MAS-字节论文解读

论文地址：https://arxiv.org/pdf/2507.11988

论文目的

阐述字节内部觉得更有效的多智能体的构建方法。

为什么会有这个需求

因为在以往的多智能体的构建多，多数是按照了“规划者-执行者”这样的要求来构建的，但是这样的构建方法会带来很多的问题：

- 僵化的计划执行。传统计划通常是静态的。一旦制定完成，规划器通常必须等待所有执行者报告完成才能汇总结果。然而个体执行者的自主性意味着其行动可能偏离既定计划，导致工作不完整或冗余。这种偏差为规划器提供了不可靠的反馈，最终会降低整体系统性能（Cemri 等人，2025）。
  【人话：有人拖后腿又得等】
- 静态代理能力局限。计划-执行模型预设代理的预定义能力足以应对所有预期任务，但实践中这一假设常被打破。例如，对代理技能描述的不准确或不完整会导致规划器做出次优任务分配（Shen 等，2023）。此外，系统无法适应需要新工具或能力的突发任务，从而限制了其在新型问题上的扩展性和表现。
  【人话：去不了想要的智能体】
- 低效通信瓶颈。代理间的信息交接常成为性能瓶颈。任务委派过程中关键上下文可能丢失或失真，阻碍了无缝协作。
  执行问题（Yan，2025 年）因缺乏共享状态管理系统而加剧。由于状态更新通常仅在任务完成时汇总，智能体往往只能掌握整体进度的不完整视图。这种实时共享意识的缺失会导致重复劳动和关键延误。
  【人话：丢失信息，执行者信息无法共享，全靠规划者】

改进思路

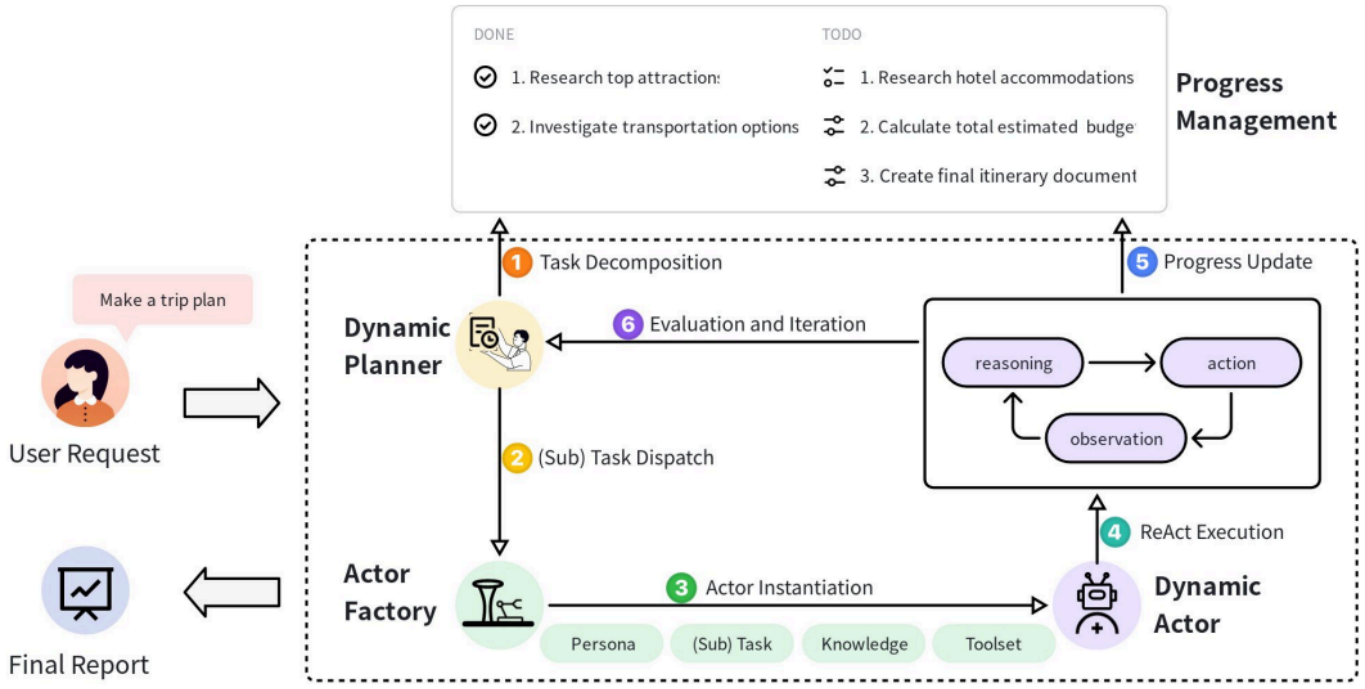

Figure 1: The workflow of Aime framework.

- 动态规划器作为任务管理的核心协调者，将高层目标分解为可执行子任务的层次结构，并维护一个全局任务列表以跟踪每个子任务的状态。它持续监控执行进度，并根据动态执行器的反馈和进度管理模块的状态更新，动态调整计划。
- 角色工厂负责实例化专为特定子任务需求定制的专业角色。当接收到子任务时，工厂会分析其规格要求，以确定角色的最佳配置方案。该过程包括选择合适的人物设定、相关知识库以及必要的工具集。随后工厂会通过定制化提示词和系统配置来组装角色，确保每个角色都为其分配的任务量身打造。
- 动态角色是由动态规划器分配执行特定子任务的自主代理。每个角色都采用 ReAct 框架（Yao 等人，2023 年），通过"推理"与"行动"的迭代循环运作。在每个循环中，执行者从其预设工具包中选择最合适的工具，执行操作，然后根据结果观察评估成效。该循环将持续运行，直至满足子任务的完成标准。
- 进度管理模块作为共享内存和中央状态协调系统全局运作。它维护着整个任务层次结构的结构化表示，以及所有子任务的实时状态。这一集中化记录确保动态规划器与所有执行者对进度保持统一认知。关键之处在于，该模块通过在每项任务条目中嵌入明确的完成标准，为验证任务完成提供了清晰客观的衡量依据。

具体步骤

【请注意这里加粗的部分，它们对应了下一章的内容】
步骤 1：任务分解。当**动态规划器**接收到任务时，工作流程随即启动。用户下达任务后，规划器将该请求分解为结构化的子任务计划，并在进度管理模块中初始化相应的任务列表。
步骤2：（子）任务派发。动态规划器随后从计划中识别出下一个可执行的子任务，并将其规范派发给**执行者工厂**。
步骤3：执行者实例化。当接收到子任务规范时，执行者工厂将初始化一个专门的**动态执行者**。该执行者专为子任务打造，配备了精确的角色设定、知识储备及执行所需的工具集，以确保高效完成任务。
步骤4：反应式执行。新实例化的执行者通过遵循反应范式来执行其分配的子任务。

它在一个推理与行动的循环中运作，调用工具包中的工具，逐步推进子任务目标的实现。
步骤 5：进度更新。在执行过程中，执行者持续向**进度管理模块**报告状态更新。该模块确保全局任务状态保持同步，并对所有组件（特别是动态规划器）可访问。
步骤 6：评估与迭代。当子任务完成后，执行者将最终结果上报至动态规划器。规划器评估该结果，更新全局计划，并返回步骤 2 以分派下一个可用子任务。此循环将持续进行，直至成功完成顶层用户请求。

每个模块的细节是什么

动态规划器

动态规划器旨在解决传统"规划-执行"框架中固有的执行僵化问题。在此类框架中，规划器通常保持闲置状态，直至所有子任务完成。
该规划器的核心在于管理两个操作层级：维护全局任务结构并确定即时下一步行动。他们将这种双重职责形式化为单一的迭代推理步骤。给定总体目G，在每一步t，规划器（记作代理$A_{planner}$）会评估当前任务列表$\mathcal{L}_t$和过往结果历史$\mathcal{H}_t = \{o_1, ..., o_t\}$。其操作由以下函数定义：

$$(\mathcal{L}_{t+1}, g_{t+1}) = LLM_{planner,(G,\mathcal{L}_t,\mathcal{H}_t)}$$

总结一下就是通过之前的总目标，之前的任务列表还有之前的历史操作来指定下一步的目标和行动。规划器在每次迭代中产生两个输出：

1. $\mathcal{L}_{t+1}$是更新后的全局任务列表，代表规划器的战略或"大步骤"输出，反映了基于新信息对任务层级的重新理解。若无需重大战略调整，$\mathcal{L}_{t+1}$可能仅是带有更新任务状态的$\mathcal{L}_t$副本。
2. g_{t+1}是系统接下来要执行的具体可操作动作，即战术或"小步骤"输出，例如向执行器工厂派发子任务。

执行者工厂

根据子任务g_{t+1}的具体需求，按需组装专用执行体。

$$A_t = \mathcal{F}_{factory}(g_t) \quad where \quad A_t = LLM\{LLM_t, \mathcal{T}_t, P_t, M_t\}$$

工厂的核心功能是选择专用工具包$\mathcal{T}_t$并构建定制化的提示词P_t。

工具包选择

复杂多智能体系统面临的主要挑战在于管理庞大而多样的工具集。将所有可用工具直接呈现给智能体的 LLM 可能导致低效选择或错误。为解决这个问题，Aime 将工具预打包成功能分类的模块化组合（例如"网络搜索"工具包、"文件系统"工具包）。工厂通过选择适当的工具包组合来形成最终工具集 T，而非从扁平化的单个工具列表中挑选。这种基于工具包的方案确保了功能完整性，并降低了关键工具遗漏的风险。

提示词构成

系统提示P_t由多个模块化组件动态组装而成，旨在为执行者创建精确的操作上下文：

$$P_t = Compose(p_t, desc(\mathcal{T}_t), \kappa_t, \epsilon, \Gamma)$$

其中组件包括：

- 角色设定（p_t）。定义执行者的专业角色与专长领域（例如"一位擅长策划独特难忘旅程的资深旅行规划师"）。该角色设定会根据子任务 g 动态生成，从而形成专属的领域专家。
- 工具描述（desc($\mathcal{T}_t$)）。对选定工具集$\mathcal{T}$提供简洁的文本说明。通过提供最小必要工具集来缩小 LLM 的决策空间，从而提升专注度与执行效率。
- 知识库（κ）。包含从知识库动态检索的、与子任务高度相关的支持信息。以旅行规划任务为例，可能包括识别当地景点的实用技巧。
- 环境参数（ϵ）。提供全局上下文信息，如操作系统详情或系统级约束条件（例如当前时间、访问权限），确保执行者的行为具有环境感知能力。
- 格式（Γ）。指定所需的输出结构（例如 JSON 模式），确保执行体的响应能够被可靠解析以实现自动化处理和状态更新。

动态执行者

当被执行体工厂实例化后，动态执行体将作为自主智能体专注于执行其分配的子任务，其行为遵循了ReAct范式。执行者A_t通过反复调用其核心 LLM 来完成子任务。在其内部循环的每一步k中，执行者会分析当前目标与局部历史记录，生成新思考（thought）及后续行动（action）。该过程形式化表示为:

$$(thougt_{k+1}, action_{k+1}) = LLM_t(P_t, (g_t, H_k))$$

其中它这个H_k是存储在本地记忆M_t重的历史(action, observation)对序列。该循环通过三个不同阶段展开：
1.推理阶段。执行者基于子任务目标、过往行动及相应观察结果进行推演，制定下一步即时行动计划。
2.行动。根据推理结果，执行者从其专用工具集T_t中选择并执行一个操作，通常是对某个工具的调用。
3.观察。执行者接收所调用工具的输出结果。这一新观察会被追加至其历史记录H_k中，并作为下一轮推理阶段的关键上下文。
它很重要的核心能力在于'update'的功能，也就是为整个框架提供新鲜的任务目标和状态的感觉。

进度管理模块

多智能体系统的核心挑战在于保持对任务进展的连贯且全局一致的认知。进度管理模块通过作为框架的集中式状态管理器来解决这一问题，为整个任务层级建立单一事实来源。这确保了动态规划器和所有动态执行者都能基于系统状态的共享统一视图进行操作。

进度列表

该模块的核心是一个全局可访问的层级数据结构，我们称之为进度列表（记作$\mathcal{L}$）。它完整呈现了从高层目标到细粒度子任务的分解过程。典型实现采用人类可读且机器可解析的格式，如 Markdown 任务列

表：

```
- Objective 1: Perform Initial Research
    - [x] Sub-objective 1.1: Research top attractions
    - [x] Sub-objective 1.2: Investigate transportation options
- Objective 2: Finalize Itinerary and Budget
    - [ ] Sub-objective 2.1: Research hotel accommodations
    - [ ] Sub-objective 2.2: Calculate total estimated budget
    - [ ] Sub-objective 2.3: Create final itinerary document
```

进度清单的核心特性包括：

1. 实时状态追踪。每个项目都标注当前状态（如已完成"[x]"，待处理"[ ]"），提供系统全局进展的一目了然视图。
2. 嵌入式上下文与依赖关系。层级结构隐式编码了任务间的依赖关系。此外，每个项目可嵌入或链接至明确的完成标准，为验证提供客观依据。

！！！
整一个逻辑都要很注意留下指针内容，方便针对问题进行回顾
！！！