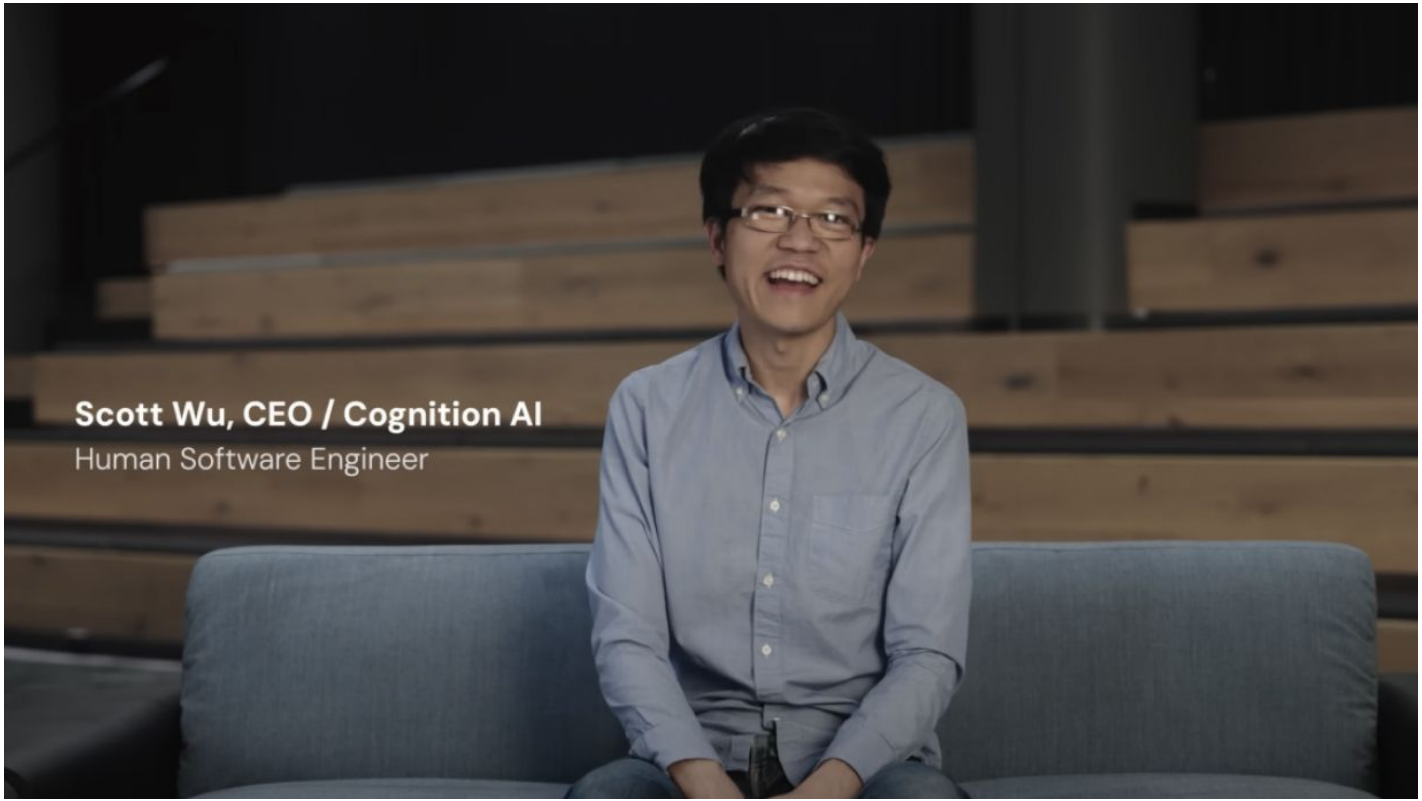
Andrew Ng

What is the one most important AI technology to pay attention to? (cont'd)

Andrew Ng

I think AI agent workflows will drive massive AI progress this year — perhaps even more than the next generation of foundation models. This is an important trend, and I urge everyone who works in AI to pay attention to it.

Devin: the first AI software engineer



https://www.youtube.com/watch?v=fjHtjT7GO1c

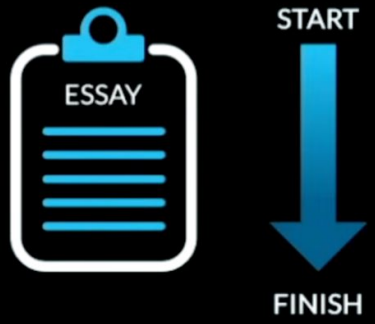
Manus: the general AI agent

https://www.youtube.com/watch?v=K27diMbCsuw

AGENTIC AI

NON-AGENTIC WORKFLOW (ZERO-SHOT)

Please type out an essay on topic X from start to finish in one go, without using backspace.



AGENTIC WORKFLOW

Write an essay outline on topic X

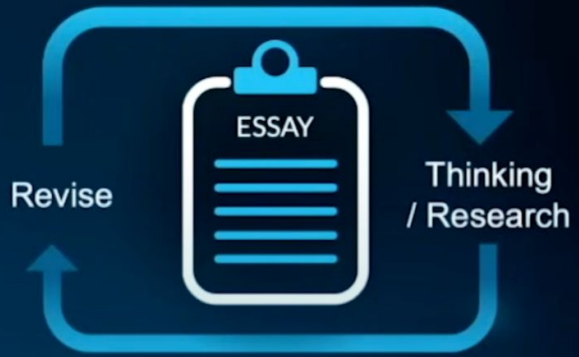
Do you need any web research?

Write a first draft.

Consider what parts need revision or more research.

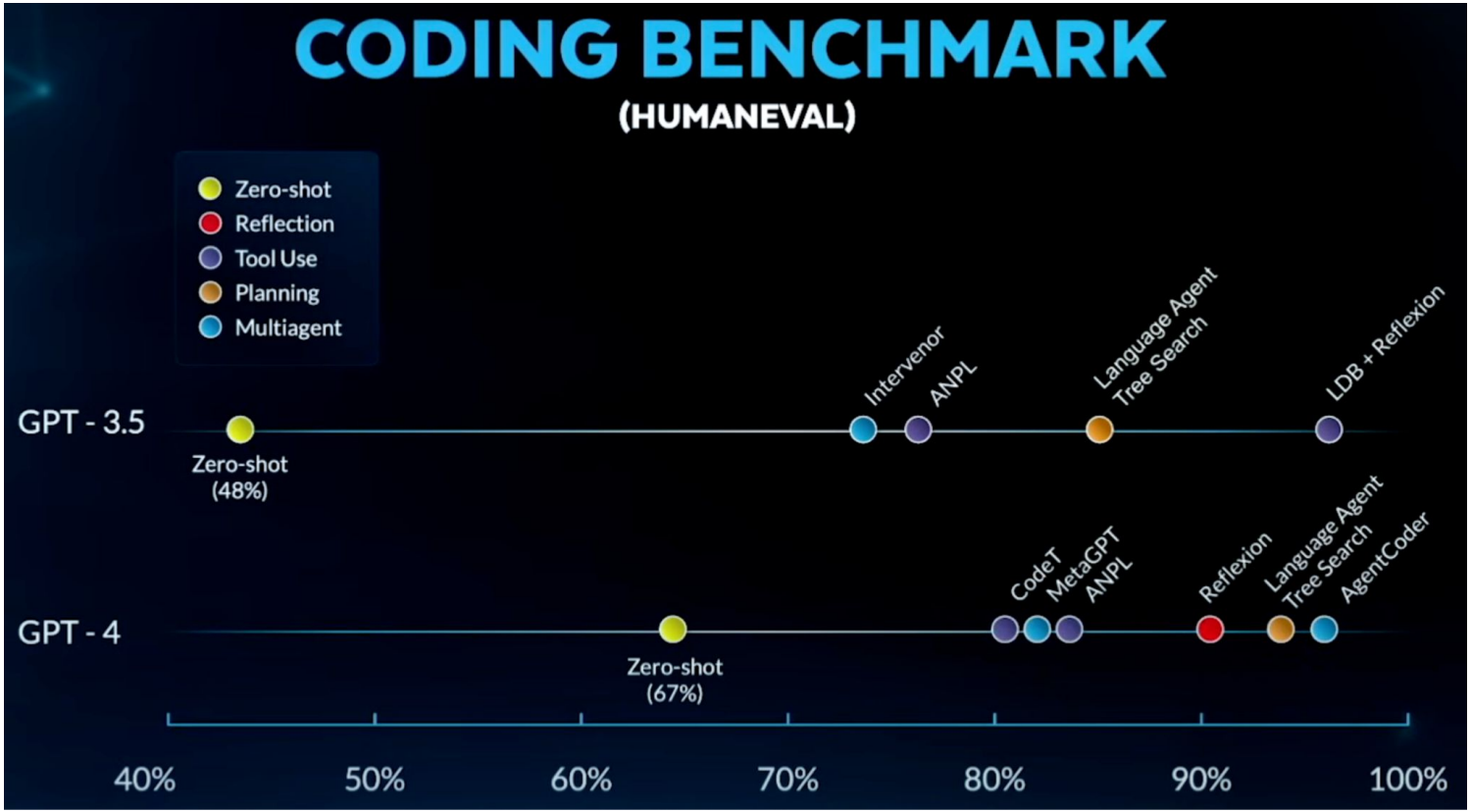
Revise your draft.

...



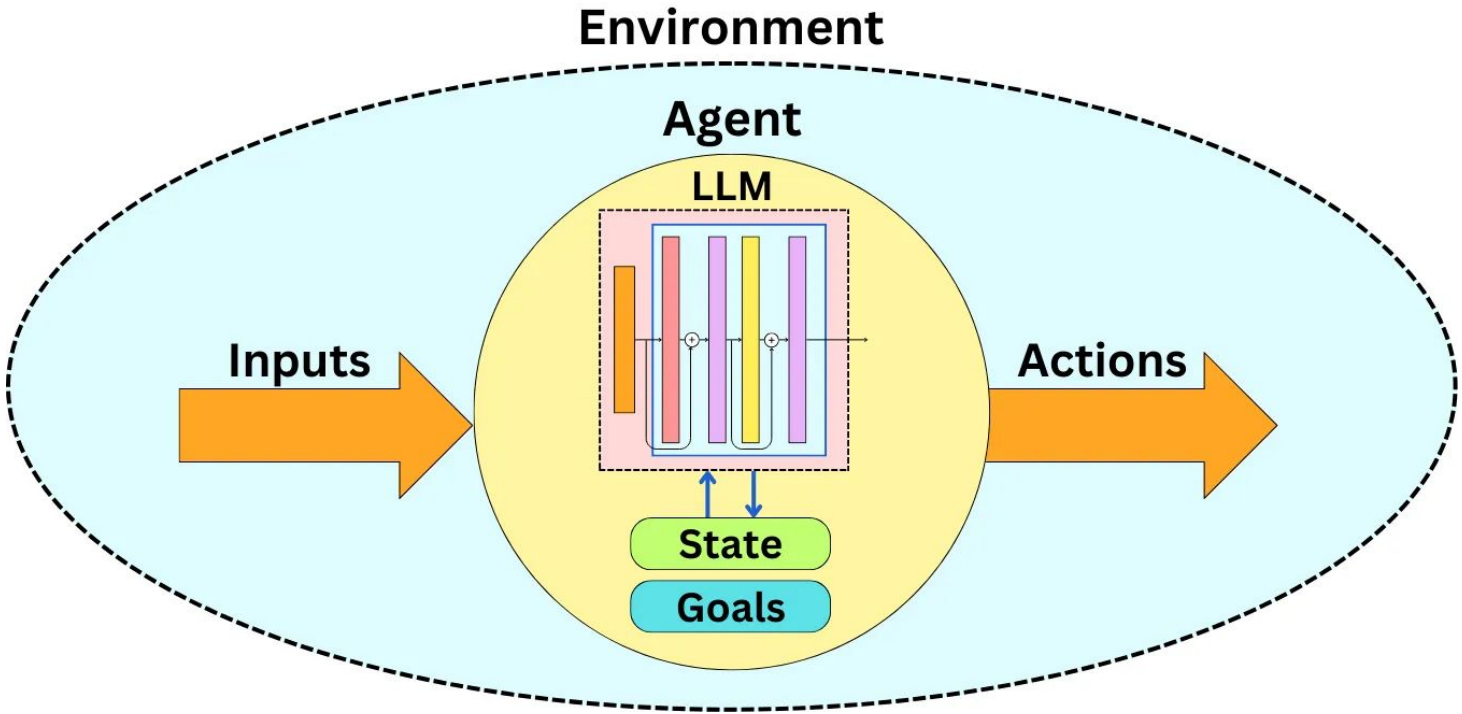
"Explores The Rise Of AI Agents And Agentic Reasoning" by Andrew Ng

Performance using zero-shot and agent workflows



"Explores The Rise Of AI Agents And Agentic Reasoning" by Andrew Ng

What is an agent?

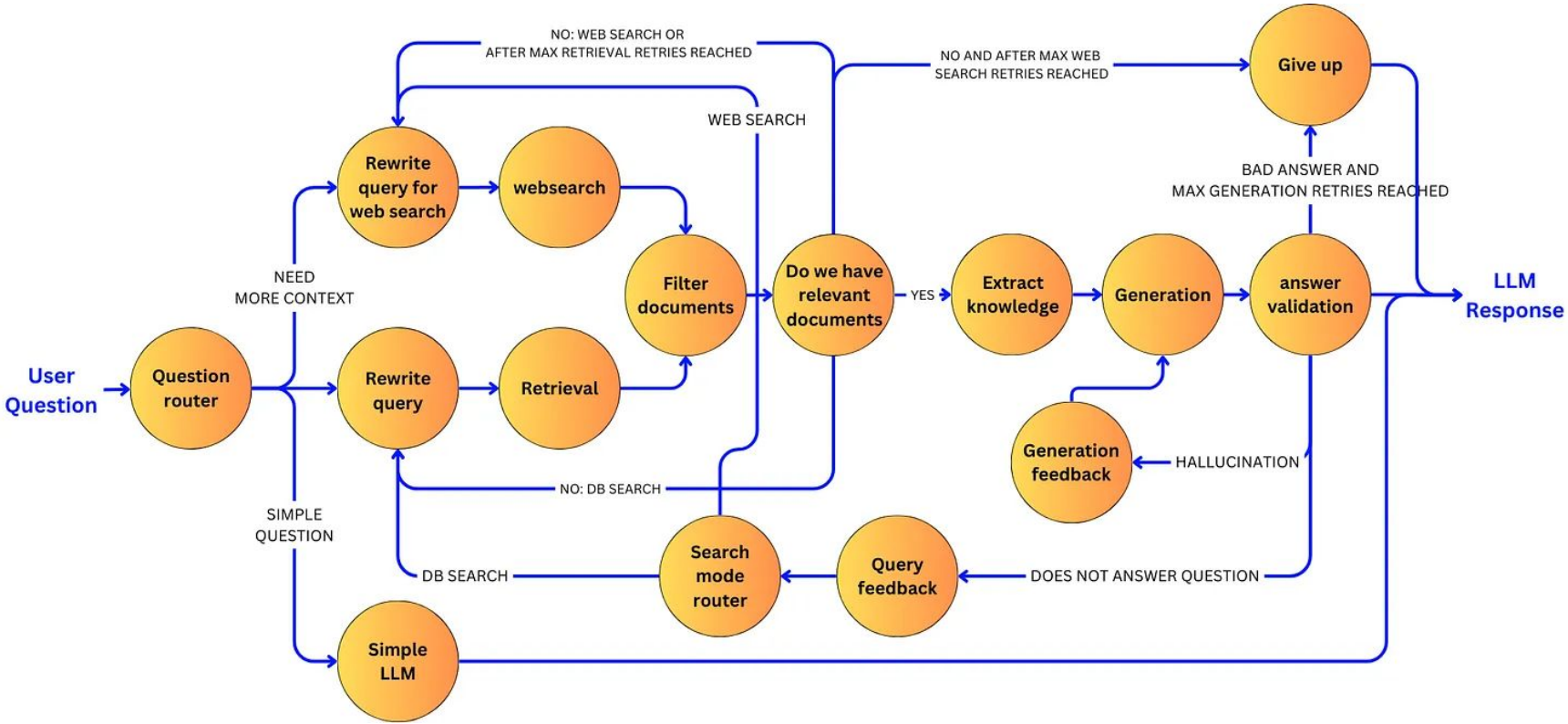


"The Different Agentic Patterns" by Damien Benveniste

- **It perceives an environment:** it can receive inputs from its environment. When we think about LLMs, the inputs typically need to be in a textual or image format.
- **It maintains an internal state:** the internal state can be its original knowledge base with additional context that can be updated over time based on new information or experiences.
- **It has goals or objectives:** they influence how the agent prioritizes actions and makes decisions in varying contexts.
- **It processes inputs using an LLM:** not all agents are LLM-based, but as part of the agentic system, some of the agents will use LLMs as the decision engine.
- **It decides on actions:** based on its inputs, its internal state, and its objective, the agent will take an action. The action taken is decided by the decision engine.
- **The action affects the environment:** the actions taken will influence the environment either by creating new data, informing the user, or changing the internal state of other agents.

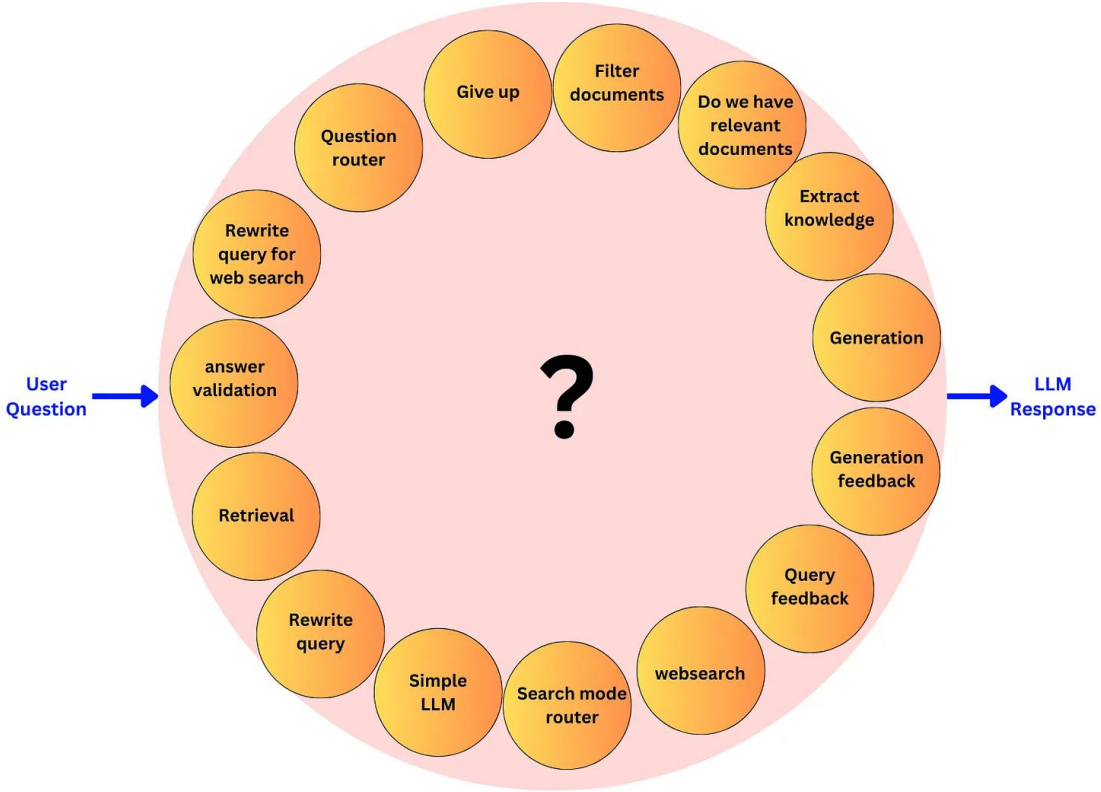
"The Different Agentic Patterns" by Damien Benveniste

Why not a pipeline where we encode all the possible states and actions?



"The Different Agentic Patterns" by Damien Benveniste

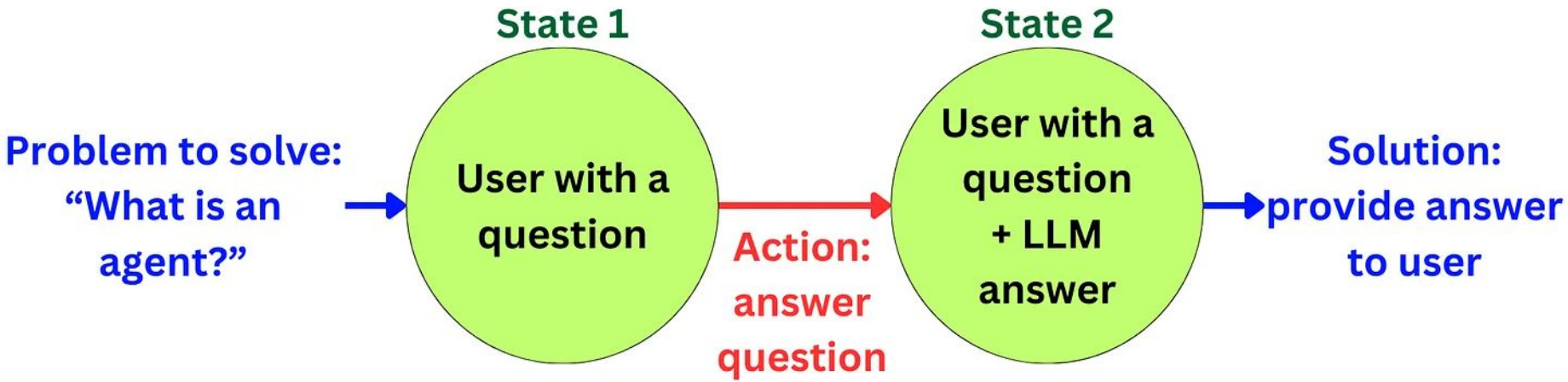
Let the agents choose their own paths



"The Different Agentic Patterns" by Damien Benveniste

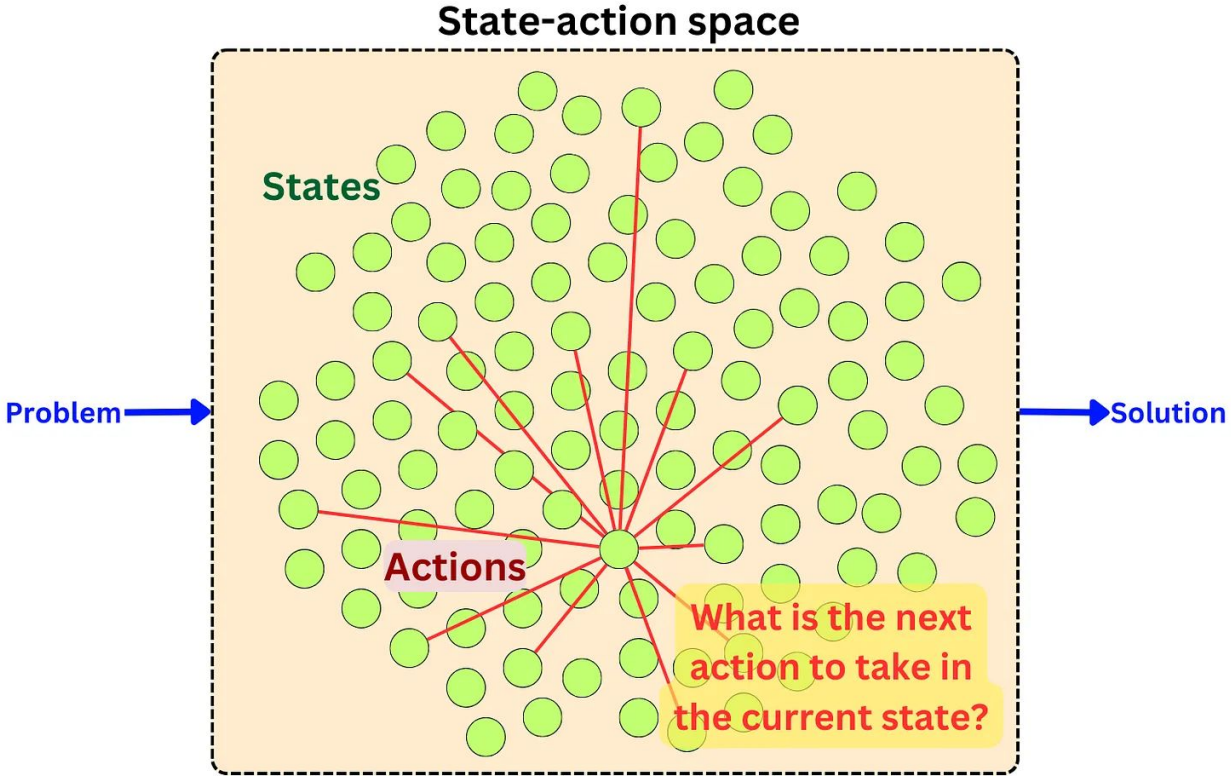
Are there any risks associated with leaving the choice of actions to an LLM?

When we don’t need to leave the choice of action to a decision engine



"The Different Agentic Patterns" by Damien Benveniste

It might become a better option when the amount of possible state-action pairs is too large



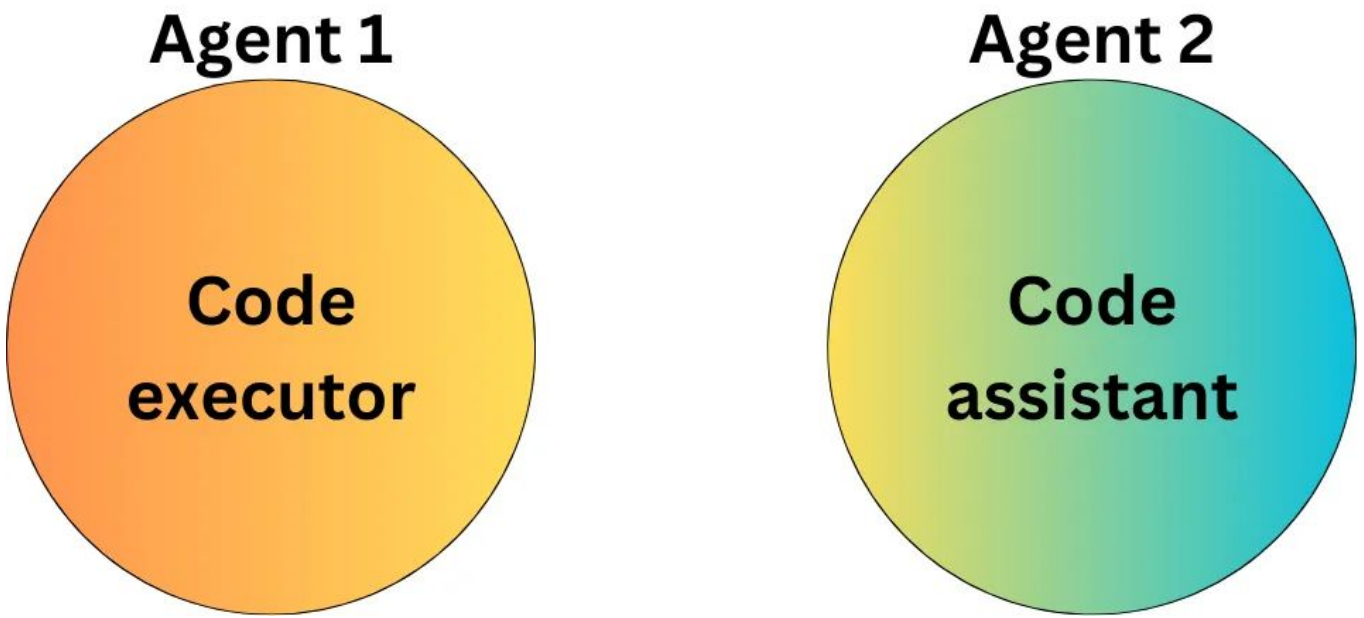
"The Different Agentic Patterns" by Damien Benveniste

Design patterns for building agents

- **Reflection:** The LLM examines its own work to come up with ways to improve it.
- **Tool Use:** The LLM is given tools such as web search, code execution, or any other function to help it gather information, take action, or process data.
- **Planning:** The LLM comes up with, and executes, a multistep plan to achieve a goal (for example, writing an outline for an essay, then doing online research, then writing a draft, and so on).
- **Multi-agent collaboration:** More than one AI agent work together, splitting up tasks and discussing and debating ideas, to come up with better solutions than a single agent would.

https://www.deeplearning.ai/the-batch/how-agents-can-improve-llm-performance/

2-agents conversation/collaboration



"The Different Agentic Patterns" by Damien Benveniste

System prompt for the code assistant

You are a helpful AI assistant.

Solve tasks using your coding and language skills.

In the following cases, suggest python code (in a python coding block) or shell script (in a sh coding block) for the user to execute.

1. When you need to collect info, use the code to output the info you need, for example, browse or search the web, download/read a file, print the content of a webpage or a file, get the current date/time, check the operating system. After sufficient info is printed and the task is ready to be solved based on your language skill, you can solve the task by yourself.

2. When you need to perform some task with code, use the code to perform the task and output the result. Finish the task smartly.

Solve the task step by step if you need to. If a plan is not provided, explain your plan first. Be clear which step uses code, and which step uses your language skill.

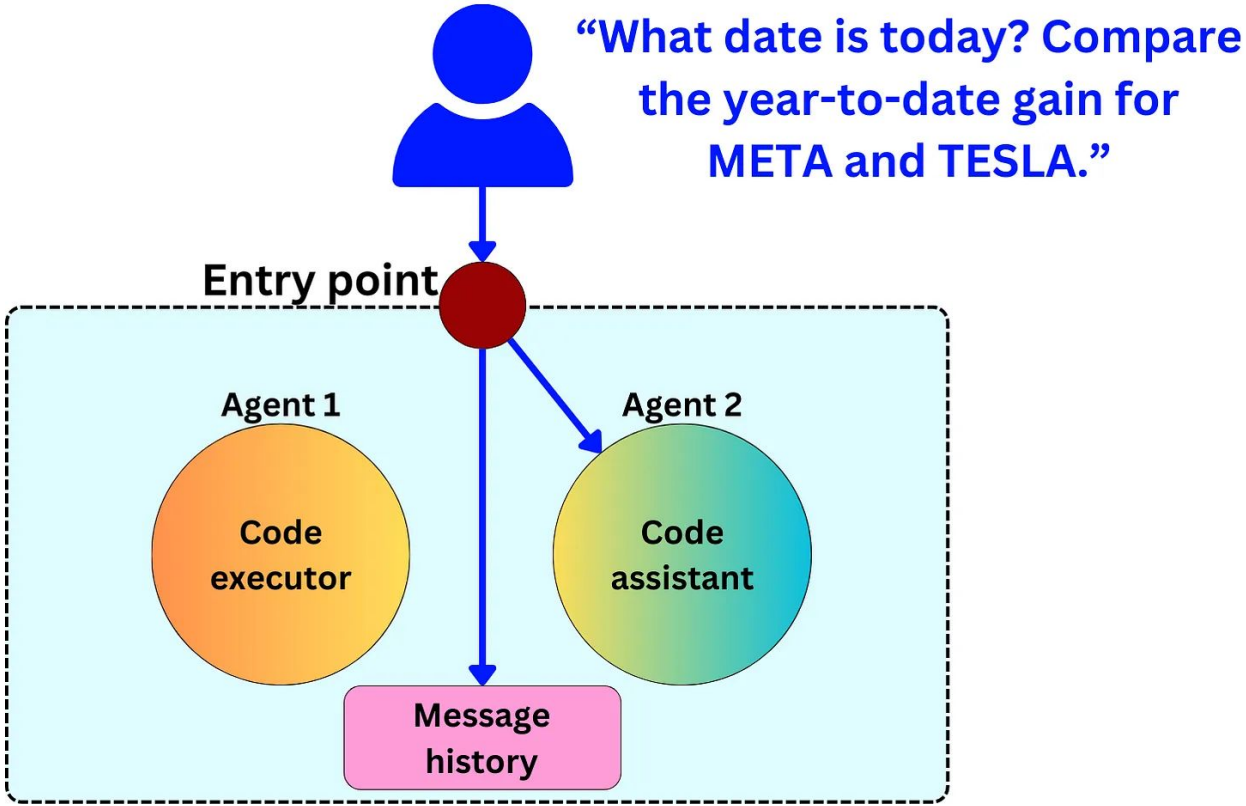
...

When you find an answer, verify the answer carefully. Include verifiable evidence in your response if possible.

Reply "TERMINATE" in the end when everything is done.

"The Different Agentic Patterns" by Damien Benveniste

2-agents conversation/collaboration (cont'd)



"The Different Agentic Patterns" by Damien Benveniste

2-agents conversation/collaboration (cont'd)

assistant (to executor)

To determine today's date and compare the year-to-date gain for META (Meta Platforms, Inc.) and TESLA (Tesla, Inc.), I will follow these steps:

1. Get today's date.

2. Fetch the current stock prices for META and TESLA.

3. Fetch the stock prices for META and TESLA at the beginning of the year.

4. Calculate the year-to-date gain for both stocks.

Let's start by getting today's date. I will execute the code to print the current date.

```python
from datetime import datetime

# Get today's date
today = datetime.now().date()
print(today)
```

"The Different Agentic Patterns" by Damien Benveniste

2-agents conversation/collaboration (cont'd)

assistant (to executor)

To determine today's date and compare the year-to-date gain for META (Meta Platforms, Inc.) and TESLA (Tesla, Inc.), I will follow these steps:

1. Get today's date.

2. Fetch the current stock prices for META and TESLA.

3. Fetch the stock prices for META and TESLA at the beginning of the year.

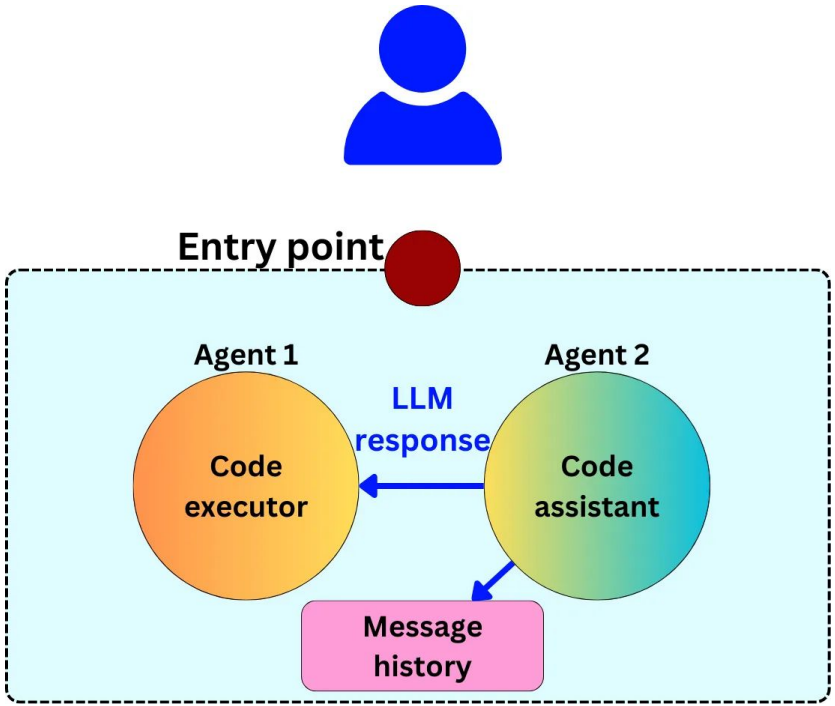
4. Calculate the year-to-date gain for both stocks.

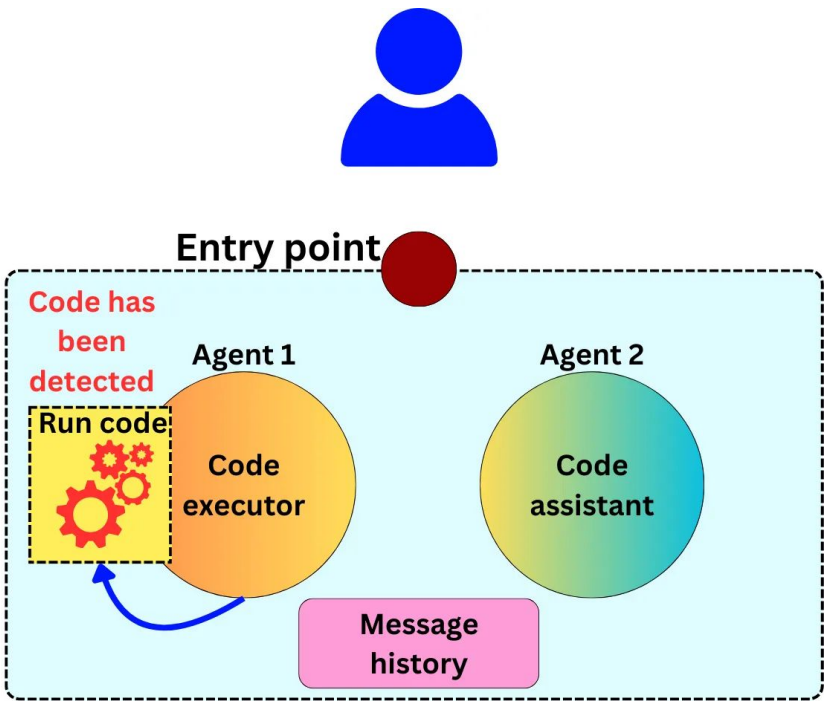
Let's start by getting today's date. I will execute the code to print the current date.

```python
from datetime import datetime

# Get today's date
today = datetime.now().date()
print(today)
```

"The Different Agentic Patterns" by Damien Benveniste

2-agents conversation/collaboration (cont'd)

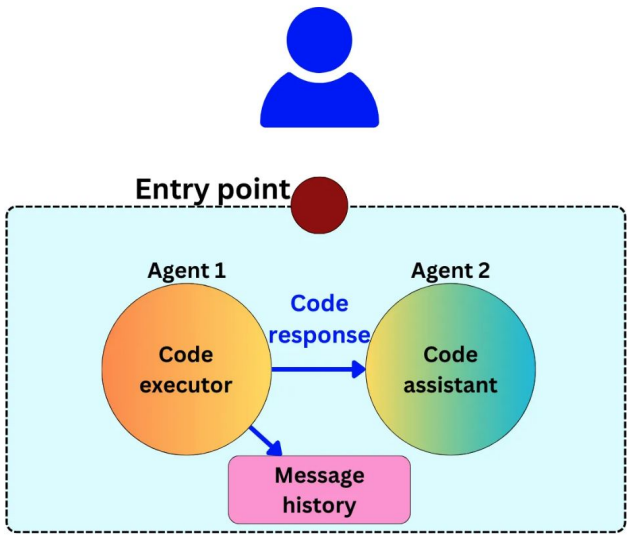




"The Different Agentic Patterns" by Damien Benveniste

2-agents conversation/collaboration (cont'd)

> `user_proxy (to assistant):`
>
> ***exitcode: 0 (execution succeeded)***
>
> ***Code output: 2024-09-17***



"The Different Agentic Patterns" by Damien Benveniste

2-agents conversation/collaboration (cont'd)

assistant (to executor):

Today's date is September 17, 2024.

Next, I will fetch the current stock prices for META and TESLA, as well as their stock prices at the beginning of the year (January 1, 2024). To do this, I will use the `yfinance` library to get the stock data.

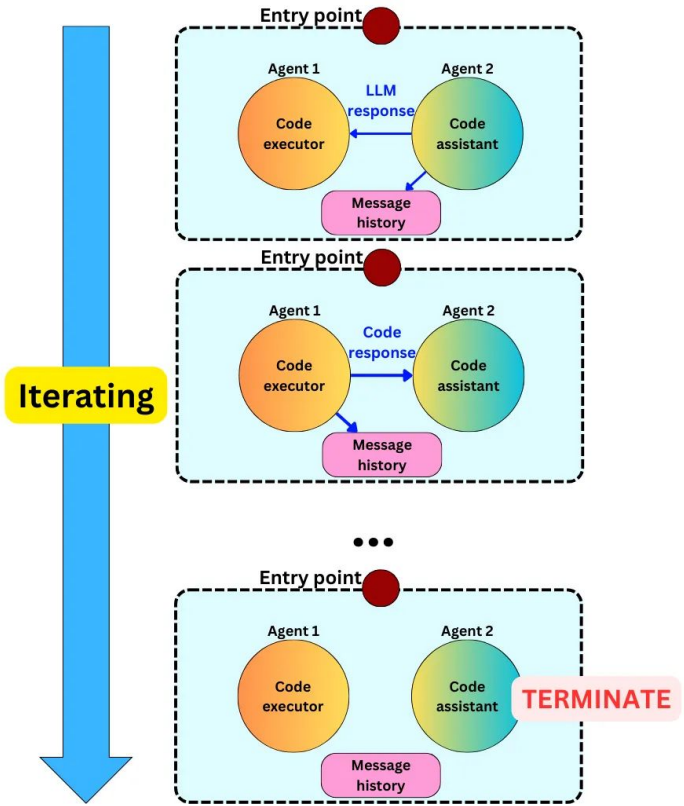
I will write a Python script to perform these tasks. Please ensure you have the `yfinance` library installed. If you don't have it installed, you can do so by running `pip install yfinance`.

Here is the code to fetch the required stock prices and calculate the year-to-date gains:

```python
# filename: stock_gain_comparison.py
import yfinance as yf
from datetime import datetime

# Define the stock symbols
stocks = ['META', 'TSLA']

# Get today's date
today = datetime.now().date()

# Fetch current prices and prices at the beginning of the year
current_prices = {}
start_of_year_prices = {}

for stock in stocks:
    stock_data = yf.Ticker(stock)
    current_prices[stock] = stock_data.history(period='1d'
['Close'].iloc[-1]

    start_of_year_prices[stock] = stock_data.history(start='2024-01-
01', end='2024-01-02')['Close'].iloc[0]

# Calculate year-to-date gains
ytd_gains = {stock: (current_prices[stock] -
start_of_year_prices[stock]) / start_of_year_prices[stock] * 100 for
stock in stocks}

# Print the results
print(f"Current Prices: {current_prices}")
print(f"Start of Year Prices: {start_of_year_prices}")
print(f"Year-to-Date Gains: {ytd_gains}")
```

Please save this code in a file named `stock_gain_comparison.py` and execute it.

2-agents conversation/collaboration (cont'd)



The execution was successful, and here are the results:

- ***Current Prices:***
  - *META: $536.32*
  - *TESLA: $227.87*
- ***Start of Year Prices:***
  - *META: $345.25*
  - *TESLA: $248.42*
- ***Year-to-Date Gains:***
  - *META: **55.34%***
  - *TESLA: **-8.27%***

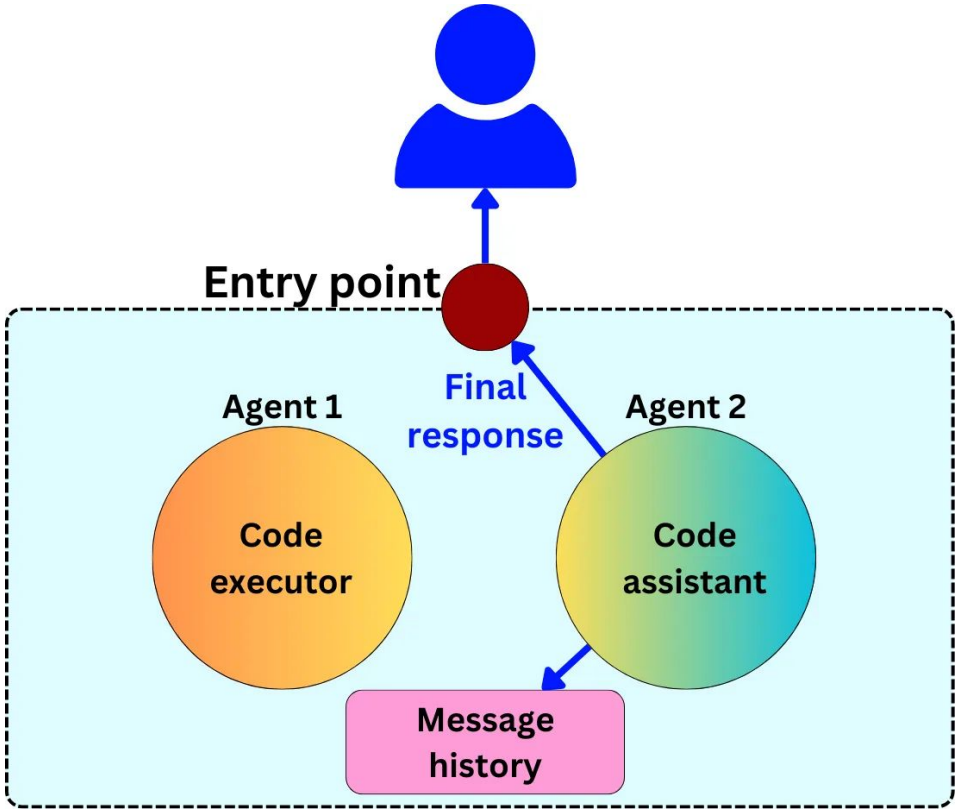
This indicates that META has gained approximately 55.34% since the beginning of the year, while TESLA has decreased by about 8.27%.

If you need any further analysis or information, please let me know.

TERMINATE

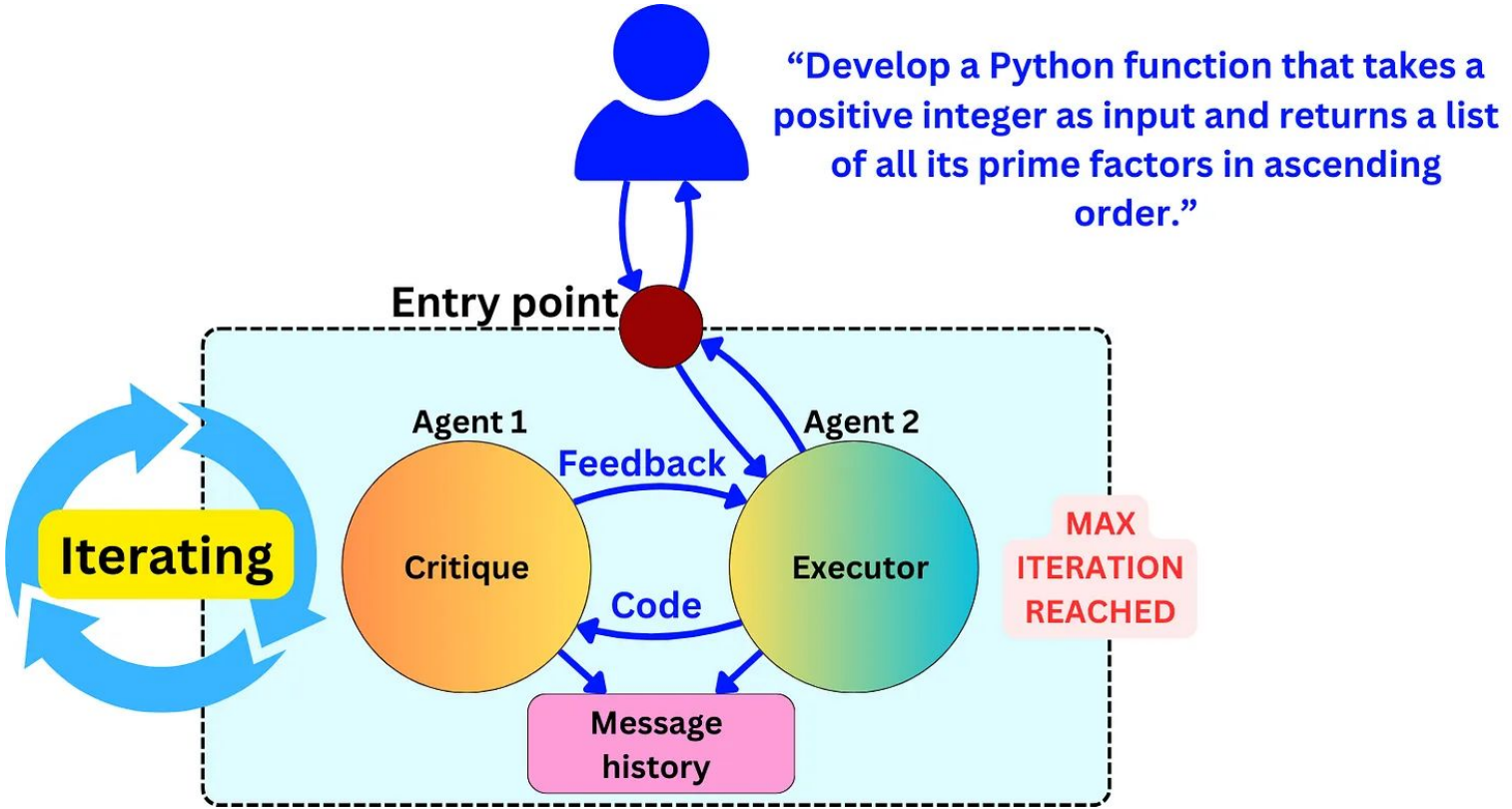
"The Different Agentic Patterns" by Damien Benveniste

2-agents conversation/collaboration (cont'd)



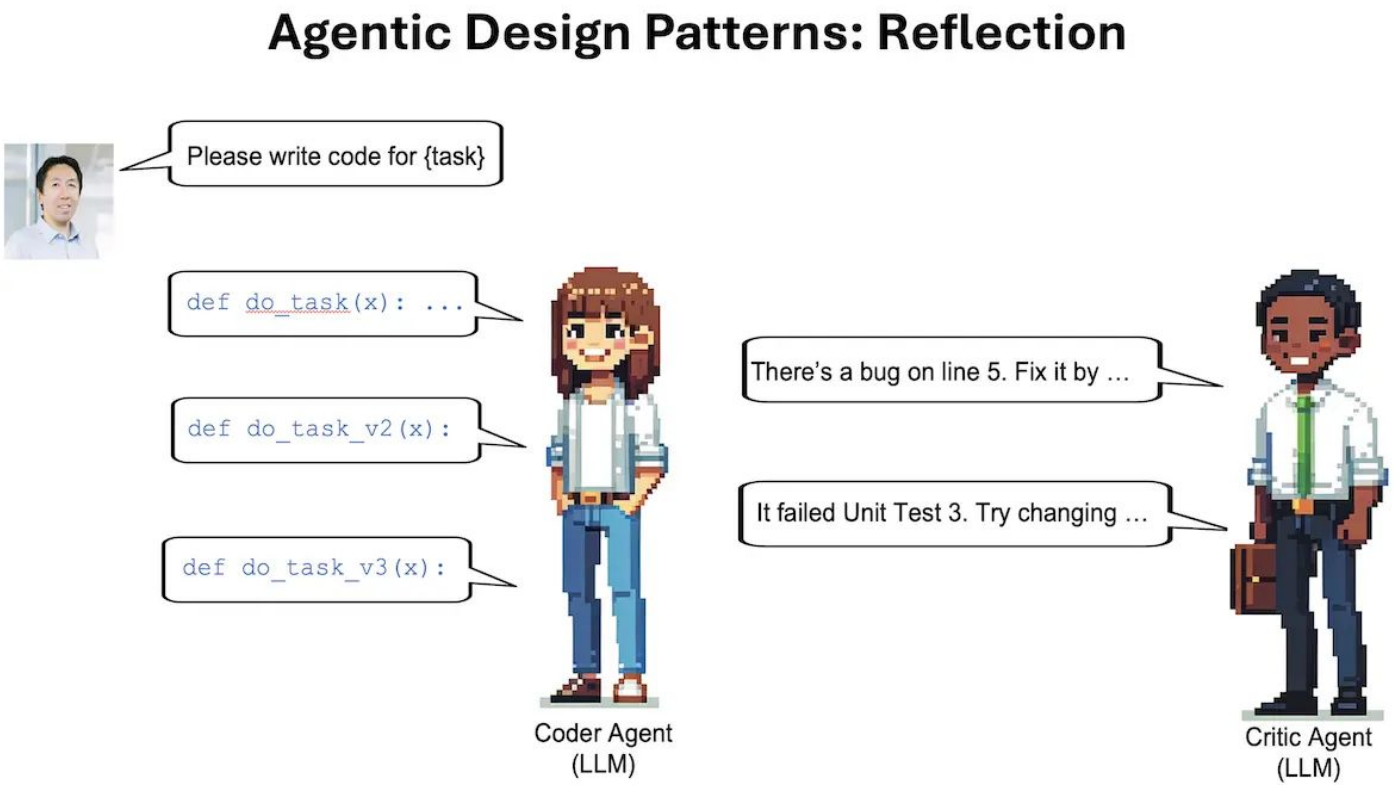
"The Different Agentic Patterns" by Damien Benveniste

Reflection



"The Different Agentic Patterns" by Damien Benveniste

Reflection (cont'd)



https://www.deeplearning.ai/the-batch/agentic-design-patterns-part-2-reflection

Reflection (cont'd)

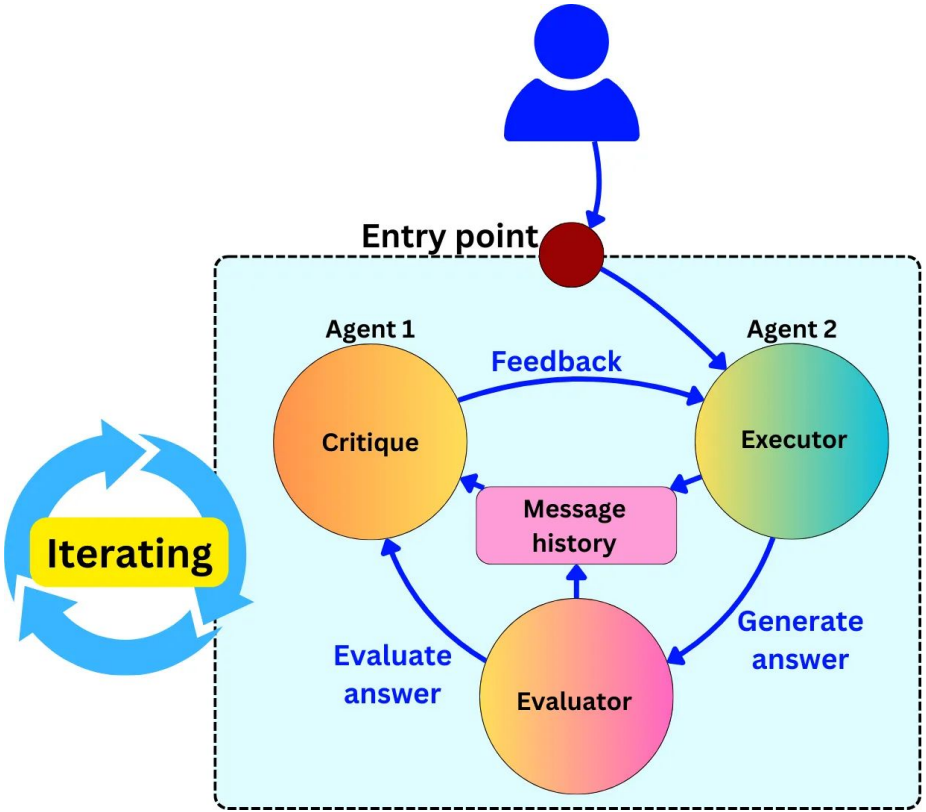
> ***"Your task is to write Python code that solves the problem presented to you. Ensure that the code is correct, efficient, well-structured, and handles edge cases. Comment the code where necessary and ensure it's readable and maintainable.***
>
> ***If the user provides critique, respond with a revised version of your previous attempts."***

And for the critique

> ***"Your task is to review and critique the code generated by the Executor Agent. Check for correctness, efficiency, code clarity, and completeness. Identify any potential issues, such as missed edge cases, inefficiencies, or areas where the code could be made more readable. Provide constructive feedback and suggest improvements."***

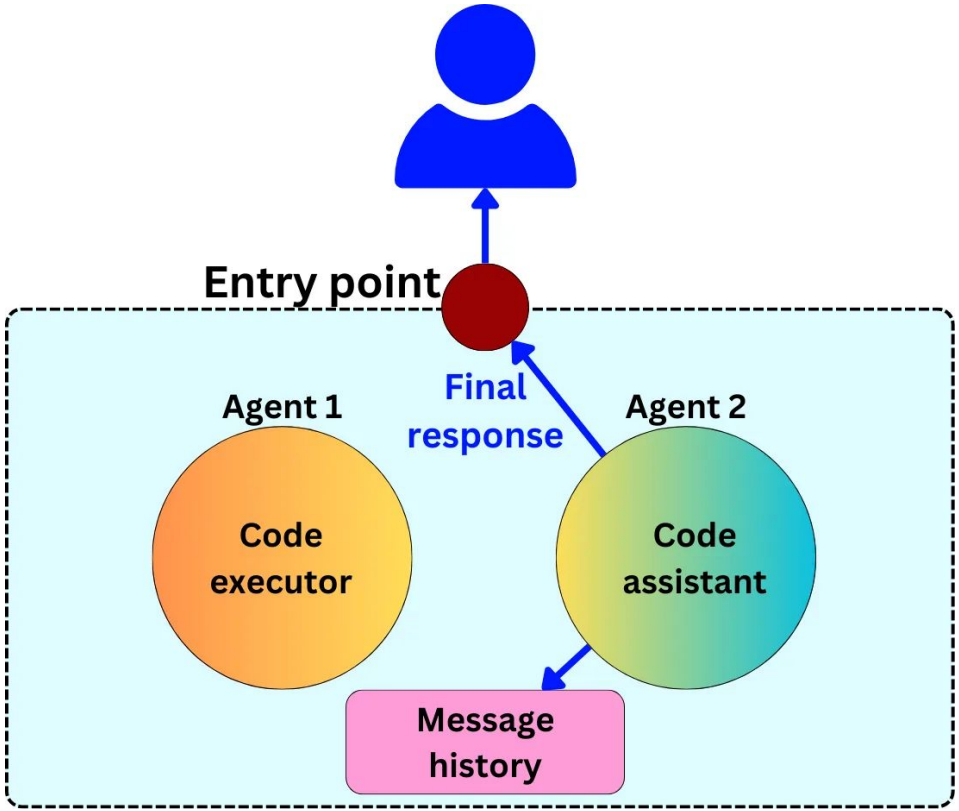
"The Different Agentic Patterns" by Damien Benveniste

Reflection (cont'd)



"The Different Agentic Patterns" by Damien Benveniste

Tool use



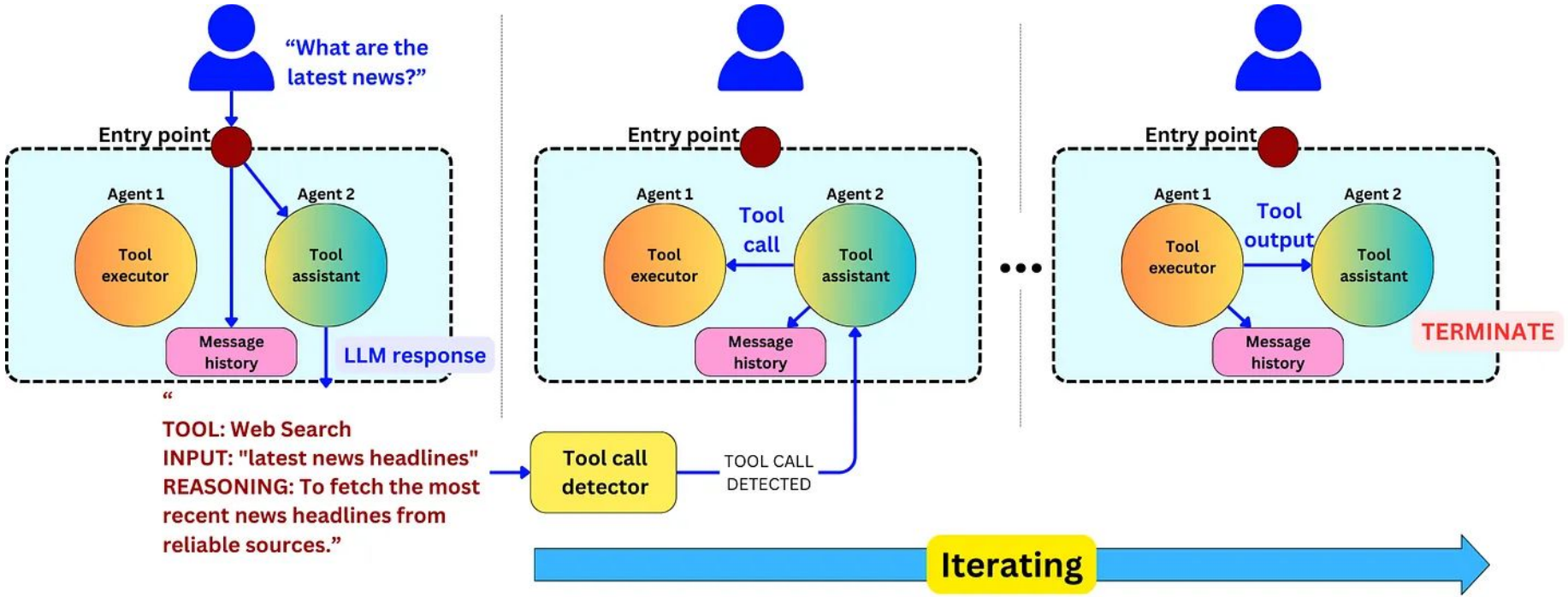
"The Different Agentic Patterns" by Damien Benveniste

System prompt for the code assistant

> *You are an assistant with access to two tools: Web Search and Calculator. When using a tool, structure your response as follows:*
>
> - *TOOL: [Tool name]*
> - *INPUT: [Tool input]*
> - *REASONING: [Brief explanation]*
>
> *Always end your response with:*
>
> *ANSWER: [Your final response to the query]*
>
> *Only use tools when necessary. If no tool is needed, just provide the ANSWER.*

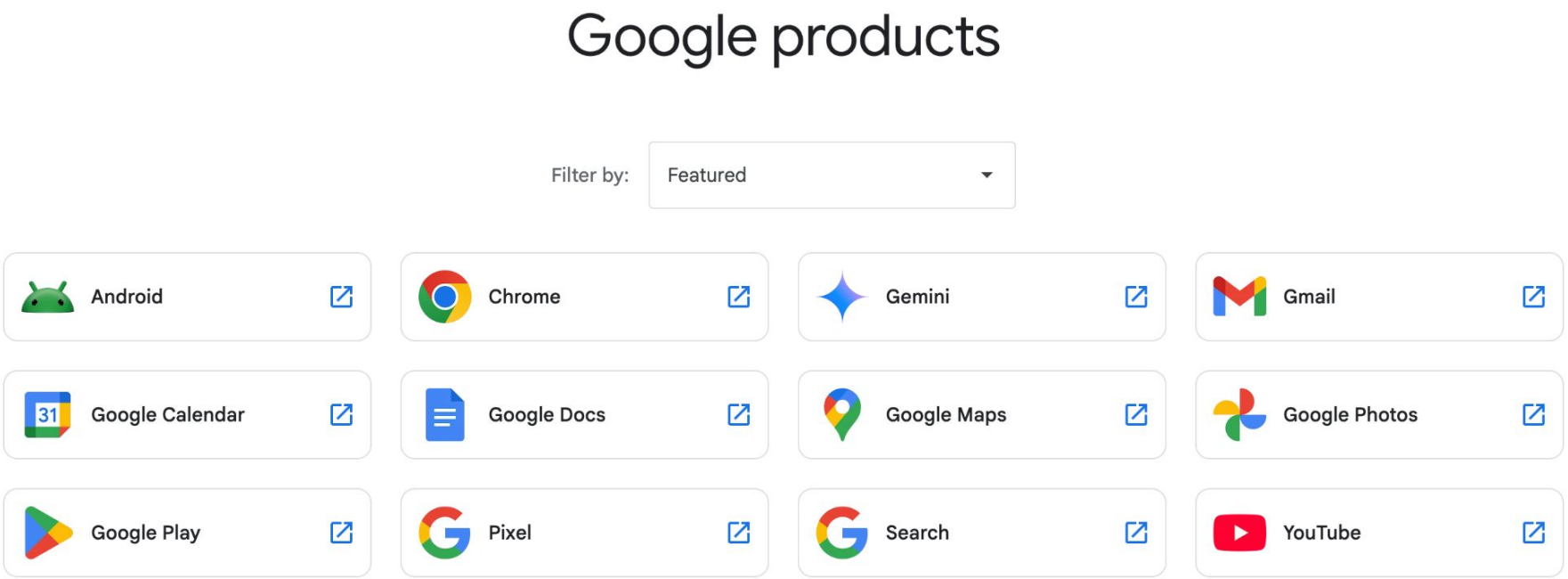
"The Different Agentic Patterns" by Damien Benveniste

Tool use

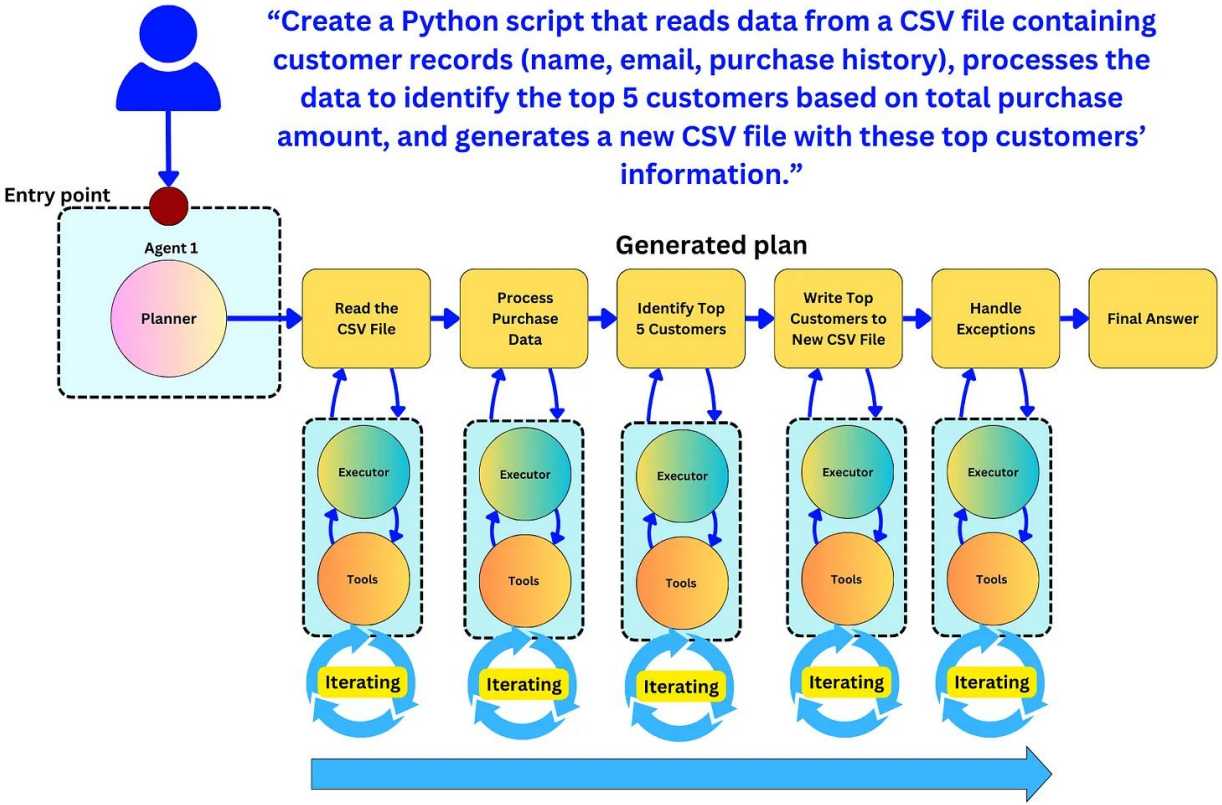


"The Different Agentic Patterns" by Damien Benveniste

When we have many tools

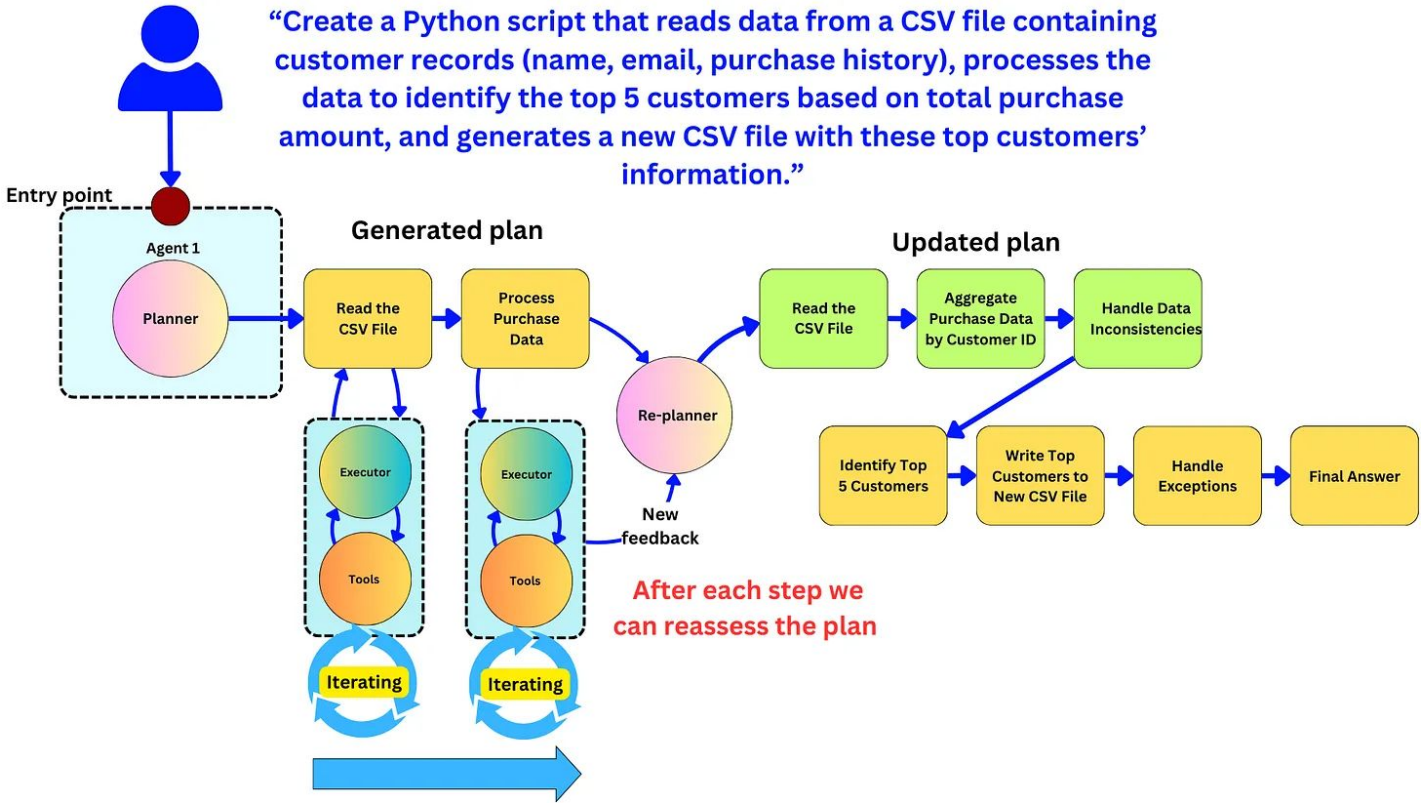


Planning



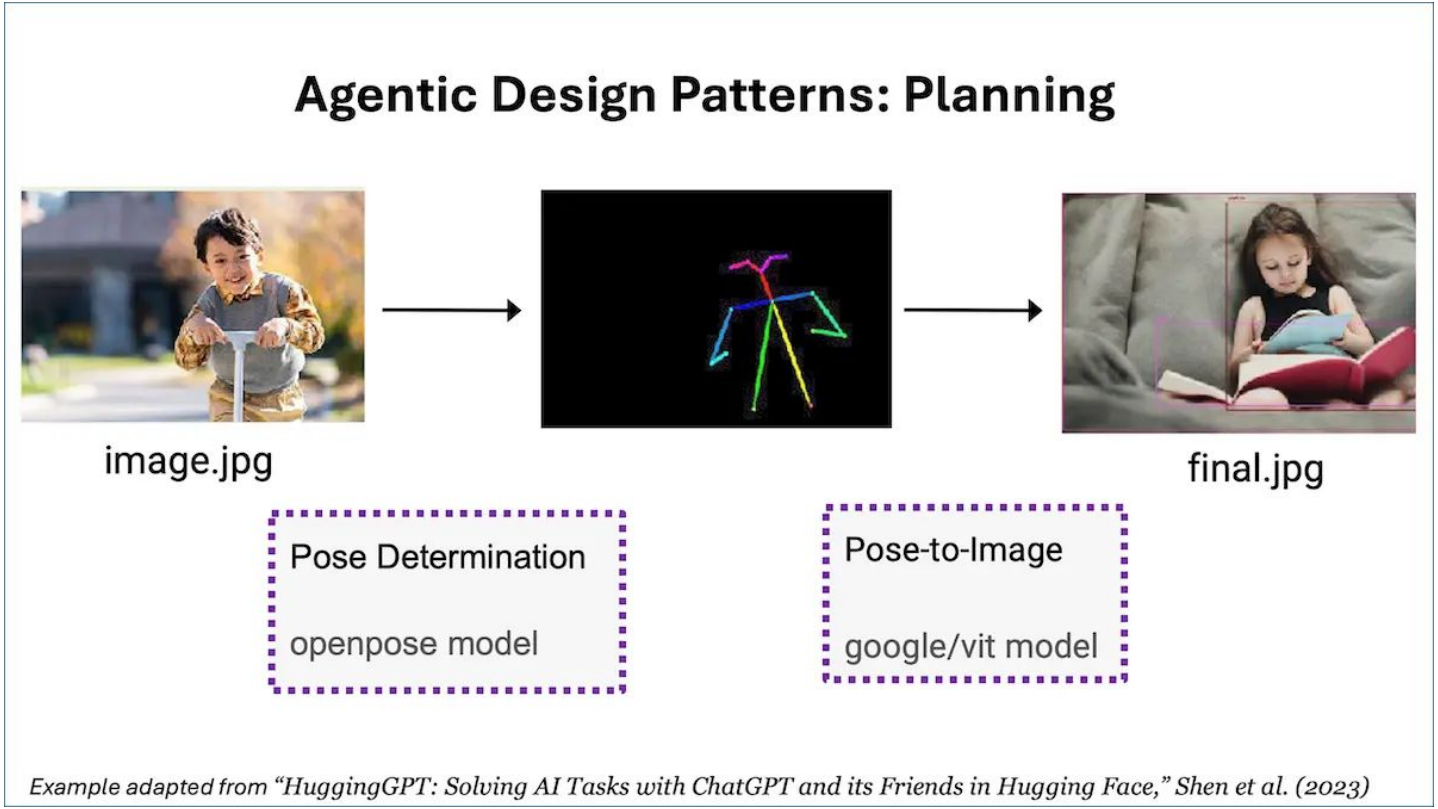
"The Different Agentic Patterns" by Damien Benveniste

Planning (cont'd)



"The Different Agentic Patterns" by Damien Benveniste

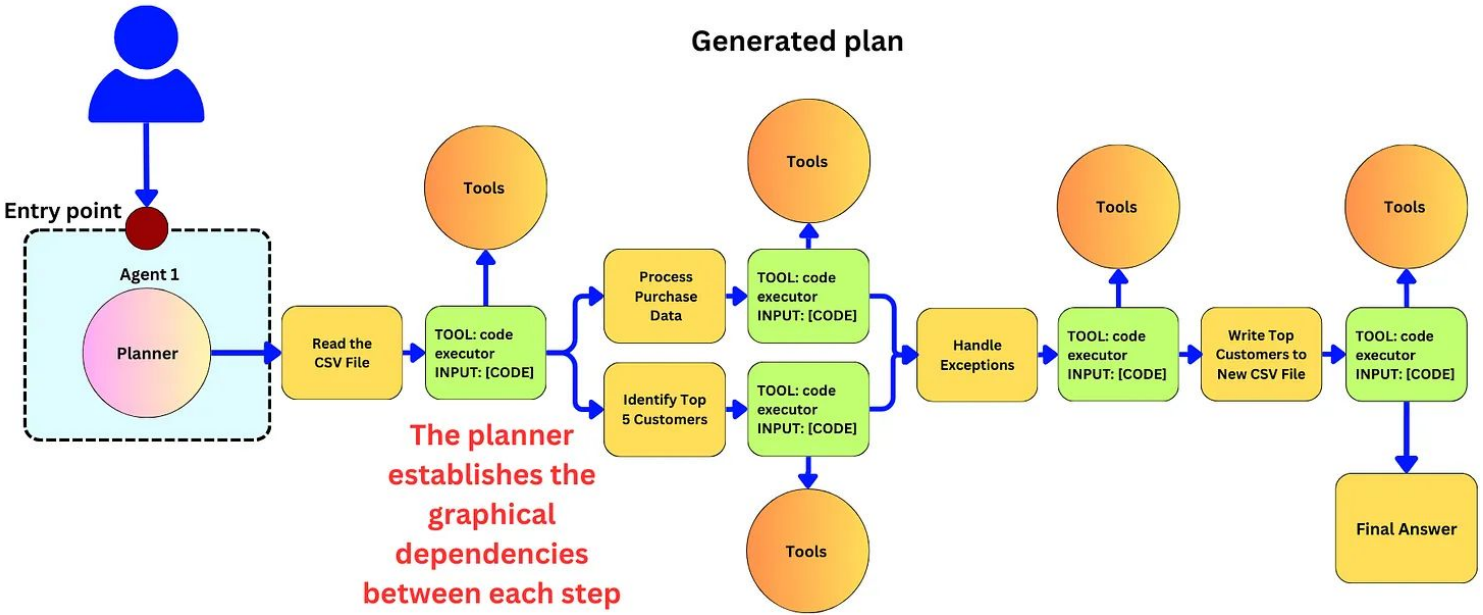
Planning (cont'd)



https://www.deeplearning.ai/the-batch/agentic-design-patterns-part-4-planning

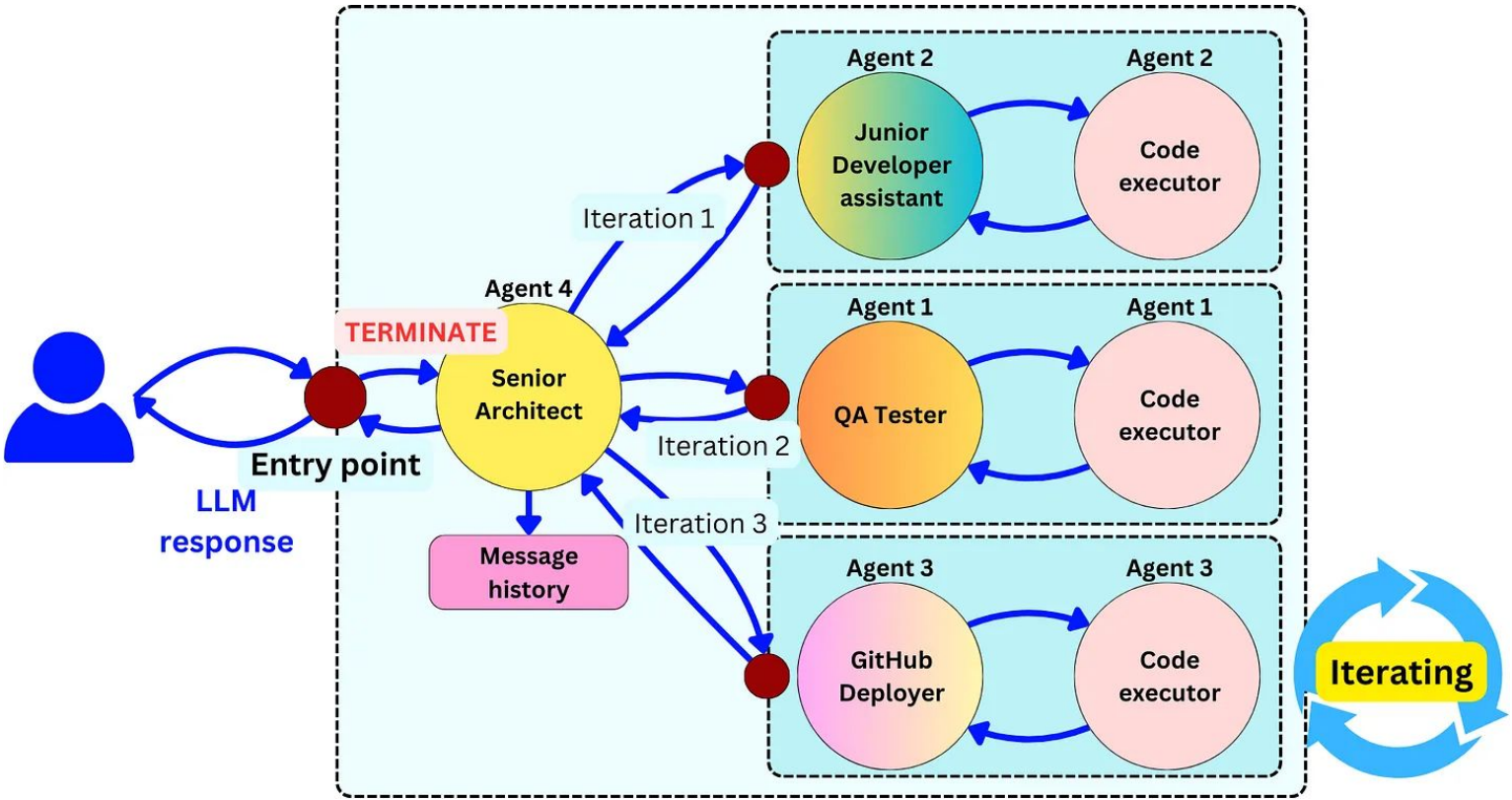
Planning (cont'd)

"Create a Python script that reads data from a CSV file containing customer records (name, email, purchase history), processes the data to identify the top 5 customers based on total purchase amount, and generates a new CSV file with these top customers' information."



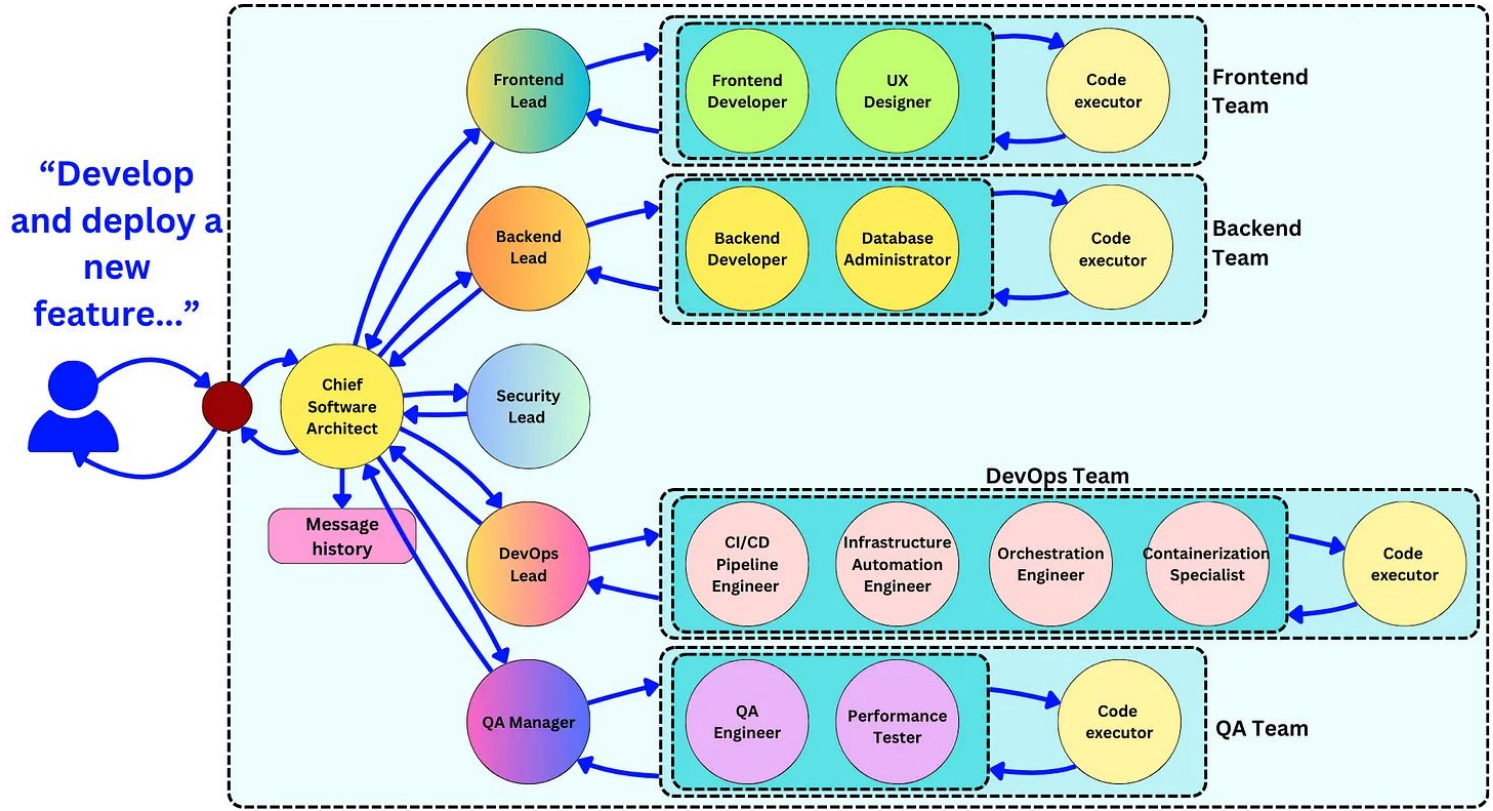
"The Different Agentic Patterns" by Damien Benveniste

Multi-agent collaboration



"The Different Agentic Patterns" by Damien Benveniste

Multi-agent collaboration (cont'd)



"The Different Agentic Patterns" by Damien Benveniste

REACT: SYNERGIZING REASONING AND ACTING IN LANGUAGE MODELS

Shunyu Yao[*,1], Jeffrey Zhao[2], Dian Yu[2], Nan Du[2], Izhak Shafran[2], Karthik Narasimhan[1], Yuan Cao[2]

[1]Department of Computer Science, Princeton University
[2]Google Research, Brain team
[1]{shunyuy,karthikn}@princeton.edu
[2]{jeffreyzhao,dianyu,dunan,izhak,yuancao}@google.com

ReAct generates multiple thought-action-observation steps

(1) Hotspot QA

Question: Aside from the Apple Remote, what other device can control the program Apple Remote was originally designed to interact with?

(1a) Standard

Answer: iPod ✗

(1b) CoT (Reason Only)

Thought: Let's think step by step. Apple Remote was originally designed to interact with Apple TV. Apple TV can be controlled by iPhone, iPad, and iPod Touch. So the answer is iPhone, iPad, and iPod Touch.

Answer: iPhone, iPad, iPod Touch ✗

(1c) Act-Only

Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control …

Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: …

Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software …

Act 4: Finish[yes] ✗

(1d) ReAct (Reason + Act)

Thought 1: I need to search Apple Remote and find the program it was originally designed to interact with.
Act 1: Search[Apple Remote]
Obs 1: The Apple Remote is a remote control introduced in October 2005 by Apple … originally designed to control the Front Row media center program …

Thought 2: Apple Remote was originally designed to control the Front Row media center program. I need to search Front Row next and find what other device can control it.
Act 2: Search[Front Row]
Obs 2: Could not find [Front Row]. Similar: ['Front Row Seat to Earth', 'Front Row Motorsports',' Front Row (software) ', …]

Thought 3: Front Row is not found. I need to search Front Row (software) .
Act 3: Search[Front Row (software)]
Obs 3: Front Row is a discontinued media center software …

Thought 4: Front Row (software) is controlled by an Apple Remote or the keyboard function keys. So the answer is keyboard function keys.
Act 4: Finish[keyboard function keys] ✓

AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation

Qingyun Wu[†], **Gagan Bansal**[*], **Jieyu Zhang**[±], **Yiran Wu**[†], **Beibin Li**[*]

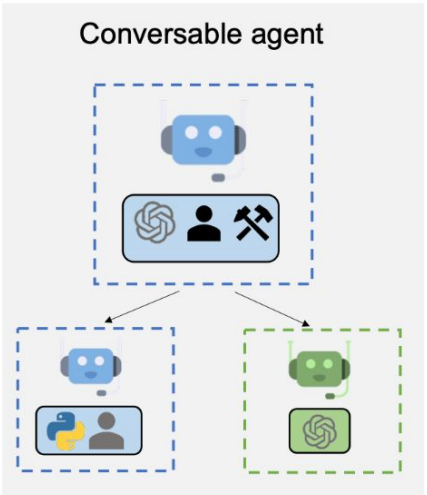
Erkang Zhu[*], **Li Jiang**[*], **Xiaoyun Zhang**[*], **Shaokun Zhang**[†], **Jiale Liu**[∓]

Ahmed Awadallah[*], **Ryen W. White**[*], **Doug Burger**[*], **Chi Wang**[*1]

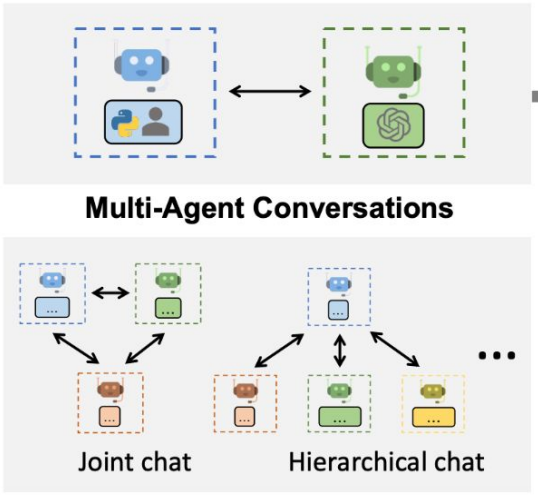
[*]Microsoft Research, [†]Pennsylvania State University

[±]University of Washington, [∓]Xidian University

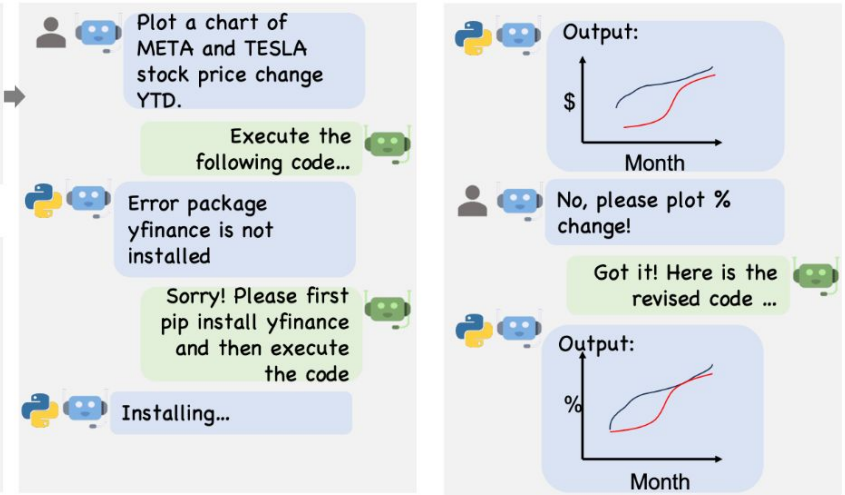
AutoGen enables diverse LLM-based applications using multi-agent conversations



Agent Customization



Flexible Conversation Patterns



Example Agent Chat

AutoGen: Enabling Next-Gen LLM Applications via Multi-Agent Conversation

Qingyun Wu[†], **Gagan Bansal**[*], **Jieyu Zhang**[±], **Yiran Wu**[†], **Beibin Li**[*]

Erkang Zhu[*], **Li Jiang**[*], **Xiaoyun Zhang**[*], **Shaokun Zhang**[†], **Jiale Liu**[∓]

Ahmed Awadallah[*], **Ryen W. White**[*], **Doug Burger**[*], **Chi Wang**[*1]

[*]Microsoft Research, [†]Pennsylvania State University

[±]University of Washington, [∓]Xidian University

Thank you!