Agent Memory 设计讨论

2026-02-21

1 背景

FlowCabal 的 `.flowcabal/` 目录是程序唯一的状态根目录，位于仓库根目录（免安装分发，不污染用户 home 空间）。其下三个子目录各司其职：

- `data/`：跨项目共享的持久化配置（Workflow 模板、LLM 配置等，对应 `types.ts` 中定义的类型）
- `runner-cache/<project>/`：运行时缓存（节点输出、执行状态），按小说项目隔离
- `memory/<project>/`：Agent 记忆，按小说项目隔离

本文档记录关于 `memory/` 设计的讨论过程与结论。讨论是渐进式的——第 3–6 节各自提出阶段性结论，后续章节会对前面的结论做修正或深化，最终综合为第 7 节的设计方案。

2 核心命题：约束查询

一切高级需求的底层能力是**约束查询**：不论是人还是 AI 给出新内容，能判断与现有内容有没有冲突，冲突在哪里。

这一命题来自与作家朋友的讨论。作家视角的记忆需求可归纳为八条：

1. 故事梗概记忆（各弧发生了什么）
2. 角色关系记忆（随时间变化）
3. 角色背景记忆（经历如何塑造性格）
4. 角色性格特征和情感记忆（关键事件的影响）
5. 角色动机和利益记忆
6. 作者对各段剧情的设想（防止前后矛盾）
7. 设定记忆（世界观、体系原理、防止违背自设）
8. 伏笔记忆（埋了什么坑没填）

八条中五条是角色相关的——角色是长篇小说的原子单位，也是 AI 最容易崩坏的地方。

3 Memory 空间与作品内容空间的关系（初步建模）

3.1 两个空间不等价

设 C 为从 manuscripts 全文可推导出的所有断言集合，M 为 memory/ 文件捕获的断言集合。

$M \subset C$，永远是有损的。全文包含的信息维度是无限的——对话暗示的情绪变化、场景的感官细节、句子节奏暗示的角色状态——无法被有限的结构化文件完整编码。

3.2 初步目标：区分结构性事实与质感信息

C 可分为两类：

- **结构性事实** C_s：可被显式矛盾的——角色是左撇子、魔法需要咏唱、A 和 B 在第五章决裂
- **质感信息** $C \setminus C_s$：不产生逻辑矛盾的——某段对话的措辞感觉、一场雨戏的氛围、叙述节奏

约束查询的目标是 $M \supseteq C_s$（memory 覆盖所有结构性事实）。如果做到，在 memory/ 中做约束查询和在 manuscripts 中做约束查询是功能等价的。损耗全部落在质感信息上。

但这一目标在下一小节即被证明不可实现，第 4 节将给出更实际的替代方案。

3.3 修正：不可能完美覆盖

C_s 本身不可穷举。“第三章提到阳台上有盆仙人掌”——它是否结构性事实，取决于未来是否有场景再次涉及这个阳台。一个事实是否“结构性的”取决于未来的写作方向，而未来不可知。

因此 $M \supseteq C_s$ 不可能精确实现。

3.4 阶段性结论：缓存模型

Memory 本质上是一个缓存系统：

- **L0**：`index.md` — 目录级，极度压缩，始终在 Agent 上下文中
- **L1**：memory/ 各文件 — 结构化事实，按需加载
- **L2**：manuscripts/ 全文 — 完整信息，验证用

Cache miss 的处理路径：Agent 在 memory/ 中未找到相关约束 → 回退到 manuscripts/ 搜索相关章节 → 提取新发现的结构性事实补入 memory/。

这意味着 **memory/ 是自增长的**。每次 cache miss 都让 memory/ 更完整。随着写作推进和约束查询次数增加，M 逐渐逼近 C_s。

缓存模型回答了“memory/ 和 manuscripts/ 的关系是什么”，但尚未回答“memory/ 内部应该优先存什么”。第 4 节引入结构性概率来解决优先级问题，第 5 节进一步修正 memory 的内容形式。

4 深化一：结构性概率与剧情张力

4.1 大道理管着小道理

虽然一个事实是不是结构性事实不可精确判定，但其结构性概率是可以估计的。这来源于战略学的基本思想：**大道理管着小道理**。

事实天然有层级，层级越高，结构性概率越高，需要越大的累积力才能推翻：

层级	例子	推翻所需的力
前提级	他是一个父亲	几乎不可能
弧级	他深爱他的女儿	重大事件
章级	他答应了女儿周末去游乐园	一个电话
场景级	他把车停在了路边	下一句话

4.2 临界点即张力

小说中几乎所有强叙事张力都可以建模为：**一组持续累积的低层事实，逐渐逼近对某个高层事实的颠覆阈值**。

以奥托为例：

- 高层事实：奥托是卡莲的守护者
- 累积的低层事实：

- 他开始研究禁忌技术
- 他对李素裳隐瞒了实验进展
- 他默许了一次无辜者的牺牲
- ……

· 临界点 → 高层事实翻转：奥托从“守护者”变成了“为目的不择手段的人”

这意味着 Agent 不仅要存储事实，还应该能感知哪些低层事实正在对哪些高层事实施压——这不再是约束检查，而是**叙事智能**。

但这属于更高层的 Agent 能力问题，不是 memory/ 文件结构本身要解决的。Memory/ 要做的是：**按结构性概率的层级来组织信息，让高层事实容易被找到，低层事实通过链接可达**。

5 深化二：生成性事实，而非派生断言

5.1 Intensional vs Extensional

第 3 节将 memory 建模为“断言集合 M”，隐含地假设 memory 的内容是断言。这一假设需要修正。

对于一个事实，为真的断言和为假的断言都是无限多的。尝试用断言式约束（“奥托绝不会放弃复活卡莲”）来描述角色，等于试图穷举一个无限集合。

正确的做法是存储**生成性事实**（intensional definition），而非**派生断言**（extensional definition）：

Extensional（断言式）	**Intensional（生成式）**
奥托绝不会放弃复活卡莲	核心驱动力：对卡莲之死的愧疚
奥托不会信任陌生人	背景：[→ manuscripts/ch03]
奥托说话总是彬彬有礼	

左列试图列举无限集合的成员；右列给出集合的生成规则。Agent 从核心驱动力实时推导“他在这个情境下会不会这样做”——推导能力正是 LLM 的强项。

5.2 文件内容的写法

Memory 文件应当**紧凑而有因果深度**，而非冗长而全面。每个事实应包含“为什么”而不只是“是什么”，因为因果关系才是约束的生成规则。

6 深化三：稀疏图与跳转链接

6.1 Memory 的真正结构是有向图

每个 memory 文件内部包含跳转链接，指向其他 memory 文件或 manuscripts/ 中的具体位置。Manuscripts/ 自然地成为图中的叶子节点——不是“另一种记忆”，而是最高精度的信息源，通过链接按需可达。

```
index.md
  ├→ premise.md
  ├→ characters.md #奥托
  │     ├→ characters.md #卡莲        （关系链接）
  │     ├→ chronicle.md #神州折剑录    （事件链接）
  │     └→ manuscripts/ch15.md#决裂    （原文链接）
```

```
├→ world.md #崩坏能
│    └→ manuscripts/ch02.md#设定    （原文链接）
└→ ...
```

6.2 稀疏性约束

每个节点的出链应当少（约 3–5 个），指向“如果需要深入了解这个事实，去这里”。如果一个节点需要链接到过多位置，说明它本身应该被拆分。

> 过于稠密的链接图等价于“读所有东西”——失去了缓存的意义。

6.3 SubAgent 上下文管理

Agent 读到跳转链接时，面临“跟进还是不跟进”的选择。这天然适合递归式 SubAgent 调用：

1. 主 Agent 读 `characters.md`，发现需要核实奥托与卡莲的决裂细节
2. Spawn SubAgent：读 `manuscripts/ch15.md`，回答“决裂的具体触发条件是什么”
3. SubAgent 返回一句话结论
4. 主 Agent 继续工作，上下文没有被 ch15 全文污染

链接的存在让信息**可达**，SubAgent 的边界让上下文**不爆炸**。

7 综合结论：预设文件结构

综合第 3–6 节的渐进讨论，最终设计方案如下。

7.1 目录结构

```
memory/<project>/
├── index.md          L0 索引（Agent 导航入口）
├── premise.md        前提：梗概、类型、主题内核、读者预期
├── characters.md     角色：背景因果 → 性格 → 动机 → 关系（含时间变化）
├── world.md          世界：硬规则、体系原理、边界
├── voice.md          叙事：POV 规则、时态、文体锚点（正例 + 反例）
├── outline.md        大纲 + 作者对各段剧情的设想与创意笔记
├── chronicle.md      已发生事件梗概（按弧/章节，中观粒度）
├── threads.md        伏笔/悬念追踪（埋在哪 → 计划何时收）
└── manuscripts/      定稿章节
```

这些是 init 时创建的种子文件。随创作推进，用户和 Agent 可自由添加新文件（如 `magic-system.md`、`ch05-notes.md`），只需更新 `index.md` 使其可被发现。

7.2 文件内部格式

每个文件的内容遵循三个原则：

1. **生成性事实**：描述因果和驱动力，不列举断言
2. **稀疏跳转链接**：指向相关的其他记忆文件或 manuscripts/ 原文
3. **层级组织**：高结构性概率的事实在前，低层细节通过链接可达

示例（`characters.md` 中的一个角色条目）：

```
## 奥托

核心驱动力：对卡莲之死的愧疚 → 不惜一切代价复活她
背景因果：[→ manuscripts/ch03.md#奥托回忆]
```

关系：
- → 卡莲：愧疚与执念的对象 [→ characters.md#卡莲]
- → 李素裳：曾经的战友，后被工具化 [→ chronicle.md#神州折剑录]

关键转折：
- 失手害死卡莲 [→ manuscripts/ch08.md#卡莲之死]
- 开始研究禁忌技术 [→ manuscripts/ch12.md#逆转实验]

8 待决问题

- Memory/ 内部的具体子目录是否需要硬编码，还是完全自由组织（仅靠 `index.md` 发现）
- `chronicle.md` 随章节增多会膨胀，何时以及如何拆分
- Agent 自动从 manuscripts 提取事实补入 memory 的触发时机和粒度
- 叙事智能（感知张力累积）如何从存储层升级为 Agent 能力层
- `runner-cache/` 的详细设计（与 core-runner 实现直接相关，另文讨论）