

MRAC 的理论起点与递进研究纲领：从不收敛诊断到工程 AGI

——多轮审计收敛 L1-L5 的命题分层与研究议程

刘明·ORCID: 0009-0001-2816-538X

版本 v1.0 · 2026 年 8 月

摘要

既有研究提出多轮审计收敛 (Multi-Round Audit Convergence, MRAC)，以同一工程对象在“审计—修复—再审计”过程中的问题变化判断系统能否稳定处理当前任务；另一项研究则以搜索空间 (S)、覆盖能力 (C) 和交付可靠性 (R) 的关系解释 AI 交付的不确定性。本文以这两项研究为前提，论证 MRAC 不收敛与收敛分别能够支持何种覆盖判断，并将其组织为 L1-L5 递进研究纲领。

本文在明确操作性定义与适用条件的基础上论证：在任务与审计条件稳定、问题真实有效、修复持续接受再审计且资源边界预先确定时，MRAC 持续不收敛可作为当前完整 AI 开发系统尚不能可靠覆盖任务搜索空间的条件性诊断。对于 L2，在修复结果能够保持且漂移受控时，收敛表明已暴露失败路径得到了更充分的处理与验证，并为当前审计配置下的局部覆盖趋于饱和并提供正向证据；这一证据对完整交付相关搜索空间的代表性及其外推强度仍需经验验证。在此基础上，L3 研究递归拆分能否使不收敛任务恢复收敛，L4 研究完整工程流程能否长期自主推进，L5 则检验同类机制能否扩展到工程以外的智能任务。

关键词：多轮审计收敛；MRAC；SCR 元关系；搜索空间；覆盖能力；交付可靠性；递归拆分与验证；工程 AGI；AI 工程

1. 引言

当前 AI 已经能够生成大量完成度较高的工程结果，但“能够生成”与“能够确认交付”之间仍存在根本差异。一次执行成功、测试通过或单轮审计未发现问题，只能说明系统在当前观测路径上没有继续暴露失败，不能说明与交付相关的关键条件和失败路径已经获得充分覆盖。对于长期复杂项目，由此产生的风险会不断累积：局部遗漏不仅影响当前交付，也会进入代码、依赖、接口和项目状态，扩大未来任务需要处理的搜索空间。

迭代反馈与修正本身并非 MRAC 所独有。Self-Refine 通过模型自反馈反复改写输出，Reflexion 将语言反馈保存为情景记忆，以改善后续决策 [3-4]。但缺少外部反馈时，模型的自我修正可能无效，甚至降低推理表现 [5]；软件工程 Agent 的表现也会受到工具接口设计的显著影响 [6]。这些研究主要考察任务表现，而 MRAC 关注的是：多轮问题轨迹能否反映系统与任务之间的覆盖关系。因此，本文不把“多轮调用”本身视为可靠证据，而把模型、Agent、工具、环境、审计和资源预算共同纳入分析。

审计器也不能被预设为无偏验证器。LLM-as-a-judge 研究已经观察到位置偏差、冗长偏好和自我增强等问题 [7]。Calboreanu 对提示规范开展的案例研究则记录了九轮审计中的非单调缺

陷收敛，并进行了有限的跨模型复现 [8]，为迭代审计轨迹具有经验信息提供了相邻证据。但该研究聚焦提示工程质量保证，没有提出不收敛与可靠覆盖不足之间的条件命题，也未进一步讨论任务拆分和工程自主推进。

本文直接继承 MRAC 与 SCR 的既有工作。MRAC 通过持续审计、修复和再审计，观察新增有效问题是否总体减少，并提出“收敛则推进，不收敛则返回任务拆分”的控制逻辑 [2]；SCR 元关系以搜索空间、覆盖能力和交付可靠性的匹配关系解释 AI 交付结果 [1]。在此基础上，本文集中回答三个问题：

1. MRAC 不收敛在何种条件下能够支持当前系统尚不能可靠覆盖任务搜索空间的诊断？
2. MRAC 收敛能够在什么范围内支持覆盖改善与局部覆盖趋于饱和的判断，哪些外推仍需经验验证？
3. 从这两个理论起点出发，MRAC 通向可靠交付、工程 AGI 与更广义 AGI 的研究任务应如何逐级展开？

本文将上述问题组织为 L1-L5。这里的“层级”不是系统成熟度评分，也不是已经同时成立的五项结论，而是从条件性理论命题逐步扩展到经验验证、工程机制、完整工程自主推进假说和跨领域研究纲领。

2. 理论前提与分析单位

2.1 两项前提

前提一：MRAC 是对覆盖过程的二阶观测。

单轮审计只产生“本轮发现了什么”的一阶结果；多轮审计还会显示新增问题如何变化、修复能否保持、严重问题是否继续出现以及稳定状态能否复现。本文将这种基于多轮轨迹的观察称为二阶观测。MRAC 的观测对象不是某一轮输出，而是整个审计—修复序列。

前提二：可靠交付取决于搜索空间与覆盖能力的匹配。

任务搜索空间与系统覆盖能力都不是脱离任务存在的固定标量。任务边界、工程状态、验证条件、模型、Agent、工具、上下文组织、审计协议和资源预算共同决定二者是否匹配。本文不要求 S、C、R 可以标量化，只使用一个方向性前提：在其他条件相同时，系统对交付相关搜索空间的有效覆盖不足，会限制可获得的交付可靠性。搜索空间治理通过改变任务边界和工程环境，使任务重新进入当前覆盖能力可稳定处理的范围。

以搜索描述问题求解具有经典人工智能理论基础 [9]，但 SCR 中的任务搜索空间并不等同于可枚举的经典状态空间，而是面向具体交付、由工程状态与治理方式共同塑造的分析对象。同样，本文所称交付可靠性是系统—任务层面的交付判断，不替代软件可信性研究对故障、错误、失效及可信属性的既有分类 [11]。

2.2 分析单位：完整系统与任务会话

本文所称“系统”不是单次模型调用，而是由以下部分组成的完整 AI 开发系统：

- 基础模型及其推理设置；
- Agent、检索、编辑与执行工具；
- 上下文加载、代码仓库与运行环境；
- 审计协议、问题分级与修复流程；
- 测试、运行反馈与其他外部验证；

- 状态保存、任务拆分、依赖、合并与回退机制；
- 允许使用的时间、Token、计算与调用预算。

分析单位是一个任务会话：从系统接受相对稳定的任务意图开始，到完成交付、终止任务或将其重新拆分为止。L1 判断的是“当前系统—当前任务—当前资源边界”的匹配状态，而不是基础模型脱离环境后的绝对能力。

3. MRAC L1：不收敛与可靠覆盖不足

3.1 L1 所需定义

为避免用审计结果循环定义覆盖状态，L1 需要区分以下四项概念。

定义 1：交付相关的实质性问题。

如果一个问题会影响关键要求、任务边界、实现成立性、结构完整性、相关状态、回归风险或验证充分性，并可能使结果不能被作为完成交付加以信赖，则它是交付相关的实质性问题。措辞偏好、无交付影响的格式差异、重复报告和已经被证伪的问题不计入。

定义 2：可靠覆盖。

在给定任务、系统条件、风险阈值与资源预算下，如果完整 AI 开发系统能够稳定识别交付所依赖的关键条件，区分重要候选路径，排除决定性错误，解决已发现问题，并形成与风险相称的验证证据，则称当前任务搜索空间处于该系统可可靠覆盖的范围内。可靠覆盖不要求枚举所有可能性，但要求系统能够通过抽象、排除、反馈和验证稳定处理决定交付的部分，而不是依赖一次偶然成功。

定义 3：有效 MRAC 序列。

一个 MRAC 序列只有在以下条件成立时才可用于 L1 判断：

1. 任务目标、非目标、核心约束、验收边界与风险阈值在观察期间保持稳定；
2. 审计范围和核心协议保持可比，协议扩展被单独记录；
3. 每轮审计实际完成，工具失败、上下文缺失和异常中断轮次被排除；
4. 新增问题经过有效性与去重判断，确属当前任务的实质性问题；
5. 已确认问题完成修复、通过重新验证并重新纳入全范围审计；仅修改表面症状、删除审计证据或缩窄范围不计为有效修复；
6. 模型、工具、上下文和资源条件的变化被记录，足以确定当前判断对应的系统边界。

定义 4：持续不收敛。

在预先确定的有效轮次、资源预算或观察窗口内，MRAC 序列持续出现新的中高严重度问题、结构性问题或新问题类别，问题轨迹没有进入与风险阈值相符的稳定区间；或者已修复问题持续复发、修复扰动持续产生新的实质问题，系统无法维持稳定结果。满足上述任一情形，称该任务在当前条件下持续不收敛。

“预先确定”十分重要。不能在观察结果出现后任意改变轮数，选择性地把偶然干净轮次解释为收敛，或把尚未完成的长序列解释为不收敛。

3.2 L1 命题

L1：不收敛诊断命题。

对一个满足有效 MRAC 序列条件的任务，如果在当前系统与资源边界内持续不收敛，则当

前系统的覆盖能力不足以可靠覆盖该任务的搜索空间。

该命题中的“不能可靠覆盖”是条件性的：它只指向当前任务边界、工程环境、模型、Agent、工具、上下文、审计与修复协议及资源预算的组合，不意味着该任务原则上不可解决，也不意味着同一模型在改变系统条件后仍然不可解决。

3.3 论证

在上述定义与前提下，可通过反证法说明 L1 的条件性诊断关系。假设一个满足有效 MRAC 序列条件的任务持续不收敛，同时假设当前系统能够在既定资源边界内可靠覆盖其搜索空间。根据可靠覆盖的定义，系统应能在预算内识别关键条件、排除决定性错误、解决已发现问题并完成验证，将可能破坏交付的不确定性压缩到风险阈值之内。

有效 MRAC 序列保持任务意图和审计范围稳定，且只计入真实、与交付相关的新问题。如果系统具备上述覆盖能力，并且有效 MRAC 序列能够持续观测这些覆盖工作，那么随着问题被识别和修复、错误路径被排除，审计轨迹应在预算内进入与风险阈值相符的稳定区间。该轨迹不必单调下降，也允许中途发现更深层问题，但不能在达到预定边界后仍持续出现新的中高严重度或结构性问题，也不能让已经解决的问题持续复发。

持续不收敛表明至少一项覆盖工作没有稳定完成：关键条件仍无法识别，重要失败路径持续逃逸，修复结果无法保持，修复不断引入新的交付风险，验证不足以支持稳定判断，或所需资源已经超出预算。这与“当前系统能够可靠覆盖该任务搜索空间”的假设矛盾。因此，在上述定义和前提成立时，可将持续不收敛作为当前系统—任务组合尚未进入可靠覆盖区间的条件性诊断信号。L1 的成立依赖于这些操作性定义及有效 MRAC 序列条件。

3.4 理论意义与边界

可靠覆盖由前述系统—任务关系独立定义，MRAC 不收敛则是可观察的过程状态；L1 不直接用审计结果定义覆盖状态，而是将不可直接观测的系统—任务匹配关系连接到可记录的审计—修复轨迹。工程系统不必预先获得任务的完整正确答案，也不必绝对量化搜索空间，便可以依据不收敛识别当前覆盖边界。

不收敛可能来自任务规模与边界、工程状态、模型与 Agent、上下文、审计器、修复过程、验证条件或资源预算。L1 不区分具体原因，只判断当前覆盖能力 (C) 与搜索空间 (S) 尚未可靠匹配，并据此触发两类干预：增强覆盖能力，或治理搜索空间。收敛能够支持何种覆盖判断及其外推边界，由 L2 进一步讨论。

4. L1 的初步工程映射

前提论文记录了两个内部软件工程案例 [2]。本文不重复完整案例，仅考察其与 L1 条件的对应关系。

案例	关键轨迹	L1 相关观察
A5PR1 Spec	R2 至 R14 持续出现 P1、P2；R3 曾单轮无新增问题，但后续恢复高严重度问题	单轮干净结果未形成稳定覆盖；原任务停止并进一步拆分

案例	关键轨迹	L1 相关观察
B2PR-01 Spec 与 PR	Spec 在 R3 暂时干净，R6 再次出现 P1，修复后 R7、R8 连续干净；需要区分 PR 修复后连续多轮无新增问题	偶然干净轮次与可复现稳定区间

案例 A5PR1 表明，单轮干净不能推翻后续持续不收敛的轨迹，也不足以支持继续推进。案例 B2PR-01 则表明，可复现的稳定区间比单轮结果提供了更多覆盖证据，但这种证据能否代表完整交付相关搜索空间，仍需独立验证和长期结果检验。

后续公开实验应锁定任务基线与审计条件，保留原始审计、修复、去重、严重度和漂移记录，以检验有限轨迹能否稳定用于 L1 判断。

5. MRAC L1–L5 递进结构

5.1 总体分层

层级	核心命题	命题性质	主要验证要求
L1 不收敛诊断	持续不收敛表明当前系统不能可靠覆盖任务搜索空间	条件性诊断命题	论证条件结构；验证现实序列满足条件
L2 收敛覆盖	有效收敛表明已暴露失败路径得到更充分的处理与验证，并为当前审计配置下的局部覆盖趋于饱和提供正向证据	已暴露路径的处理与验证改善可由过程直接支持；局部饱和及其外推范围需经验验证	确定不同审计配置的覆盖范围、共同盲区、漂移风险、交付可靠性外推与审计 ROI
L3 拆分恢复	研究不收敛工程任务能否经搜索空间治理和递归拆分恢复收敛，并提高交付可靠性	可实验的工程机制	配对案例、对照实验、独立交付结果与组合验证
L4 工程自主推进	常见工程活动可共同转化为可审计、可修复、可拆分、可组合的任务图	工程 AGI 核心假说	完整项目长期无人化推进及其必要条件验证
L5 跨领域扩展	MRAC 类递归验证与治理机制可扩展到工程以外的重要智能任务	长期 AGI 研究纲领	跨领域验证任务能否形成明确中间结果、接受审计和修复并继续拆分

各层具有独立的验证要求，低层成立不自动推出高层成立。

6. L2：收敛、当前审计覆盖范围与经验验证

6.1 L2 命题

L2：收敛覆盖判断。

在任务意图与审计对象稳定、审计有效执行、已发现问题得到解决、修复结果能够保持且

审计与修复漂移受控的条件下，MRAC 收敛表明已暴露失败路径得到更充分的处理与验证，并为当前审计配置下的局部覆盖趋于饱和提供正向证据。该证据对完整交付相关搜索空间的代表程度，以及由此能够支持的交付可靠性判断强度，需要通过证据持续验证。

这里的**当前审计覆盖范围**，指在给定模型、Agent、工具、上下文、审计方向、审计提示、验证方式和资源预算下，该配置在重复且有效的执行中能够稳定进入、区分并验证的交付相关条件与失败路径。它只是完整交付相关搜索空间中当前配置可能覆盖的部分，不能由单次收敛序列直接观测，只能通过重复实验、差异化审计和后续遗漏等证据估计。

6.2 理论可支持的范围与经验外推

MRAC 通过发现和解决问题，继续搜索尚未暴露的失败路径。如果修复通过验证且没有因后续修改或漂移而失效，每轮有效循环都会使已暴露失败路径得到更充分的处理与验证。因此，收敛序列可以直接支持一个较强结论，并为另一个判断提供正向证据。

第一，**已暴露路径得到更充分的处理与验证**。与初始对象及中间未收敛状态相比，系统已经识别、排除或验证了更多已暴露的交付相关失败路径。该结论来自审计—修复过程本身，不依赖事后观察到交付成功才成立。

第二，**收敛为局部覆盖趋于饱和提供正向证据**。在预设协议、有效轮次与风险阈值下，如果新增实质问题逐步减少并稳定停止，说明当前配置在该序列中已难以继续暴露新的主要失败路径，可以作为局部覆盖趋于饱和的正向证据；但不能仅由有限收敛轨迹严格推出潜在可发现路径已经接近耗尽。其证据强度取决于审计轮次、搜索多样性、配置差异和历史遗漏率。这同样不表示完整交付相关搜索空间已被充分覆盖，也不表示结果必然正确。

6.3 共同盲区与覆盖扩展

多轮审计可能存在共同盲区。系统即使在可观察范围内收敛，仍可能遗漏某些决定性交付风险；收敛后出现新的重大遗漏，说明当前审计配置存在能力边界。当前审计覆盖范围越能代表完整交付相关搜索空间，收敛能够支持的交付可靠性声明就越强。

交叉使用不同模型、审计方向、审计提示、工具和外部验证，可以改变搜索路径并扩大覆盖。只有不同方法能够稳定暴露不同类型的问题时，多样化才真正降低共同盲区；同质模型和提示的重复调用可能增加轮次，却不显著增加覆盖。后续研究需要确定不同配置能发现哪些失败类型、彼此的搜索路径有多相似，以及新增审计方式能减少多少收敛后遗漏。

6.4 漂移、可靠性外推与审计 ROI

审计与修复漂移决定收敛序列是否仍对应同一个任务。如果修复改变原始意图、缩小验收范围，或通过副作用破坏已经解决的问题，系统可能对改变后的对象产生表面收敛。理论上，任务与审计对象保持可比、修复结果能够保持，是 L2 成立的必要条件；工程上仍需研究漂移的发生概率、识别方法、预防机制及其成本。

L2 的经验研究集中于四类问题：不同审计配置的覆盖边界；共同盲区及其降低方式；审计和修复漂移的发生与控制；收敛轨迹对真实交付可靠性的预测强度。隐藏测试、差异化独立审计、实际运行以及交付后的缺陷、返工和回滚，都可以用来验证这些关系。

审计效率同样属于核心工程问题。审计 ROI 应同时考察每单位 Token、时间或计算资源带来的有效问题发现、风险解决和最终返工减少。随着轮次增加，边际覆盖收益可能下降；何时继续原协议、何时更换模型或审计方向、何时停止，以及何时依据 L1 触发拆分，需要在覆盖收益、交付风险与资源成本之间持续优化。

7. L3：通过递归拆分恢复收敛

L3：拆分恢复假说。

对一类超出当前可靠覆盖范围的常见工程任务，通过任务边界调整、上下文隔离、依赖治理和递归拆分，有可能将原任务转换为一组处于可收敛区间的子任务；在子任务与集成对象均完成有效收敛和独立验证后，整体交付可靠性有望得到提高。

L3 把 L1 从诊断机制推进为恢复机制。L1 只说明当前组合不匹配；L3 进一步研究能否主动改变搜索空间，使任务重新进入覆盖范围。

模块分解是软件工程的经典治理手段，分解标准会直接影响模块之间的信息依赖和后续修改成本 [10]。L3 并不重新提出模块化原则，而是增加一种动态触发与验证机制：以不收敛识别当前拆分边界失效，再以子任务和集成对象能否重新收敛检验新边界是否有效。

一个能够支持 L3 的案例必须包含完整链条：

原任务不收敛 → 明确失配边界 → 重新拆分或治理 → 子任务分别收敛 → 集成对象重新收敛 → 独立结果验证改善

仅仅观察“拆分后子任务更容易完成”还不够。拆分可能把风险转移到依赖、接口或集成阶段；所有子任务分别收敛，也不保证组合后的项目状态收敛。因此，集成本身必须被视为新的工程对象。如果集成不收敛，系统应继续定位跨边界问题、调整拆分或回退，而不能将子任务的局部稳定直接等同于整体可靠性。

L3 可以通过同一初始任务的配对实验验证：比较在原边界内继续修复与达到不收敛阈值后重新拆分两条路径，并以最终残留缺陷、集成结果、成本和交付时间评价治理收益。数量有限但链条完整且可复现的案例，可以证明该机制在现实工程中存在；其普遍性则需要跨任务、跨项目的累积证据。

8. L4 与 L5：从完整工程自主推进到跨领域扩展

8.1 L4：完整工程自主推进

为明确本文的讨论范围，本文将工程 AGI 定义为：AI 能够在仅接受必要外部目标、授权与现实反馈的情况下，无需人类承担工程过程，完成原本需要普通人类团队长期协作的复杂项目。仅证明代码实现可以递归拆分并不足够，完整工程交付还包括意图结构化、需求澄清、产品与技术决策、任务规划、实现验证、集成发布和长期演化。L4 研究这些活动能否共同转化为可审计、可修复、可拆分并可重新组合的任务图。

L4 的前提不是预先写出全部实现细节，而是目标、约束、偏好、决策权限和反馈接口足以支持系统继续推进；暂时缺失的信息能够被识别并转化为检索、实验、澄清或授权任务。L4 成立需要满足五项条件：不收敛任务能够继续拆分或重构；阶段性结果和关键决策能够接受审计；子任务能够重新组合，集成失败仍能返回处理流程；状态与证据能够跨会话保存；缺失的意图能够被识别，并明确交由人类或外部系统处理。

如果这些条件在常见工程活动中成立，系统便不必一次覆盖完整项目，而可以通过局部覆盖、保存阶段性状态与组合审计持续推进。L4 的验证对象因此是完整项目：系统能否长期自主推进任务图，在可靠交付的同时持续减少人类在任务拆分、状态追踪和局部修复中的过程性介入。

8.2 L5：跨领域扩展

L5：跨领域扩展命题。

对于能够被转化为具有持续状态、可审计中间对象、可执行修复、可获得反馈并允许递归拆分的过程的 AI 任务，可以研究建立 MRAC 类机制的可能性：以不收敛识别当前覆盖边界，以收敛积累正向证据，并通过递归拆分与验证持续提高任务成功率。若这一转换能够覆盖足够广泛的重要智能活动，MRAC 的研究路线将从工程 AGI 延伸至更广义 AGI。

L5 的关键问题，是工程以外的任务能否同样形成明确、可检查的中间结果，接受审计和修复，继续拆分和组合，并获得有效反馈。软件工程在这些方面条件较好；研究、战略、创作和现实行动则可能面对开放目标、延迟反馈、不可逆后果和价值冲突。L5 是否成立，取决于这些领域能否建立能够发挥相同作用的状态记录和验证方式。

9. 理论命题的经验验证与高层假说的可证伪性

经验工作的核心，是判断现实序列是否满足 L1 与 L2 的适用条件，并确定有限轨迹能够支持多强的工程判断。对 L1，需要确定有限轮次何时足以判断“持续不收敛”；对 L2，则需要建立不同审计配置、收敛轨迹与覆盖声明之间的经验关系。成功和失败案例都应进入证据集，用于识别能力边界。

L3-L5 提出了更强的恢复、完整工程自主推进和跨领域扩展主张，可能因现实证据而被削弱、受到适用范围限制，乃至被反驳。如果拆分不能改善整体交付，工程活动不能形成持续递归推进的完整流程，或重要非工程任务无法获得必要的状态与反馈结构，相应高层命题的适用范围就必须收缩，甚至应予否定。

10. 后续研究与工程方向

近期研究应优先建立统一的轨迹记录与结果验证方法，在不同模型、审计方向、审计提示、工具和验证方式之间进行比较，并将收敛轨迹与真实交付结果关联起来。在此基础上，可以围绕 L3 积累“不收敛—拆分—重新收敛”的配对证据，比较不同治理方式对搜索空间、集成风险、成本和交付时间的影响。

中期目标是建立支持长期状态管理、递归拆分、组合审计与回退的完整项目原型，使 MRAC 从离线观测信号转化为能够驱动任务继续、停止、切换审计策略、拆分和回退的工程控制机制。只有在完整工程自主推进获得充分证据后，才有必要系统检验 L5 在其他领域中的适用条件。具体基础设施架构不在本文展开。

11. 结论

本文建立了 MRAC 的理论起点与 L1-L5 递进研究纲领，并将不同层级的主张按证据强度区分开来：L1 提供条件性诊断，L2 提供有限范围内的正向覆盖证据，L3-L5 则依次进入需要实验和长期证据支持的拆分恢复、完整工程自主推进与跨领域扩展。由此，MRAC 不必以高层假说预先成立为前提，而可以从可观测轨迹出发，沿不同证据层级逐步推进研究与工程验证。

参考文献

- [1] 刘明. 从 SCR 元关系到搜索空间治理: AI 软件工程交付可靠性的基础机制 [EB/OL]. Zenodo, 2026. DOI: [10.5281/zenodo.21902420](https://doi.org/10.5281/zenodo.21902420).
- [2] 刘明. 多轮审计收敛: 面向工程 AGI 的外部验证机制与递归任务闭环 [EB/OL]. Zenodo, 2026. DOI: [10.5281/zenodo.21804751](https://doi.org/10.5281/zenodo.21804751).
- [3] MADAAN A, TANDON N, GUPTA P, et al. Self-Refine: Iterative refinement with self-feedback[C]// Advances in Neural Information Processing Systems 36. 2023. DOI: [10.52202/075280-2019](https://doi.org/10.52202/075280-2019).
- [4] SHINN N, CASSANO F, GOPINATH A, NARASIMHAN K, YAO S. Reflexion: Language agents with verbal reinforcement learning[C]//Advances in Neural Information Processing Systems 36. 2023. DOI: [10.52202/075280-0377](https://doi.org/10.52202/075280-0377).
- [5] HUANG J, CHEN X, MISHRA S, et al. Large language models cannot self-correct reasoning yet[C]//The Twelfth International Conference on Learning Representations. 2024. URL: <https://openreview.net/forum?id=lkmD3fKBPQ>.
- [6] YANG J, JIMENEZ C E, WETTIG A, et al. SWE-agent: Agent-computer interfaces enable automated software engineering[C]//Advances in Neural Information Processing Systems 37. 2024. DOI: [10.52202/079017-1601](https://doi.org/10.52202/079017-1601).
- [7] ZHENG L, CHIANG W L, SHENG Y, et al. Judging LLM-as-a-judge with MT-Bench and Chatbot Arena[C]//Advances in Neural Information Processing Systems 36. 2023: 46595–46623. DOI: [10.52202/075280-2020](https://doi.org/10.52202/075280-2020).
- [8] CALBOREANU E. Iterative audit convergence in LLM-managed multi-agent systems: A case study in prompt-engineering quality assurance[J]. Software, 2026, 5(2): 26. DOI: [10.3390/software5020026](https://doi.org/10.3390/software5020026).
- [9] NEWELL A, SIMON H A. Computer science as empirical inquiry: Symbols and search[J]. Communications of the ACM, 1976, 19(3): 113–126. DOI: [10.1145/360018.360022](https://doi.org/10.1145/360018.360022).
- [10] PARNAS D L. On the criteria to be used in decomposing systems into modules[J]. Communications of the ACM, 1972, 15(12): 1053–1058. DOI: [10.1145/361598.361623](https://doi.org/10.1145/361598.361623).
- [11] AVIŽIENIS A, LAPRIE J C, RANDELL B, LANDWEHR C. Basic concepts and taxonomy of dependable and secure computing[J]. IEEE Transactions on Dependable and Secure Computing, 2004, 1(1): 11–33. DOI: [10.1109/TDSC.2004.2](https://doi.org/10.1109/TDSC.2004.2).