Phân tích Kiến trúc Bộ nhớ cho Hệ thống Multi-Agent

Nam Nguyen

1 Phân tích 4 Kiến trúc Đề xuất

1.1 Zep (Temporal Knowledge Graph)

- **Loại bộ nhớ:** Episodic (Sự kiện) + Semantic (Ngữ nghĩa) + Temporal (Thời gian).
- **Kiến trúc:** Sử dụng **Graphiti Engine**. Dữ liệu được tổ chức thành đồ thị 3 tầng:
  1. *Episodes:* Sự kiện thô.
  2. *Entities/Edges:* Thực thể và mối quan hệ được trích xuất.
  3. *Communities:* Các cụm thông tin liên quan.
- **Điểm đột phá: Bi-temporal Tracking**. Zep lưu trữ 2 dòng thời gian: thời điểm sự kiện xảy ra (`valid_at`) và thời điểm hệ thống ghi nhận (`transaction_time`).
- **Cơ chế: Edge Invalidation**. Khi thông tin thay đổi (VD: User đổi địa chỉ), Zep không xóa địa chỉ cũ mà đánh dấu nó là "hết hạn". Điều này cực kỳ quan trọng cho các Agent cần audit trail (truy vết) hoặc hiểu sự thay đổi trạng thái.

1.2 Mem0 (Production-Ready Memory)

- **Loại bộ nhớ:** Hybrid (Vector + Graph) tập trung vào User Profile.
- **Kiến trúc:** Thiết kế như một layer trung gian tối ưu hóa.
  - *Mem0 Base:* Vector DB lưu trữ các "fact".
  - *Mem0 Graph:* Đồ thị thực thể để tăng cường ngữ cảnh.
- **Điểm đột phá: Memory Management Operations**. Thay vì chỉ lưu đè, Mem0 dùng LLM để quyết định hành động: `ADD` (thêm), `UPDATE` (sửa), `DELETE` (xóa mâu thuẫn), `NOOP` (bỏ qua).
- **Cơ chế:** "Tool Call Approach". Nó biến việc quản lý bộ nhớ thành một công cụ (tool) mà Agent có thể gọi, giúp giảm thiểu rác dữ liệu ngay từ đầu vào.

1.3 A-Mem (Agentic Memory)

- **Loại bộ nhớ:** Self-Evolving Knowledge Base (Cơ sở tri thức tự tiến hóa).
- **Kiến trúc:** Lấy cảm hứng từ phương pháp ghi chú **Zettelkasten**.
  - Lưu trữ dưới dạng các "Atomic Notes"(Ghi chú nguyên tử).
  - Mỗi note có metadata phong phú (Tags, Keywords, Context).
- **Điểm đột phá: Memory Evolution**. Khi nạp thông tin mới, hệ thống kích hoạt một "Agent nội bộ"để xem xét các ghi chú cũ. Nếu thấy liên quan, nó sẽ *viết lại* hoặc *cập nhật tags* của ghi chú cũ để phản ánh sự hiểu biết mới.
- **Cơ chế:** Tạo liên kết động (Dynamic Linking) giữa các ghi chú dựa trên sự tương đồng ngữ nghĩa, giúp Agent có khả năng suy luận đa bước (multi-hop reasoning) rất tốt.

1.4 AriGraph (World Model)

- **Loại bộ nhớ:** State-based World Model (Mô hình thế giới dựa trên trạng thái).
- **Kiến trúc:** Tách biệt rõ ràng nhưng liên kết chặt chẽ giữa:
  - *Semantic Graph:* Kiến thức tĩnh (A là chìa khóa của B).
  - *Episodic Graph:* Chuỗi sự kiện theo thời gian (Tại t1, tôi thấy A ở phòng khách).
- **Điểm đột phá: State Tracking**. Nó giải quyết vấn đề "trạng thái ẩn"trong môi trường. Ví dụ: Agent đi ra khỏi phòng, quay lại thì phải nhớ cửa đang mở hay đóng.
- **Cơ chế:** Kết hợp tìm kiếm ngữ nghĩa để hiểu vật thể là gì, và tìm kiếm sự kiện để biết vật thể đó đang ở đâu/trạng thái thế nào.

2 Bảng So sánh Hiệu năng & Kỹ thuật

Dưới đây là so sánh giữa phương pháp **Hiện tại (Full Context)** và 4 giải pháp đề xuất.

Tiêu chí	Full Context (Hiện tại)	Mem0	Zep	A-Mem	AriGraph
Độ Trễ (Latency)	**Rất cao** (khi context dài)	**Thấp nhất** (Search ∼0.15s)	**Trung bình** (Search ∼3.2s)	**Cao** (Do xử lý tiến hóa note)	**Cao** (Do duyệt 2 tầng graph)
Độ Chính Xác	Cao (trong cửa sổ context)	**Rất cao** (SOTA trên LOCOMO)	Cao (với dữ liệu thời gian)	Cao (với suy luận phức tạp)	Cao (với trạng thái vật lý)
Chi Phí Token	**Rất cao** (Lặp lại toàn bộ lịch sử)	**Thấp nhất** (∼7k tokens lưu trữ)	Cao khi Ingestion (∼600k tokens xây graph)	Trung bình (∼1.2k tokens/op)	Trung bình
Khả năng Suy luận	Tốt (ngắn hạn), Kém (dài hạn)	Tốt (Fact Retrieval)	Tốt (Temporal Reasoning)	**Xuất sắc** (Multi-hop reasoning)	Xuất sắc (Spatial/State planning)
Triển khai	Dễ	Dễ (Có SDK/Cloud)	Trung bình (Cần Neo4j/Postgres)	Khó (Cấu trúc phức tạp)	Khó (Cần tùy biến nhiều)
Loại Agent phù hợp	Chatbot đơn giản	User Assistant, Customer Support	Legal Bot, HR Bot, Finance Bot	Researcher, Analyst Agent	Game NPC, Robot, Simulation

Bảng 1: So sánh hiệu năng các kiến trúc bộ nhớ

3 Insights & Khuyến nghị cho Hệ thống Multi-Agent

Vì công ty đang xây dựng **Multi-Agent System**, việc chọn một kiến trúc duy nhất cho tất cả các Agent có thể không tối ưu. Dưới đây là chiến lược phối hợp và phương án triển khai cụ thể:

3.1 Insight 1: "One size does not fit all"(Không có giải pháp vạn năng)

- **User-facing Agent (Giao tiếp người dùng):** Cần phản hồi nhanh và nhớ sở thích người dùng.
  - *Khuyến nghị:* **Dùng Mem0**. Với độ trễ thấp và khả năng quản lý User Profile tốt, Mem0 giúp trải nghiệm chat mượt mà và cá nhân hóa cao.
- **Knowledge Worker Agent (Nghiên cứu/Phân tích):** Cần tổng hợp thông tin từ nhiều tài liệu, báo cáo.
  - *Khuyến nghị:* **Dùng A-Mem**. Khả năng "tiến hóa bộ nhớ"và liên kết các ghi chú rời rạc giúp Agent này tìm ra các insight sâu mà các phương pháp RAG thường bỏ sót.
- **Audit/Monitor Agent (Giám sát/Lưu trữ):** Cần theo dõi lịch sử thay đổi của hệ thống hoặc dữ liệu doanh nghiệp.
  - *Khuyến nghị:* **Dùng Zep**. Khả năng Bi-temporal (dòng thời gian kép) giúp trả lời chính xác câu hỏi: "Dữ liệu này đúng vào thời điểm nào?".

3.2 Insight 2: Chiến lược thay thế Full Context (Hybrid Context)

Không cắt bỏ hoàn toàn Full Context, áp dụng mô hình lai:

1. **Short-term:** Giữ Full Context (Sliding Window) cho 5-10 turn hội thoại gần nhất để đảm bảo độ mượt.
2. **Long-term:** Đẩy các hội thoại cũ vào Memory Layer (Mem0/Zep).
3. **Retrieval:** Query Memory Layer trước khi gọi LLM để inject "facts"vào System Prompt.

3.3 Kết luận Phương án Triển khai (Implementation Plan)

Dựa trên phân tích hiệu năng và chi phí (Effort vs. Reward), hệ thống sẽ tiến hành triển khai theo định hướng sau:

Quyết định: Triển khai tích hợp **Mem0** vào pipeline của **LangGraph**.

Lý do lựa chọn:

- Đây là phương án tối ưu nhất để giải quyết ngay lập tức vấn đề chi phí token và độ trễ của Full Context hiện tại.
- Mem0 cung cấp khả năng quản lý bộ nhớ (CRUD) thông minh thông qua Tool Call, phù hợp với kiến trúc Graph của LangGraph.
- Các kiến trúc phức tạp hơn (A-Mem, Zep) sẽ được xem xét bổ sung sau nếu phát sinh nhu cầu chuyên biệt (như Time Travel hoặc Deep Research).

4 Tài liệu tham khảo

1. **A-Mem**: *Agentic Memory for LLM Agents* (Published 8 Oct 2025).
2. **AriGraph**: *Learning Knowledge Graph World Models with Episodic Memory for LLM Agents* (Published 5 Jul 2024).
3. **Mem0**: *Building Production-Ready AI Agents with Scalable Long-Term Memory* (Published 28 April 2025).
4. **ZEP**: *A Temporal Knowledge Graph Architecture for Agent Memory* (20 Jan 2025).