2024/09/05 #StudyCo

LangGraphでの Human-in-the-Loopの実装

自己紹介

大嶋勇樹（おおしまゆうき）

https://x.com/oshima_123

株式会社ジェネラティブエージェンツ取締役CTO/Co-founder
大規模言語モデルを組み込んだアプリケーションやAIエージェントの開発を実施
個人ではエンジニア向けの勉強会開催やUdemy講座の作成など
勉強会コミュニティStudyCo運営

「ChatGPT/LangChainによるチャットシステム構築［実践］入門」（共著）

株式会社ジェネラティブエージェンツ - 会社概要

AIエージェントが「ハブ」となり
人間とAIエージェントの協働が
当たり前になる世界を実現する

AIエージェントによるBPaaSの提供

複雑な業務管理を自律的におこなうLLMエージェントサービスを提供します。独自のワークフローエンジンと複数プロフィールを設定したマルチエージェントシステムが御社のワークフォースとして業務を遂行します。

生成AIを活用したソフトウェア開発支援

AIエージェントの開発技術をコアに、生成AIを活用したソフトウェア開発をレンタルCAIO（最高AI責任者）として支援します。

会社名	株式会社ジェネラティブエージェンツ （英文：Generative Agents, Inc.）
所在地	東京都港区 ※ 全社員リモート勤務
役員構成	CEO 西見 公宏 COO 吉田 真吾 CTO 大嶋 勇樹
設立年月	2024年3月14日
事業内容	AIエージェントによるBPaaSの提供／生成AIを活用したソフトウェア開発／技術顧問、AIエージェントに関わるコミュニティ運営、法人向けコンサルティング

運営メンバー

代表取締役CEO / Founder

西見 公宏 Masahiro Nishimi

事業会社の顧問CTOとして活動するソフトウェア開発のスペシャリスト。AIエージェントを経営に導入することにより、あらゆる業種業態の生産性を高めるための活動に尽力している。

「その仕事、AIエージェントがやっておきました。――ChatGPTの次に来る自律型AI革命」（技術評論社）単著、Software Design「実践LLMアプリケーション開発」（技術評論社）連載。

主な著書

『その仕事、AIエージェントがやっておきました』

取締役COO / Co-founder

吉田 真吾 Shingo Yoshida

AWS Serverless Heroとして日本におけるサーバーレスの普及を促進。「ChatGPT/LangChainによるチャットシステム構築［実践］入門」（技術評論社）共著、「Azure OpenAI ServiceではじめるChatGPT/LLMシステム構築入門」（技術評論社）共著、「AWSによるサーバーレスアーキテクチャ」（翔泳社）監修、「サーバーレスシングルページアプリケーション」（オライリー）監訳、「AWSエキスパート養成読本」（技術評論社）共著。ChatGPT Community（JP）主催

主な著書

『ChatGPT/LangChainによるチャットシステム構築［実践］入門』
『Azure OpenAI ServiceではじめるChatGPT/LLMシステム構築入門』
エンジニア選書』

取締役CTO / Co-founder

大嶋 勇樹 Yuki Oshima

大規模言語モデルを組み込んだアプリケーションやAIエージェントの開発を実施。
個人ではエンジニア向けの勉強会開催や教材作成など。オンラインコースUdemyではベストセラー講座多数。

「ChatGPT/LangChainによるチャットシステム構築［実践］入門」（技術評論社）共著。勉強会コミュニティStudyCo運営。

主な著書

『ChatGPT/LangChainによるチャットシステム構築
［実践］入門』

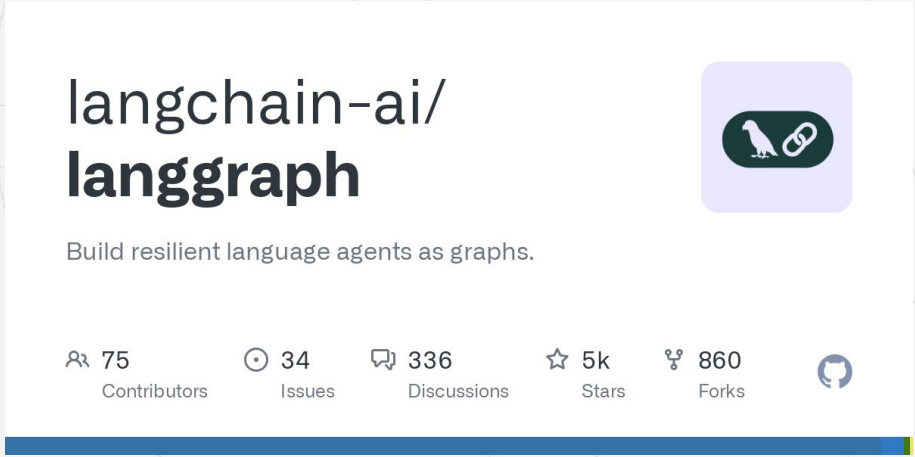
2024/09/05 #StudyCo

LangGraphでの Human-in-the-Loopの実装

LangGraphとは

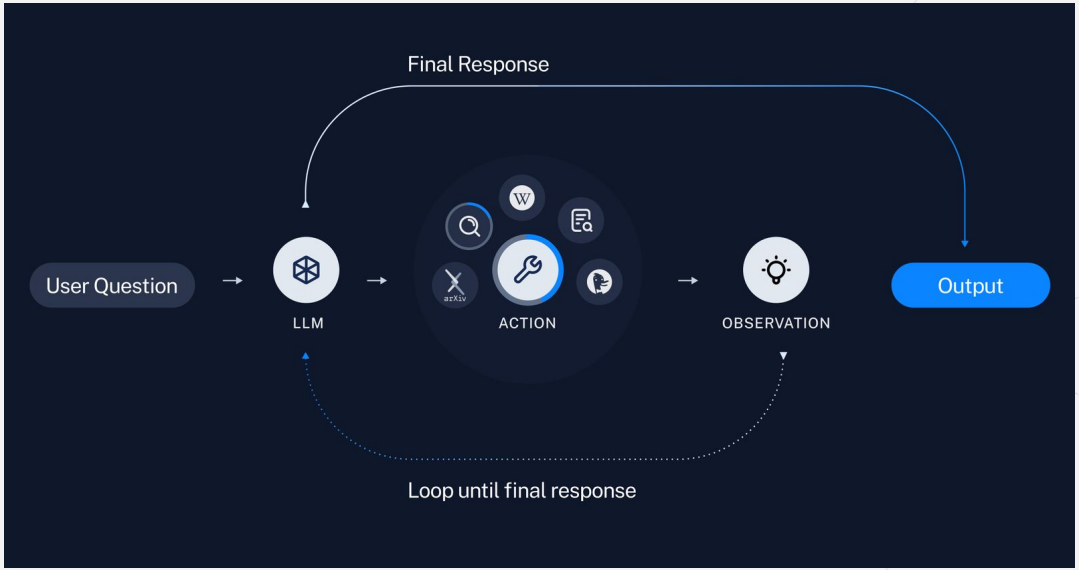
LangGraphは、LangChain社が提供するLLMエージェントを実装するためのパッケージです

LangChain・LangSmithとシームレスに統合されています



エージェントの処理の基本的な流れ

シンプルなLLMエージェントは、LLMによる推論・ツールの使用（アクション）・観察を繰り返し、ユーザの入力から最終的な応答まで下図の流れで動作します



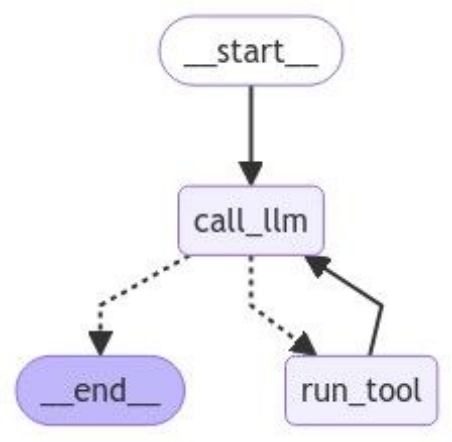
OpenAI's Bet on a Cognitive Architecture

https://blog.langchain.dev/openais-bet-on-a-cognitive-architecture/

LangGraphでのエージェントの実装

LangGraphでは、「ノード」と「エッジ」を定義してワークフローのグラフを作成して実行します
シンプルなLLMエージェントのグラフは以下のように定義できます

```
builder = StateGraph(MessagesState)
builder.add_node("call_llm", self._call_llm)
builder.add_node("run_tool", self._run_tool)
builder.add_edge(START, "call_llm")
builder.add_conditional_edges("call_llm", self._route_after_llm)
builder.add_edge("run_tool", "call_llm")
self.graph = builder.compile()
```



Human-in-the-Loopを入れるとしたら...

- **人間によるツールの利用の承認**

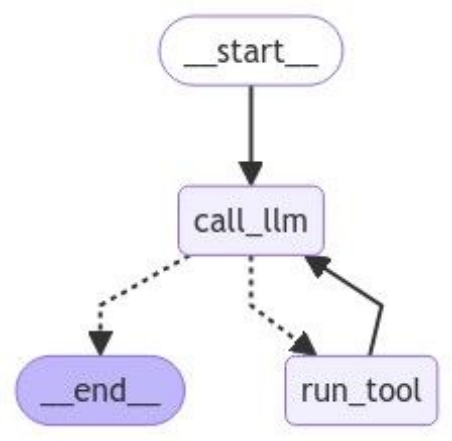
  LLMがツールを使いたいと判断した際に、人間の承認を得るようにする

- **人間によるツールの引数の修正**

  LLMが生成したツールの引数を、人間が修正可能にする

- **人間に質問する**

  ツールに加えて、LLMが人間に質問できるようにする



人間による承認を実装した例

https://github.com/os1ma/langgraph-human-in-the-loop-demo

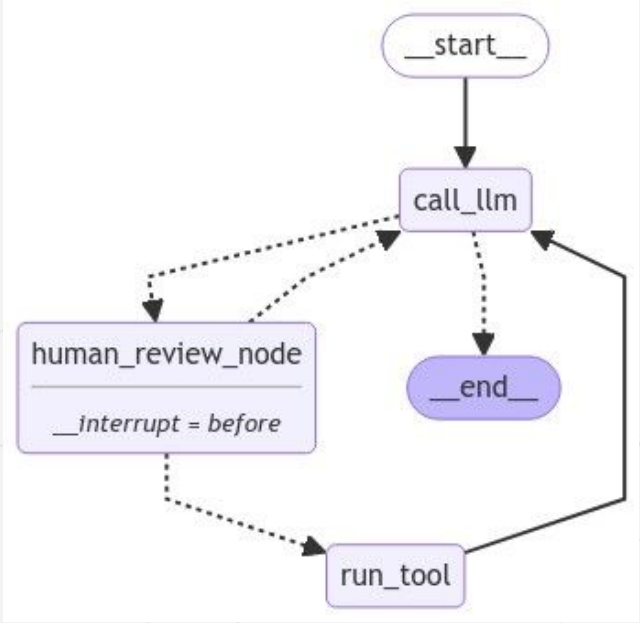
LangGraphでの人間による承認の実装

LangGraphで人間による承認を実装する例は、以下のようになります

```
builder = StateGraph(HumanInTheLoopAgentState)
builder.add_node("call_llm", self._call_llm)
builder.add_node("run_tool", self._run_tool)
builder.add_node("human_review_node", self._human_review_node)
builder.add_edge(START, "call_llm")
builder.add_conditional_edges("call_llm", self._route_after_llm)
builder.add_conditional_edges("human_review_node", self._route_after_human)
builder.add_edge("run_tool", "call_llm")

memory = MemorySaver()

self.graph = builder.compile(
    checkpointer=memory,
    interrupt_before=["human_review_node"],
)
```



human_review_nodeが追加されていることに加えて、

「checkpointer」「interrupt_before」の2つがポイントです

checkpointer

checkpointerは、LangGraphでのグラフの実行状況を保存する機能です

```
memory = MemorySaver()

self.graph = builder.compile(
    checkpointer=memory,
    interrupt_before=["human_review_node"],
)
```

- MemorySaver：インメモリで保存する
- SQLiteSaver：SQLiteに保存する
- PostgresSaver：PostgreSQLに保存する
- カスタム：BaseCheckpointSaverを継承して独自に実装可能

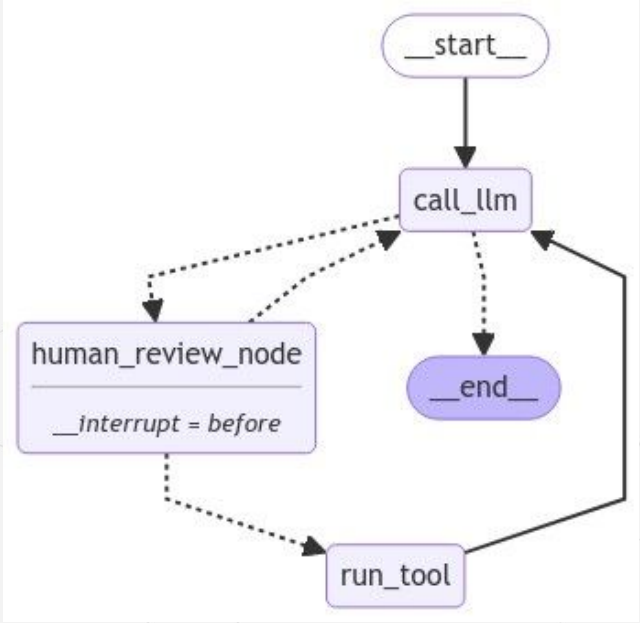
公式ドキュメント：https://langchain-ai.github.io/langgraph/how-tos/persistence/

interrupt_before

グラフの構築時に「interrupt_before」を設定すると、指定したノードの手前で処理が一時停止します

```
self.graph = builder.compile(
    checkpointer=memory,
    interrupt_before=["human_review_node"],
)
```

処理が一時停止したタイミングで承認ボタンを表示して、
承認されたら処理の続きを進めることができます
（どこまで処理したかはcheckpointerで保存されています）



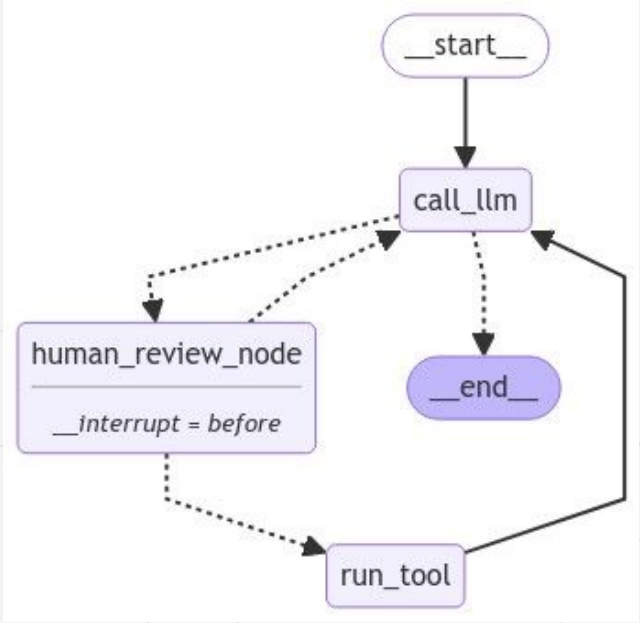
（再掲）LangGraphでの人間による承認の実装

LangGraphで人間による承認を実装する例は、以下のようになります

```
builder = StateGraph(HumanInTheLoopAgentState)
builder.add_node("call_llm", self._call_llm)
builder.add_node("run_tool", self._run_tool)
builder.add_node("human_review_node", self._human_review_node)
builder.add_edge(START, "call_llm")
builder.add_conditional_edges("call_llm", self._route_after_llm)
builder.add_conditional_edges("human_review_node", self._route_after_human)
builder.add_edge("run_tool", "call_llm")

memory = MemorySaver()

self.graph = builder.compile(
    checkpointer=memory,
    interrupt_before=["human_review_node"],
)
```



human_review_nodeが追加されていることに加えて、

「checkpointer」「interrupt_before」の2つがポイントです

ちなみに...

承認する代わりに人間から追加の指示があったケースには、

ツールの呼び出しに失敗したメッセージをStateに追加することで対応できます

```python
def handle_human_message(self, human_message: str, thread_id: str) -> None:
    if self.is_next_human_review_node(thread_id):
        last_message = self.get_messages(thread_id)[-1]
        tool_reject_message = ToolMessage(
            content="Tool call rejected",
            status="error",
            name=last_message.tool_calls[0]["name"],
            tool_call_id=last_message.tool_calls[0]["id"],
        )
        self.graph.update_state(
            config=self._config(thread_id),
            values={"messages": [tool_reject_message]},
            as_node="human_review_node",
        )
```

https://github.com/os1ma/langgraph-human-in-the-loop-demo/blob/14e09b6cbf9cde9602d5e3542cd773d24d8faf4d/agent.py#L76

https://langchain-ai.github.io/langgraph/how-tos/human_in_the_loop/review-tool-calls/#give-feedback-to-a-tool-call

まとめ

- LangGraphでのHuman-in-the-Loopの実装を紹介しました
- checkpointerとinterrupt_beforeの設定がポイントでした
- 人間によるツールの利用の承認以外にも、ツールの引数の修正や、人間への質問も実装可能です

https://langchain-ai.github.io/langgraph/how-tos/human_in_the_loop/edit-graph-state/

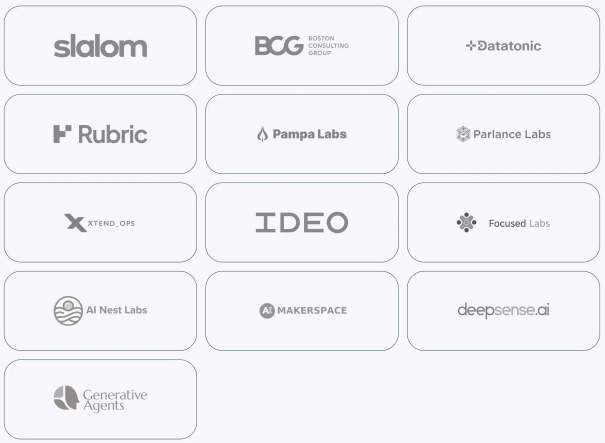
https://langchain-ai.github.io/langgraph/how-tos/human_in_the_loop/wait-user-input/

最近はLangGraphのアップデートがとても活発です

今後のアップデートにも期待しましょう！

（宣伝）LangChain公式エキスパートに認定

株式会社ジェネラティブエージェンツは、日本企業初のLangChain公式エキスパートに認定されました



https://www.langchain.com/experts

法人向けの研修もご提供しているので、SNS（https://x.com/oshima_123）や
会社サイト（https://www.generative-agents.co.jp/）からご相談ください

ご清聴ありがとうございました

Generative Agents