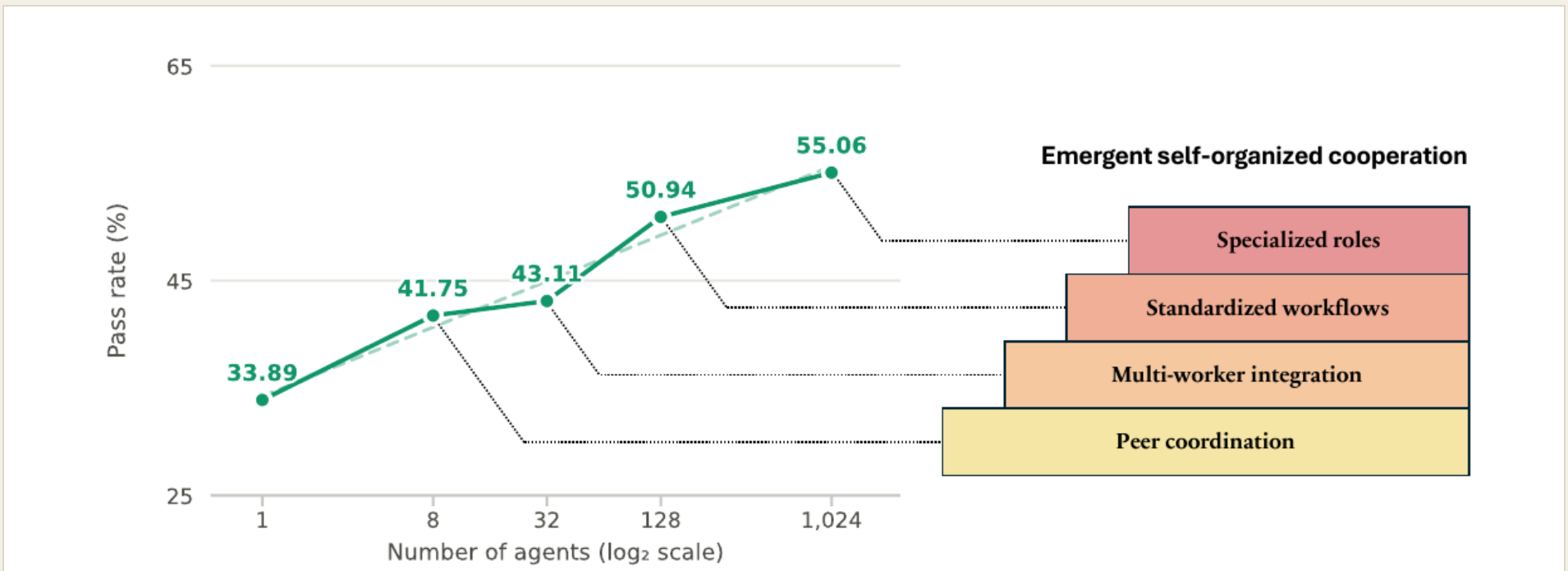
论文导读　微软研究院 · Agensh

多个 AI 一起写代码，怎么分工？

1,024 个 Agent 重建程序的实验
拆解任务分工、查错、合并与扩容收益。



论文图1 · 重建 pandoc 的最终测试通过率

任务分工　共享发现　同伴查错　合并代码

13 页原文精读 · 看懂做法，也看清实验边界

它们在重建什么程序

给 Agent 一个能运行的参考程序，让它们观察输入输出，再从零写出行为相近的实现。这就是论文采用的 ProgramBench 任务。

01	→	02	→	03
运行参考程序		观察输入和输出		编写并验证实现

允许	运行参考程序，读取仓库文档和素材。
禁止	读取程序字节、反编译、联网下载原实现。
评估	6 小时后，用隐藏测试检查最终仓库。

从 200 个任务中选取 5 个最难的任务。结果适用于这组软件重建实验，不能直接外推企业应用。

千 Agent 提高了多少

19.31 → 28.78%
五任务均值 · 1 → 128 个

33.89 → 55.06%
pandoc · 1 → 1,024 个

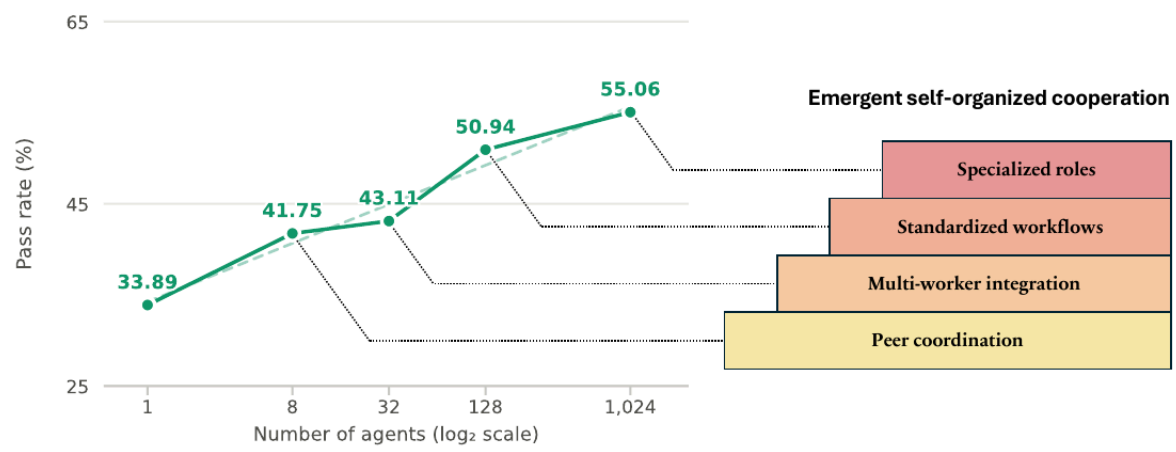


图1原图局部 · 1,024 仅是 pandoc 实验

五任务均值增加 9.47 个百分点，相对提升约 49%。千 Agent 那组增加 21.17 个百分点，但只报告了 pandoc——一个文档转换程序。

模型均为 GPT-5.6-sol（high），底层使用 Copilot，时限均为 6 小时。比较没有固定总 token 或总费用。

通过率是 canonical-kept 测试指标；55.06% 不能解释为完整软件功能已经实现了一半。

增加人数，值不值

+4.12

pandoc 128 → 1,024 的增量
单位：百分点

未报告

同总 token / 费用预算
下的收益对照

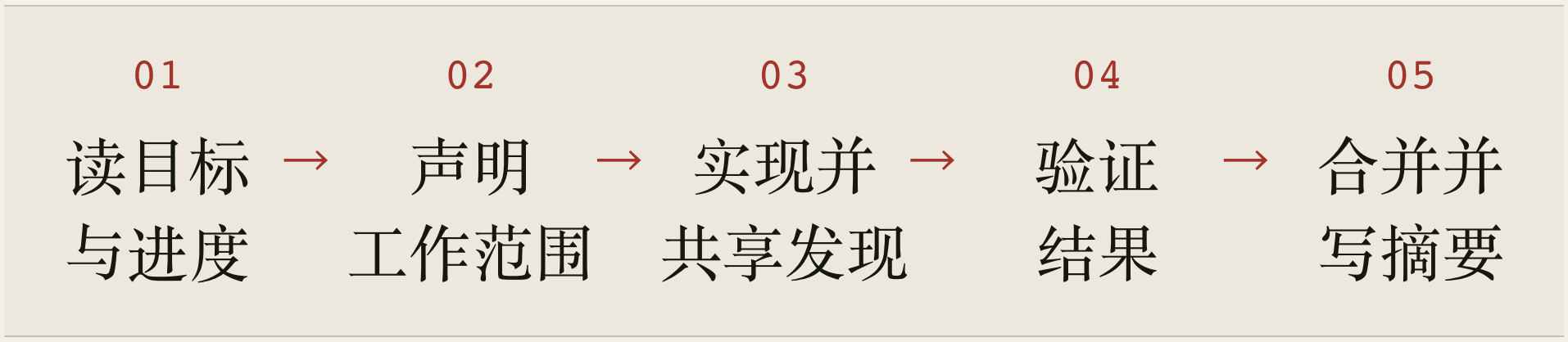
人数上限增加 8 倍，最终通过率从 50.94% 到 55.06%。这是一次具体实验的增量，不能换算成“成本贵了 8 倍”。

已有结果	相同时限下，部分任务更快达到较高通过率。
仍需区分	提升来自更多并行试错，还是协作机制本身?

Yu Cao 的公开讨论提出：在同总 token 预算下，与独立 Agent 的择优结果、简化共享机制做对照。这是待验证的问题。

任务由谁来分

Agensh 没有中央编排 Agent。每个 Worker 读取总目标、进度和同伴发现，自己提出子任务；认领重叠时，直接找同伴协调。

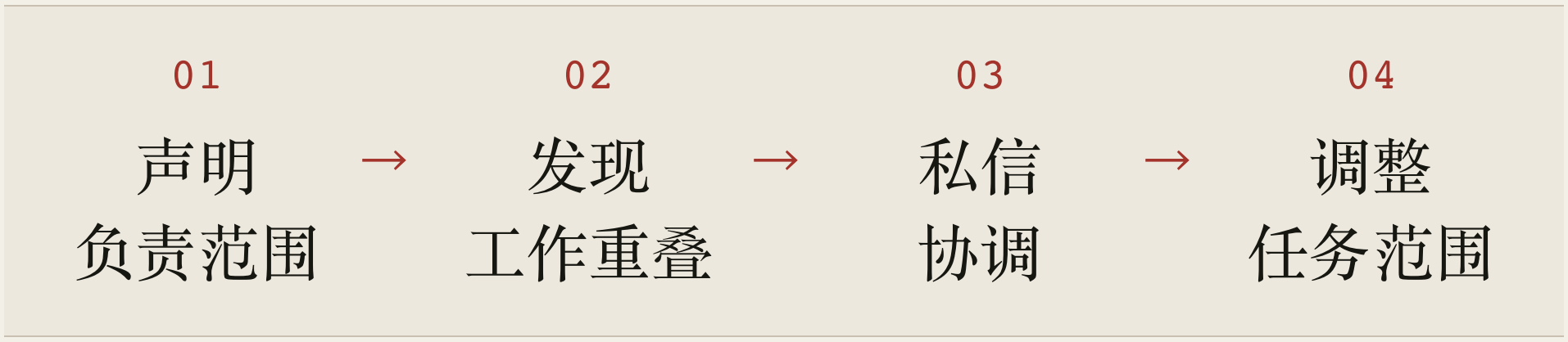


循环是异步的，不必等所有人做完一轮。Worker 初始协作指令相同，仅 ID 不同；后面的评审和集成分工是在工作中协商出来的。

这套做法依赖共享仓库、消息接口和发现记录。自主选任务仍有明确的协作指令与基础设施。

怎样避免重复工作

FFmpeg 的 8 Agent 轨迹里，两位 Worker 发现认领范围重叠。沟通后，其中一位调整范围，去做互补工作。



CLAIM 记录的是正在做什么。同伴先读到这个声明，才有机会换方向，或者商量接口怎么切。

论文没有报告严格排他锁或租约。对实际项目，我会把涉及的文件、接口和资源写进认领范围；任务名称不同，仍可能改到同一处。

试错结果怎样共享

共享记录区分观察、已确认结论、失败、认领和代码变更。Worker 指令要求把证伪的想法写成 FAIL，让队友避开已知失败路径。

OBSERVED 观察　FACT 确认结论　FAIL 失败　CLAIM 认领　PATCH_SUMMARY 改动

FAIL

项目建议：失败记录应能回答

试了什么输入，出现什么结果?

对应哪个版本，证据在哪里?

条件变化后，是否需要重测?

上面三项是项目建议。记录被标为 FACT，并不自动保证正确；能追到证据和适用条件，才方便复查。

队友的消息何时收到

分工会变，旧消息也会过期。Agensh 将普通更新和优先消息分开送达。

下一轮开始时

仓库与频道的普通更新

先排队，在 Worker 下一轮开始时加入输入。

工具返回时

私信与新共享记录

基础设施 / MCP 工具返回时，追加进工具结果。

它能在一轮工作中送达新信息，但不会停止任意运行中的程序。长命令如何取消、任务何时失效，需要执行系统另行设计。

已经批准，还能撤回

PHP-src 的 32 Agent 轨迹里，几位同伴已经批准一项贡献。另一位找到具体反例，原批准随即被撤回；作者修复后，重新评审，再由同伴合并。

01	02	03	04	05
已有批准	→ 找到反例	→ 撤回并修复	→ 重新评审	→ 同伴合并

论文未公开这条反例的具体输入和代码。从轨迹能确认的是：这次评审允许用反例推翻已有批准，再修复和复查。

用于项目时，我会要求评审对应提交版本、测试基线和运行证据。只读作者的成功总结，很难判断行为是否真的正确。

代码最终由谁合并

Worker 在各自目录和分支开发，再合回主分支。128 配置的 pandoc 轨迹里，作者先更新、测试分支，再交出提交哈希。后来改由同伴接手更新、测试、检查、合并全流程。

作者
实现与交接代码
→
集成者
更新 · 测试 · 检查 · 合并

1,024 配置中的交接方式

01 联系候选集成者 → 02 选首个有效回应者 → 03 取消其他请求 → 04 选定后交接代码

1,024 配置出现了多位集成者。同领域 Worker 也曾在同伴尝试失败后接手工作。

这些是论文观察到的轨迹。它没有证明到了某个人数，团队就必然长出某种岗位。

交付看哪个版本

评测从仓库 HEAD 导出源码，删除已提交的可执行文件，再重新构建。私有目录里能跑，并不足以证明最终仓库能交付。

01 导出仓库 HEAD

02 删除现成可执行文件

03 运行根目录 compile.sh

04 生成 ./executable

05 执行隐藏测试

根目录缺少 compile.sh 得 0 分。脚本必须生成 ./executable，不能只给一个现成文件加执行权限。

项目验收也可以从指定提交的干净目录开始，重建后跑回归测试。这是我建议采用的交付检查。

扩容后的任务差异

图5完整成绩，单位为%。高亮表示比上一档下降；这五个任务的比较只到 128 个 Agent。

Agent 配置规模 / 最终测试通过率（%）

任务	1	8	32	128
FFmpeg	6.54	11.97	13.22	**12.89**
gromacs	19.69	**19.37**	19.37	22.12
pandoc	33.89	41.75	43.11	50.94
PHP-src	10.63	11.86	12.23	12.35
ctags	25.82	**18.47**	44.69	45.62

ctags 1 → 8：下降 7.35 个百分点；gromacs 1 → 8：下降 0.32；FFmpeg 32 → 128：下降 0.33。

这些下降的原因没有被实验单独识别。均值上涨，仍不代表每项任务、每次扩容都会受益。

怎样做小规模对照

如果要借用这套做法，我会先比较 1 / 4 / 8 个 Agent，固定模型、工具、输入和验收测试。

01 先看截止时间内的结果

相同时限比较通过率、达标用时和实际消耗。

02 同预算比较独立与协作

相同总预算下，独立 Agent 用验证集择优，再与协作结果在留出测试集上比较；选择结果的消耗也计入预算。

03 记录协调代价

重复工作、冲突、返工和集成等待。多次运行，观察波动。

这样才有机会分清：并行多试几条路带来了什么，共享记录和协作又增加了什么。

什么项目值得试

任务能拆出清楚的接口，结果能自动测试，贡献能统一集成；这类项目值得先做小范围试验。

主 Agent	保留目标、权限、预算和最终交付确认。
执行者	自主认领，交接时附范围、版本与验证证据。
集成者	检查合在一起的结果，重建、回归，再确认交付。

这是项目采用建议；论文的实验没有中央编排 Agent。强耦合、重复劳动或合并积压，值得先查接口与验收，再决定是否扩容。

任务认领、共享证据、同伴评审与干净构建，可以先在小项目里逐项试验。后面补充运行细节与阅读来源。

摘要、检索和重试

100 字符	300 字符	2,000 条
普通条目上限	改动摘要上限	近期记录上限

查详情	长说明放 detail；board_unfold 展开，board_grep 检索完整历史。
送消息	事件进持久队列，按 ID 去重；会话执行完成后标记已发送。
遇失败	延后重试。路由器重启后，未完成事件重排，尽可能恢复会话。

闲置 10 分钟会收到继续工作的提示；一轮结束没发布共享记录，也有跟进提醒。

事件去重不保证业务写入幂等。项目还应另测重复写入、预算耗尽和停止条件；这些是工程建议。

人数与时间怎样计算

前1小时	每 30 秒启动 1 个 Agent；之后每 3 秒启动 1 个。
统一截止	从首个 Agent 启动算起，6 小时后结束；1,024 配置为 16 节点 × 64 Worker。
收尾提醒	提前 45 分钟、5 分钟，提醒优先集成并确认默认分支能构建。

pandoc 超过 30% 的检查点：128 配置在 30 分钟，32 配置在 60 分钟，8 配置在 90 分钟；单 Agent 最初 2 小时仍低于 30%。

检查点不是精确首次达标时间，配置规模也不是当时在线人数。按启动节奏估算，千 Agent 约 1 小时 45 分钟启动齐；这是近似推算。收尾指令是提醒，不能当成强制冻结。

原文与证据范围

Agensh: Scaling Organizational Intelligence to 1,024 Agents

微软研究院 · arXiv:2609.26781v1 · 13 页

论文原文 PDF

实验、图表、Worker 指令与附录。

官方项目页

原图、论文入口与协作演示。

Yu Cao 的公开讨论

同预算下比较协作与并行试错。属于读者观点，学习时可作为待补对照。

论文未提供同规模中央编排、组件消融、同成本收益和重复运行方差。本次未审计框架实现或复现。

证据字段、认领租约、过期写入拒绝、权限隔离和主 Agent 交付确认，均为工程建议。

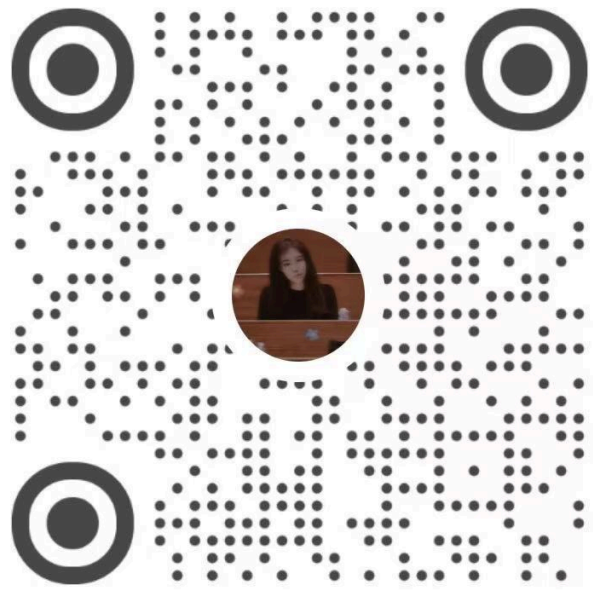
继续学习与交流

前面 17 页是论文导读与项目试验建议。
需要继续交流，可以从下面两个入口找到我。

我的 Agent 助理

扫码进入，或点击二维码

小红书 · 心晴九天



小红书号：howru999

阅读时可以留意自己的项目：哪些任务能并行，哪些结果能复查，谁负责把贡献合起来?

论文成绩、公开讨论与项目建议已分开标注。试验效果需要在具体任务中验证。