Streaming và Human-in-the-loop trong LangGraph

Streaming = "Truyền liên tục", nghĩa là agent hoặc LLM sẽ trả về **kết quả dần dần**, **theo dòng**, **chứ không phải chờ tính toán xong mới gửi hết**.

- **Streaming Values:** bạn nhận kết quả từng bước nhỏ (thường là dạng text hoặc dữ liệu).
- **Streaming Update:** bạn vừa nhận được giá trị ban đầu, vừa có thể **nhận bản cập nhật** mới hơn về sau (khi tác vụ tiến triển).

Thuật ngữ	Ý nghĩa	Ví dụ thực tế
Streaming Values	Nhận các giá trị từng phần	Từng đoạn text của câu trả lời
Streaming Updates	Các bản cập nhật của 1 giá trị ban đầu	Đang xử lý... Đã hoàn thành

Ví dụ đơn giản:

Giả sử bạn có một AI Agent thực hiện tóm tắt 1 tài liệu lớn.

- Ban đầu, nó đọc tài liệu và trả về **"Đang đọc chương 1..."** (streaming value)
- Khi đọc thêm, nó **cập nhật** thành **"Đã đọc chương 1, đang đọc chương 2..."** (streaming update)

Bạn nhận được **dòng text** đầu tiên nhanh chóng, sau đó **nhiều bản cập nhật liên tiếp**.

Ví dụ cụ thể bằng Python (giả lập):

```
import time

# Streaming giá trị ban đầu
def stream_value():
    print("□□ Loading document...")
    time.sleep(1)
    yield "□  Step 1: Read Chapter 1"

    time.sleep(2)
    yield "□  Step 2: Read Chapter 2"
```

```
    time.sleep(1)
    yield "□  Step 3: Summarizing Chapters"

# Xử lý streaming
for value in stream_value():
    print(f"□□ Received update: {value}")
```

Kết quả

```
Loading document...
Received update: □  Step 1: Read Chapter 1
Received update: □  Step 2: Read Chapter 2
Received update: □  Step 3: Summarizing Chapters
```

Ví dụ đơn giản về cách streaming câu trả lời trong chatbot

Tình huống:

Người dùng hỏi: **"Tóm tắt tác phẩm Dế Mèn Phiêu Lưu Ký"**

Ta sẽ cho chatbot trả lời dần dần (streaming) theo từng câu hoặc từng đoạn văn bản.

Ví dụ Python (giả lập):

```
import time

def chatbot_streaming_response():
    yield "Dế Mèn Phiêu Lưu Ký là tác phẩm nổi tiếng của nhà văn Tô Hoài."
    time.sleep(1)

    yield "Tác phẩm kể về hành trình phiêu lưu của chú dế Mèn thông minh, gan dạ và thích khám phá."
    time.sleep(1)

    yield "Trong hành trình đó, dế Mèn đã học được nhiều bài học quý giá về tình bạn, lòng dũng cảm và sự trưởng thành."
    time.sleep(1)

    yield "Tác phẩm không chỉ hấp dẫn thiếu nhi mà còn chứa đựng nhiều triết lý sống sâu sắc."
    time.sleep(1)
```

```
# Giả lập chatbot gửi từng đoạn
for chunk in chatbot_streaming_response():
    print(f"□□ {chunk}")
```

Kết quả mô phỏng trên terminal:

```
□□ Dế Mèn Phiêu Lưu Ký là tác phẩm nổi tiếng của nhà văn Tô Hoài.
□□ Tác phẩm kể về hành trình phiêu lưu của chú dế Mèn thông minh, gan dạ và thích khám phá.
□□ Trong hành trình đó, dế Mèn đã học được nhiều bài học quý giá về tình bạn, lòng dũng cảm và sự trưởng thành.
□□ Tác phẩm không chỉ hấp dẫn thiếu nhi mà còn chứa đựng nhiều triết lý sống sâu sắc.
```

Sự khác biệt chính giữa `.invoke()` và `.stream()`

Đặc điểm	`.invoke()`	`.stream()`
Cách hoạt động	Gọi toàn bộ chain hoặc agent và chờ kết quả	Trả về kết quả **từng phần một** (token, chunk, etc.)
Tốc độ phản hồi	Trả lời sau khi xử lý xong toàn bộ	Trả lời gần như **ngay lập tức**, theo thời gian thực
Trường hợp sử dụng	Tốt cho các xử lý ngắn hoặc không cần realtime	Lý tưởng cho **chatbot**, **UI realtime**, hoặc truyền dữ liệu
Trả về	Toàn bộ output cùng lúc (ví dụ một string)	Một generator (yield từng phần kết quả)
Tích hợp UI	Không thể hiện đang gõ	Có thể hiện đang gõ như ChatGPT

Lời khuyên thực tế khi triển khai

1. **Ưu tiên dùng stream_mode="values"** khi phát triển để dễ debug
2. Chỉ chuyển sang "updates" khi ứng dụng đã ổn định và cần tối ưu hiệu suất
3. Luôn thiết kế các "điểm kiểm soát" (checkpoints) cho human-in-the-loop tại:
   - Bước ra quyết định quan trọng
   - Khi cần xác nhận tính chính xác
   - Khi agent không chắc chắn
4. Xử lý lỗi thông minh:

```
try:
    for chunk in agent.stream(...):
        process(chunk)
except HumanInterventionRequired as e:
```

```
    handle_exception(e)
    wait_for_human_decision()
```

Bổ sung kiến thức quan trọng

Tại sao .stream() mạnh mẽ hơn?

- Cho phép xây dựng **agent có khả năng tự sửa lỗi**
- Tạo điều kiện cho **học tăng cường từ con người** (RLHF)
- Hỗ trợ **xử lý tác vụ dài** mà không bị timeout
- Cho phép **hiển thị tiến trình thời gian thực** cho người dùng

Pattern hay gặp:

```
# Hybrid approach - Kết hợp cả invoke và stream khi cần
if is_simple_task(task):
    return agent.invoke(task)
else:
    return process_stream(agent.stream(task))
```

Hãy coi `.stream()` như "cửa sổ vào bộ não" của agent - bạn có thể quan sát và can thiệp vào quá trình xử lý theo cách chưa từng có với `.invoke()` thông thường!

Tác giả: Đỗ Ngọc Tú

Công Ty Phần Mềm VHTSoft

Phiên bản #4
Được tạo 29 tháng 4 2025 05:41:24 bởi Đỗ Ngọc Tú
Được cập nhật 2 tháng 5 2025 07:45:21 bởi Đỗ Ngọc Tú