CookMAS

2026.02.05

김명진

Collab-Overcooked: Benchmarking and Evaluating Large Language Models as Collaborative Agents

Haochen Sun[1*], Shuwen Zhang[1*], Lujie Niu[1], Lei Ren[2], Hao Xu[2], Hao Fu[2], Fangkun Zhao[1], Caixia Yuan[1†], Xiaojie Wang[1]

[1]Beijing University of Posts and Telecommunications, [2]Li Auto Inc.
{haochen_sun, zhangshuwen2023, lujien, yuancx, xjwang}@bupt.edu.cn,
{renlei3, fuhao8}@lixiang.com, kingsleyhsu1@gmail.com

EMNLP 2025

Motivation & Contribution

Motivation

- LLM-MAS가 복잡한 실제 작업을 효과적으로 수행하려면 단순히 목표를 해석하는 것을 넘어 세 가지 필수 협업 능력이 필요함:

  1. Competence Boundary Awareness → 혼자 못 푸는 상황을 인식하고 도움을 요청할 수 있는가?
  2. Communication → 필요한 정보를 명확한 프로토콜로 전달할 수 있는가?
  3. Dynamic Adaptation → 상대의 요청에 맞게 행동 계획을 바꿀 수 있는가?

- 기존 벤치마크들은 여러가지 한계를 갖는데:

  1. 협업이 강제되지 않고
  2. End-to-End 지표만 사용하여 중간 협업 과정에 대한 고려 없이 성공 여부만 고려하며
  3. 세분화된 평가가 부족함

Environment Settings

Collaboration Capability

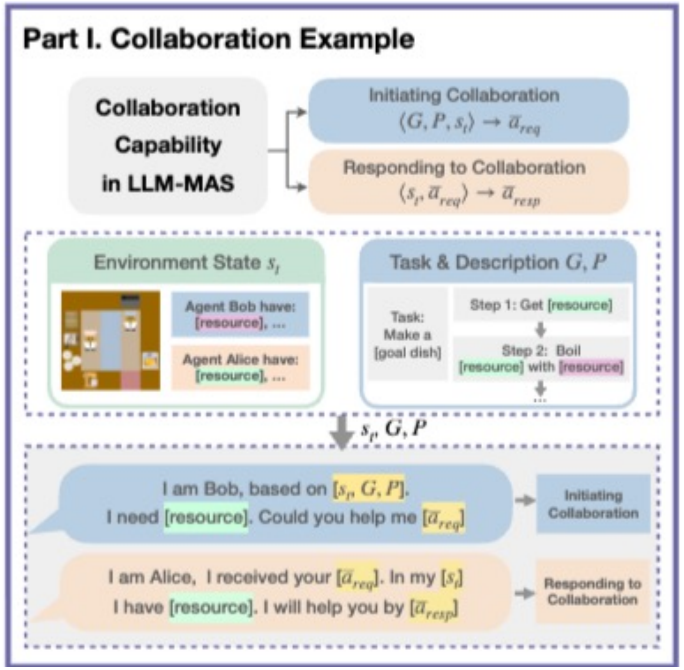
- LLM-MAS 내 task는 다음 4-tuple로 정의: $T = (G, E, P, R)$

  (G: natural language description of the task goal, E: description of the environment,
  P: optional natural language guidance (recipes, helpful hints, task constraints),
  R: Referential Action Trajectory (RAT))

- 핵심 협업 능력 정의

  1. Initiating Collaboration → 한계에 봉착했을 때 도움 요청

  2. Responding to Collaboration → 요청을 이해하고 환경 상태에 맞게 행동 수행



Environment Settings

Evaluation

- Trajectory Efficiency Score (TES)

$$TES(\overline{h_k}) = max_j \left\{ \frac{(1+\beta^2) D^j_{max}(\overline{h_k}, \bar{g}^j_k)}{m_k + \beta^2 n_k} \right\}$$

$$\overline{h}_k = \{a^1_k, a^2_k, \ldots, a^T_k\}$$

$$\bar{g}^j_k = \{g_i\}^{m_k}_{i=1} \in R$$

$$D^j_{max} = \max_d \{d \mid \forall 1 \le i_1 < \cdots < i_d \le n_k, \text{ s.t. } a_{i_1} = g_1, a_{i_2} = g_2, \ldots, a_{i_d} = g_d\} \quad (2)$$

Agent의 실제 행동 시퀀스가 최적 행동 시퀀스(RAT) 와 얼마나 잘 맞는지 측정, 순서 보존 + 중복 페널티 포함

- Incremental TES (ITES)

$$ITES(\bar{a}, \overline{h_k}) = TES(\overline{h_k} \cup \bar{a}) - TES(\overline{h_k})$$

Individual collaborative action의 중요도 평가

- Evaluation metrics

1. Progress Completeness (PC)

$$PC = \frac{1}{K} \sum_{k=1}^{K} TES(\bar{h}_k)$$

2. Initiating Capability (IC)

$$IC = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left(ITES\left(\bar{a}^{(i)}_{req}, \bar{h}_j\right) > 0\right)$$

3. Responding Capability (RC)

$$RC = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left(ITES\left(\bar{a}^{(i)}_{resp}, \bar{h}_j\right) > 0\right)$$

Benchmark

Environment

- Grid-based kitchen simulation 환경을 testbed로 사용

환경 내에 조리 관련 도구들을 (e.g. dispensers, utensils, counters, delivery location) interactive element들로 설정.

Agent들은 function format의 predefined action들로 해당 element들과 상호작용.

```
Action Space for Agent Alice:
    1. pickup(obj,place)
    2. cut(chopping_board_name)
    3. stir(blender_name)
    4. place_obj_on_counter()
    5. put_obj_in_utensil(utensil)
    6. wait(num)

Action Space for Agent Bob:
    1. pickup(obj,place)
    2. cook(pot_name)
    3. place_obj_on_counter()
    4. put_obj_in_utensil(utensil)
    5. fill_dish_with_food(utensil)
    6. bake(oven_name)
    7. deliver()
    8. wait(num)
```

Task Construction

Complexity Level	Acquiring New Ingredients	Processing the Ingredients by Agent Alice	Acquiring a New Dish	Processing the Ingredients by Agent Bob	Total Number of Collaborative Actions
Level 1	1	0	0	1	2
Level 2	1	1	1	1	5
Level 3	1	1	1	2	7
Level 4	2	1	1	2	9
Level 5	2	2	1	3	12
Level 6	3	3	1	4	17

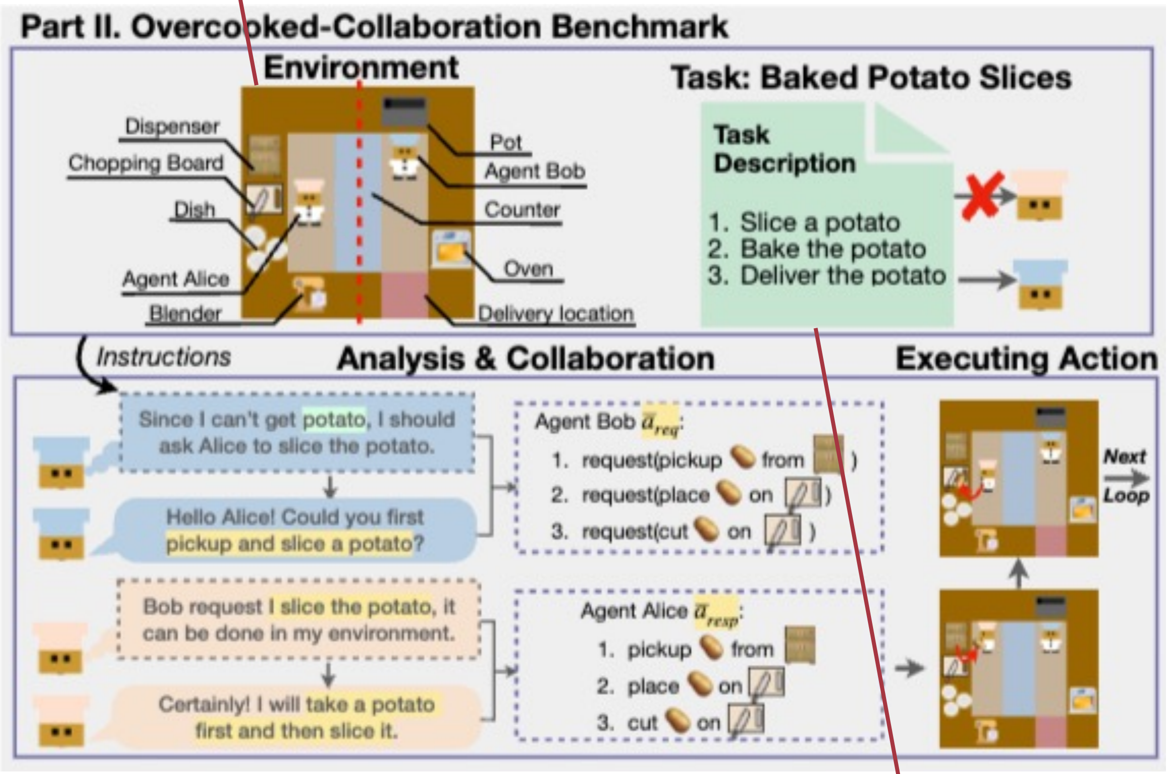
총 30개 task (recipe) 생성.

Minimum number of collaborative actions에 따라 6단계로 난이도 세분화.

Benchmark

Resource Isolation

- 각 agent는 resource-isolated sub-environment에서 활동, 상호간 interaction은 중간 counter 에서만 가능
- Collaborative dependency 강조



Asymmetric Task Knowledge

- 한 agent에게만 task completion 을 위한 정보 제공
- Agent 간 communication 강제

Experiments & Results

Benchmark Overview

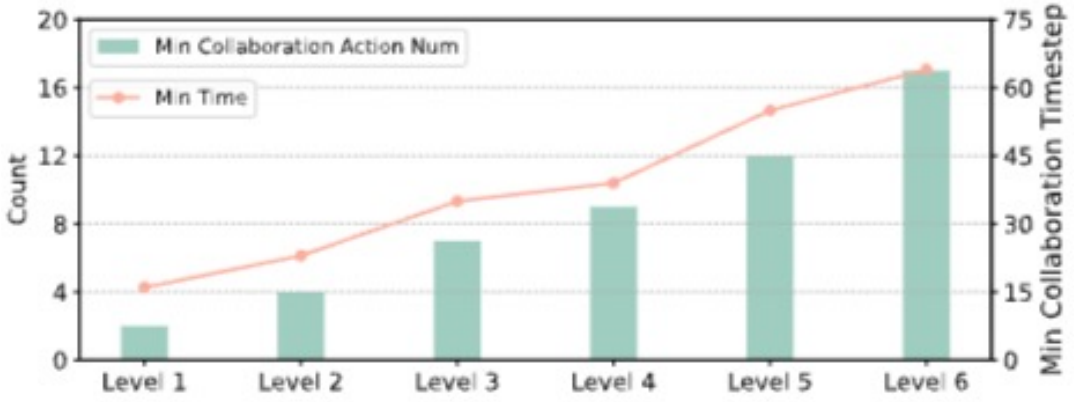
- 난이도 (min collaborative action num / min timestep)에 따라 task를 6단계로 세분화
- 2-agent 상황 가정

  Agent A (8가지 action 수행, 4가지 interactive element 접근 가능)
  Agent B (6가지 action 수행, 5가지 interactive element 접근 가능)

Models

- 다양한 size (7B ~ 671B+)의 13개 LLM을 foundation model로 사용

  Open-source: DeepSeek-R1, DeepSeek-V3, Qwen2.5 (7B, 14B, 32B, 72B), Llama-3.1 (8B, 70B)
  Closed-source: GPT-4o-1120, Claude Sonnet 4, o4-mini,
  o1-mini, GPT-3.5-turbo-0125



Experiments & Results

Task Completion Efficiency

		Level 1		Level 2		Level 3		Level 4		Level 5		Level 6	
		SR	PC	SR	PC	SR	PC	SR	PC	SR	PC	SR	PC
Closed Source	GPT-4o	94.00	85.92	86.00	84.96	68.00	76.61	34.00	44.42	2.00	29.13	4.00	22.45
	Claude Sonnet 4	**100.00**	**96.00**	**100.00**	**98.67**	**96.00**	**95.82**	**92.00**	**94.48**	**74.00**	**78.15**	**58.00**	60.69
	o4-mini	92.00	90.93	**100.00**	89.60	**96.00**	86.15	86.00	88.39	62.00	68.59	54.00	**60.79**
	o1-mini	70.00	74.18	2.00	36.36	0.00	33.60	0.00	24.80	0.00	20.28	0.00	13.07
	GPT-3.5-turbo	42.00	68.20	8.00	43.42	0.00	36.44	0.00	24.74	0.00	15.21	0.00	12.03
Open Source	DeepSeek-R1	**100.00**	**96.53**	**100.00**	**94.40**	**98.00**	**91.10**	**82.00**	**82.75**	**44.00**	**49.79**	**30.00**	**48.33**
	DeepSeek-V3	88.00	77.74	76.00	71.90	56.00	66.61	22.00	50.01	4.00	30.41	6.00	33.44
	Qwen2.5-72B-Instruct	78.00	76.84	64.00	68.00	14.00	46.88	8.00	30.80	0.00	22.67	0.00	18.45
	Qwen2.5-32B-Instruct	64.00	73.36	44.00	62.02	14.00	40.08	4.00	33.78	2.00	22.16	0.00	18.93
	Qwen2.5-14B-Instruct	32.00	50.36	4.00	26.66	0.00	24.41	0.00	19.00	0.00	14.14	0.00	14.27
	Qwen2.5-7B-Instruct	8.00	44.79	0.00	13.00	0.00	9.29	0.00	8.35	0.00	5.57	0.00	4.51
	Llama-3.1-70B-Instruct	70.00	75.42	42.00	63.15	22.00	54.58	6.00	45.04	0.00	29.77	0.00	17.69
	Llama-3.1-8B-Instruct	4.00	33.03	0.00	15.49	0.00	12.33	0.00	11.24	0.00	9.05	0.00	7.45

- Claude Sonnet 4가 최고 성능, 특히나 challenging tasks에서 독보적
- Open-source 모델들 중에선 DeepSeek-R1이 excel하지만 토큰 폭증 (GPT-4o의 18.6배)
- 고난이도 (Level 4 이상) task에선 모든 모델 성능 붕괴 -> 단순 scale-up 한계

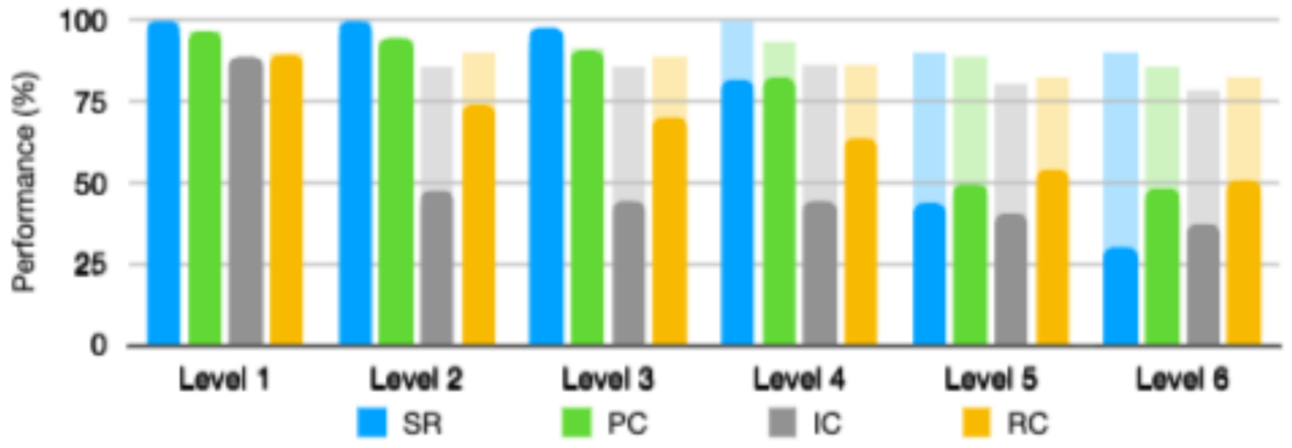
Experiments & Results

Process-Oriented Evaluation

- 대부분의 모델들에서 RC > IC 경향성이 나타남 → 요청 받으면 수행은 잘하는데, 먼저 요청은 못한다
- Task difficulty가 증가함에 따라 모든 모델의 collaboration capability가 collapse -> scale-up 한계
- Simpler tasks에서는 reasoning model들이 outperform

Experiments & Results

Human Performance Evaluation

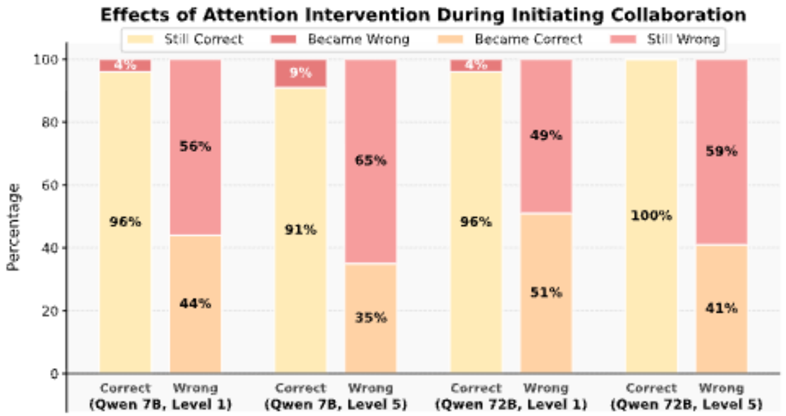


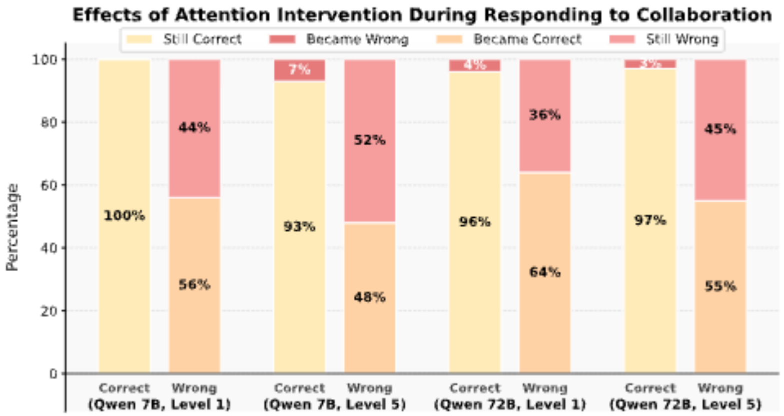
- 10 human participants에 대한 human 실험 결과를 performance ceiling으로 사용
- DeepSeek-R1 성능과 비교했을 때 human 성능이 지속적으로 높게 나옴

  High-level task abstraction이 가능한 human과 다르게 current LLM-MAS들의 shallow memory mechanisms에 의한 한계 나타냄

Experiments & Results

Analysis of Collaboration Failures





- Collaboration의 성공/실패 여부가 input prompt의 attention distribution과 밀접히 연관된다는 사실 발견

Case) Initiating Collaboration
- +) Collaborative rules
- -) Recipe information

Case) Responding to Collaboration
- +) Environmental observations, collaboration rules
- -) Partner instructions

Conclusion

Collab-Overcooked Benchmark

- A framework evaluating LLM-MAS collaboration from end-to-end and process-oriented perspectives
- 모델의 reasoning 능력 향상, Experience Abstraction, Attention-constraints 등에 대한 future work 제시

Case) Responding to Collaboration

+) Environmental observations, collaboration rules

-) Partner instructions

ROBOTOUILLE: AN ASYNCHRONOUS PLANNING BENCHMARK FOR LLM AGENTS

Gonzalo Gonzalez-Pumariega*, Leong Su Yean, Neha Sunkara, Sanjiban Choudhury
Cornell University

ICLR 2025

Motivation & Contribution

Motivation

- 최신 LLM들은 short-horizon single-agent environment에서는 impressive reasoning, task planning capabilities를 보임.

- 그러나 실제 세계에서의 의사 결정은 이런 environment들보다 훨씬 복잡한 경우가 많음

1. Time delays
2. Diverse long-horizon tasks
3. Multiple agents

=> Asynchronous planning의 필요성!

ROBOTOUILLE

Task Definition

- ROBOTOUILLE task를 time-delayed effect가 있는 MDP로 설계

  $M = < S, A, T, R >$

  State $(s \in S)$: $s = (\hat{s}_t, H_t)$ ($\hat{s}_t$: observable state elements, H_t: set of timer variables)

  Action $(a \in A)$: grounded action

  Transition function $(T: S \times A \rightarrow S)$: returns next state $s' = (\hat{s}_{t+1}, H_{t+1})$ based on given s and a

  Reward function $(R: S \rightarrow \{0, 1\})$: defines the goal of a given task

ROBOTOUILLE

JSON Example

```
"predicate_defs": [{
  "name": "istable",
  "param_types": ["station"],
  "language_descriptors": {
      "0": "{0} is a table"
  }},{
  "name": "item_on",
  "param_types": ["item", "station"],
  "language_descriptors": {
    "0": "{0} is directly on top of {1}",
    "1":
    ↪  "{1} has {0} directly on top of it"
}},...]
```

(a) Predicate Definitions

```
"name": "move",
"precons": [{
  "predicate": "loc",
  "params": ["p1", "s1"],
  "is_true": true
}, ...],
"immediate_fx": [{
    "predicate": "loc",
    "params": ["p1", "s2"],
    "is_true": true
  }, {
    "predicate": "loc",
    "params": ["p1", "s1"],
    "is_true": false
  }, ...],
"sfx": [],
"language_description":
↪   "Move {p1} from {s1} to {s2}"
```

(b) Action Definitions

```
"sfx": [{
  "type": "conditional",
  "param": "i1",
  "conditions": [{
    "predicate": "item_on",
    "params": ["i1", "s1"],
    "is_true": true
  }],
  "fx": [{
    "predicate": "iscooking",
    "params": ["i1"],
    "is_true": true
  }],
  "sfx": [{
    "type": "delayed",
    "param": "i1",
    "fx": [{
      "predicate": "iscooked",
      "params": ["i1"],
      "is_true": true
    }, {
      "predicate": "iscooking",
      "params": ["i1"],
      "is_true": false
    }],
    "sfx": []
  }]
}]
```

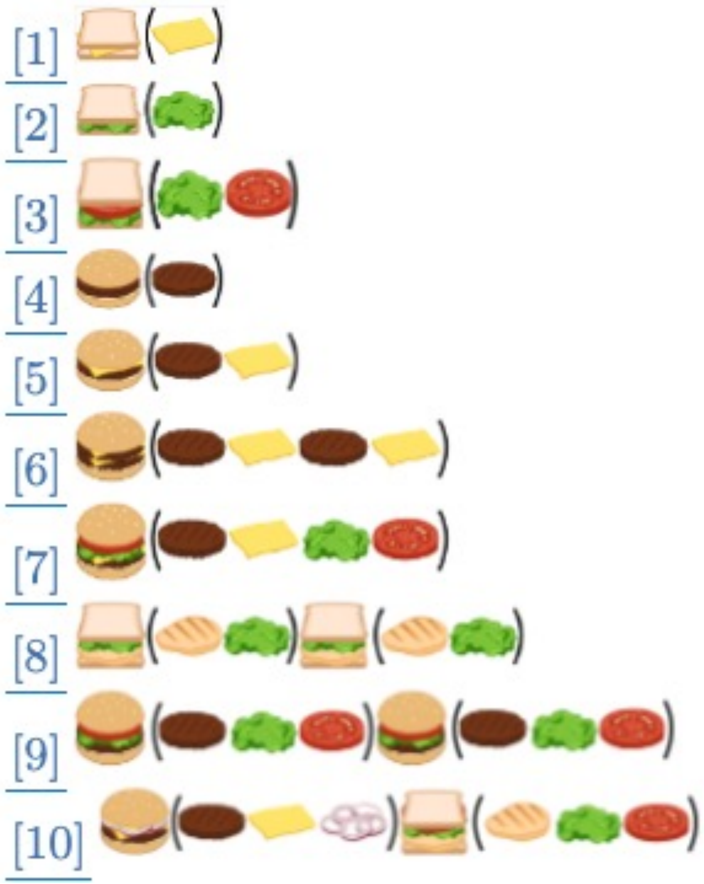
(c) Nested special effects for 'cook' action

```
"goal_description":
↪   "Make lettuce cheese sandwich on table",
"goal": [{
    "predicate": "item_on",
    "args": ["bread", "table"],
    "ids": [1, 2]
  }, {
    "predicate": "item_at",
    "args": ["lettuce", "table"],
    "ids": [3, 2]
  }, {
    "predicate": "item_at",
    "args": ["cheese", "table"],
    "ids": [4, 2]
  }, {
    "predicate": "item_at",
    "args": ["bread", "table"],
    "ids": [5, 2]
  }, {
    "predicate": "clear",
    "args": ["bread"],
    "ids": [5]
}]
```

(d) Goal Description

ROBOTOUILLE - Dataset Details

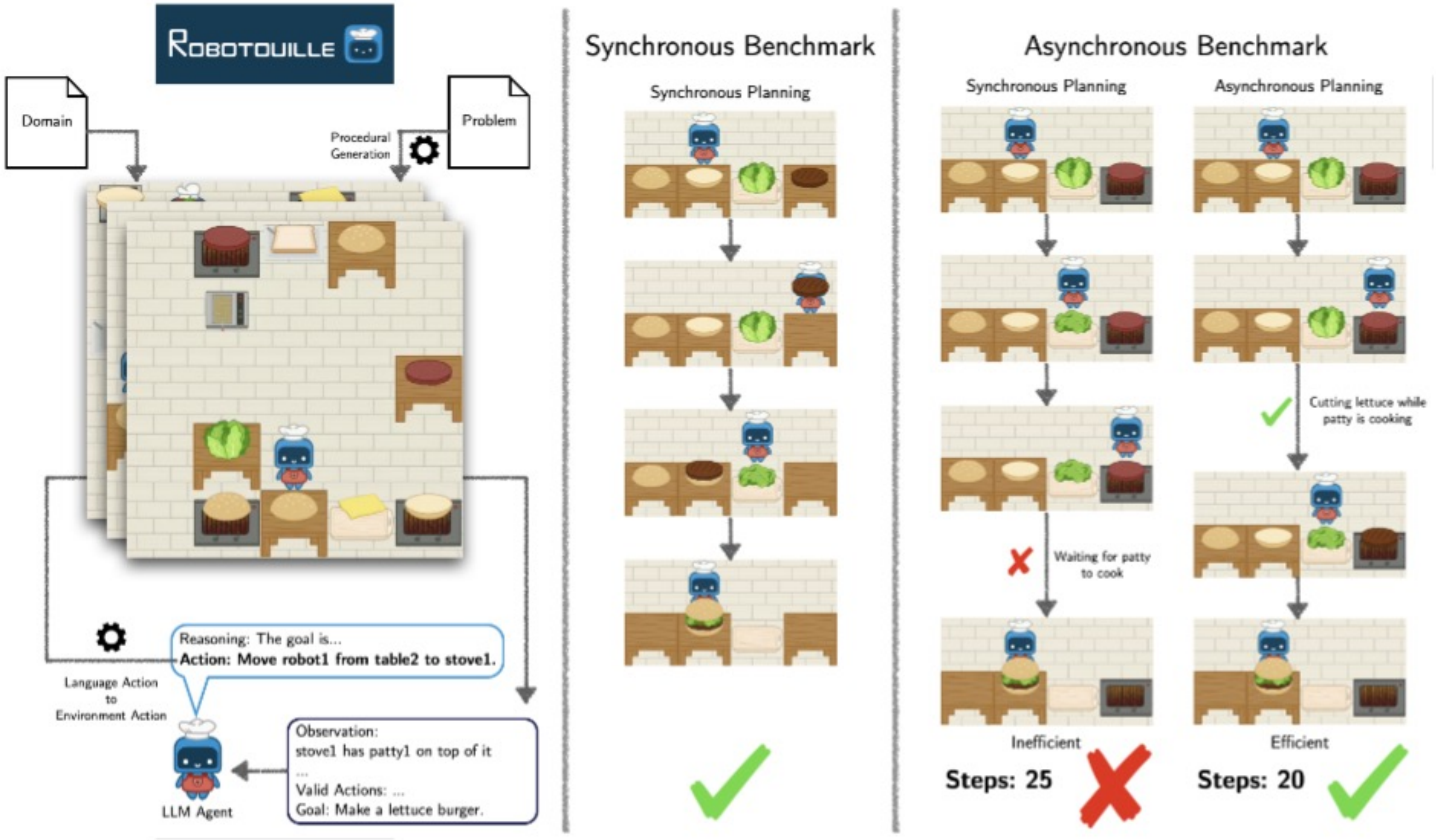
Synchronous Dataset



Asynchronous Dataset

ROBOTOUILLE

ROBOTOUILLE



Experiments & Results

Baselines

- I/O: initial state, valid actions, goal을 input으로 받아 전체 plan을 output
- I/O CoT: initial state을 input으로 받음 + 매 action을 plan하기 전에 chain-of-thought 생성
- ReAct: current state를 기반으로 next action + reasoning output

Overall Results

- Closed-loop agents are superior
- Poor feedback incorporation leads to decreased asynchronous performance
- Synchronous and asynchronous failures are closely related
- Task prioritization is critical in asynchronous planning

	Synchronous (%)			Asynchronous (%)		
	I/O	I/O CoT	ReAct	I/O	I/O CoT	ReAct
gpt4-o	4.00	14.0	**47.0**	1.00	1.00	**11.0**
gpt-4o-mini	4.00	10.0	11.0	0.00	1.00	0.00
gemini-1.5-flash	0.00	13.0	0.00	0.00	0.00	0.00
claude-3-haiku	1.00	2.00	2.00	0.00	0.00	0.00

Experiments & Results

Success and Optimality

Finding 1. Closed-loop baselines outperform open-loop baselines

Success 기준: Reaching the goal within $1.5 \times \{optimal\ \#\ of\ steps\}$

ReAct + gpt 4-o가 synchronous, asynchronous 모두에서 최고 성능

정성평가 결과 long-context로 인한 성능 저하가 나타나는 경우가 많이 나타남
e.g. ReAct + gemini-1.5-flash case의 경우 current environment가 아닌
few-shot example을 solve하는 경우들이 많음

gpt-4o baseline에 대한 task-specific success rate를 보면 ReAct가
대부분의 task에서 highest performance를 기록

Horizon length를 주로 난이도를 판별하는 지표로서 사용하지만 정작
success rate가 그에 완전히 dependent한 것은 아니다

	Synchronous (%)			Asynchronous (%)		
	I/O	I/O CoT	ReAct	I/O	I/O CoT	ReAct
gpt4-o	4.00	14.0	**47.0**	1.00	1.00	**11.0**
gpt-4o-mini	4.00	10.0	11.0	0.00	1.00	0.00
gemini-1.5-flash	0.00	13.0	0.00	0.00	0.00	0.00
claude-3-haiku	1.00	2.00	2.00	0.00	0.00	0.00

	I/O	I/O CoT	ReAct	Horizon Length
Synchronous (%)				
[1]	20.0	40.0	**70.0**	10
[2]	0.00	20.0	**80.0**	14
[3]	10.0	30.0	**80.0**	24
[4]	0.00	10.0	**40.0**	10
[5]	0.00	0.00	**60.0**	15
[6]	10.0	**20.0**	**20.0**	23
[7]	0.00	0.00	**50.0**	36
[8]	0.00	10.0	**30.0**	44
[9]	0.00	10.0	**20.0**	63
[10]	0.00	0.00	**20.0**	57
Total	4.00	14.0	**47.0**	
Asynchronous (%)				
[1]	10.0	0.00	**20.0**	21
[2]	0.00	0.00	**30.0**	27
[3]	0.00	0.00	**40.0**	37
[4]	0.00	0.00	**10.0**	42
[5]	0.00	**10.0**	0.00	46
[6]	0.00	0.00	**10.0**	19
[7]	0.00	0.00	0.00	42
[8]	0.00	0.00	0.00	46
[9]	0.00	0.00	0.00	68
[10]	0.00	0.00	0.00	82
Total	1.00	1.00	**11.0**	

Experiments & Results

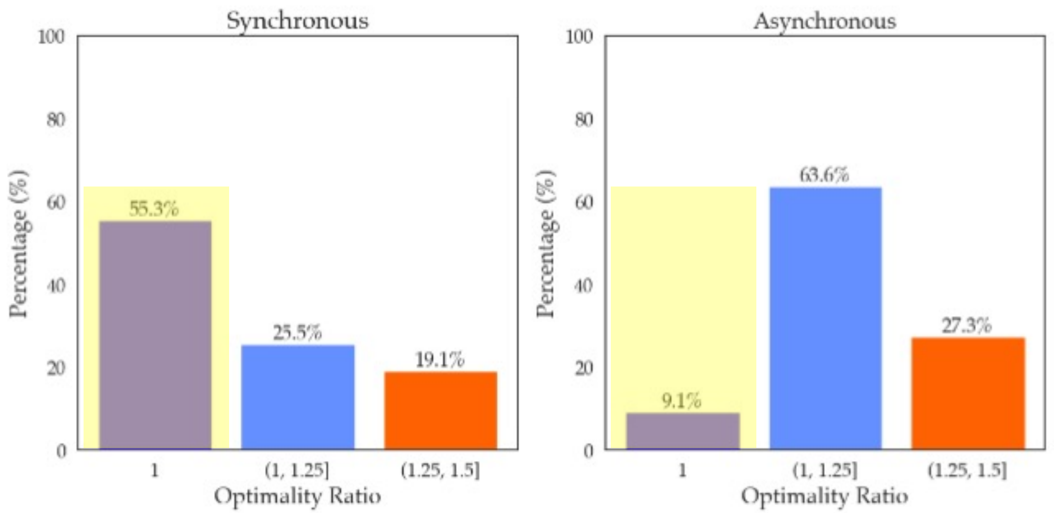
Success and Optimality

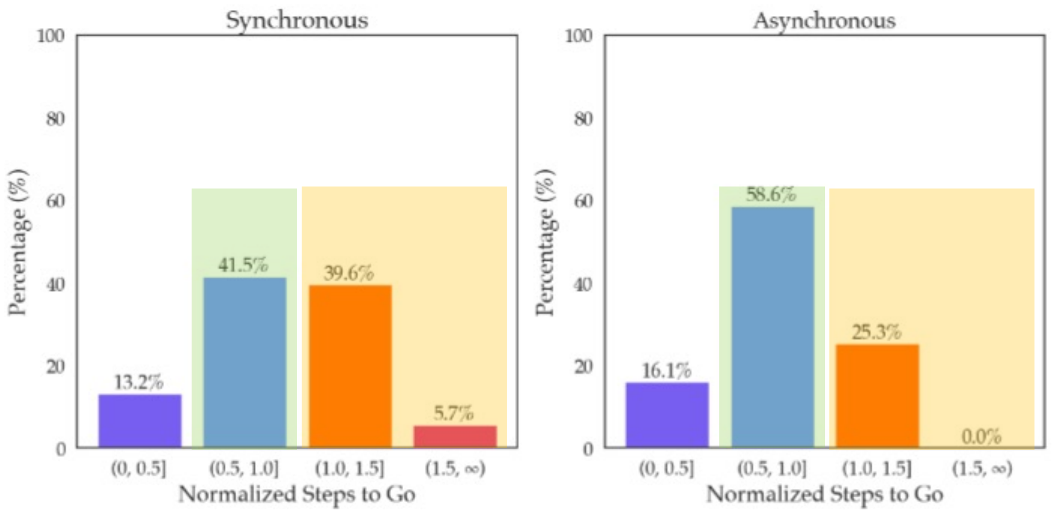
Finding 2. Asynchronous successes are less optimal than synchronous ones

Finding 3. Asynchronous failures make little progress toward the goal

$$Optimality\ Rate = \frac{\| \hat{\tau} \|}{\| \tau^* \|}$$

$$Steps\ to\ Go = \frac{\| \tau^*_{left} \|}{\| \tau^* \|}$$



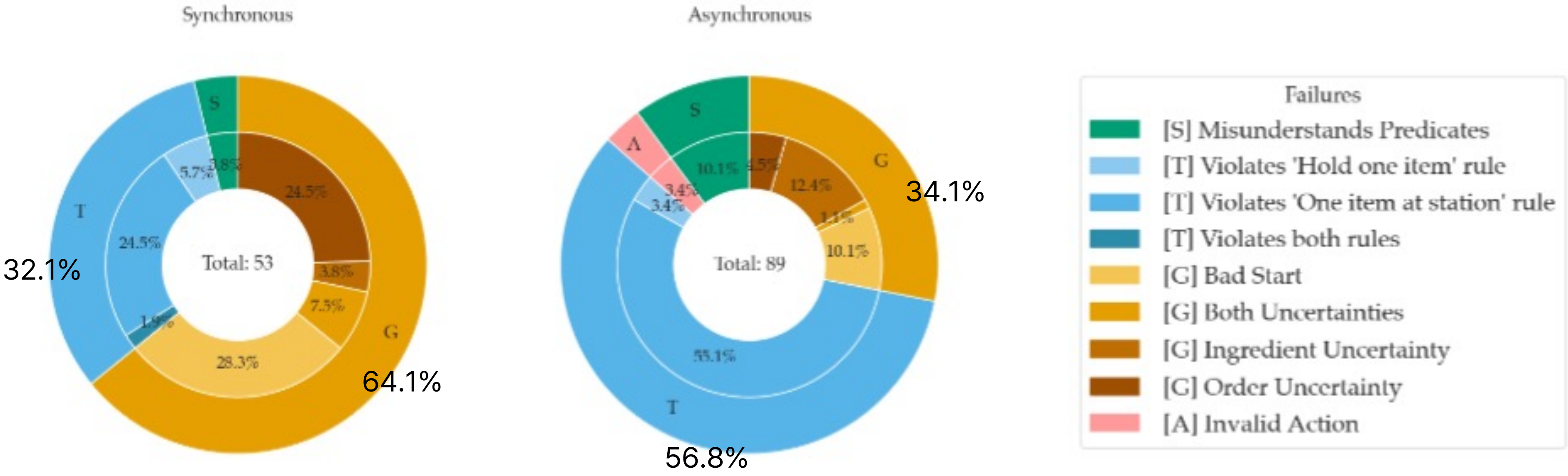


Experiments & Results

Failure Mode Analysis

Finding 4. Dominant failures in both settings stem from rule violations and goal misinterpretations

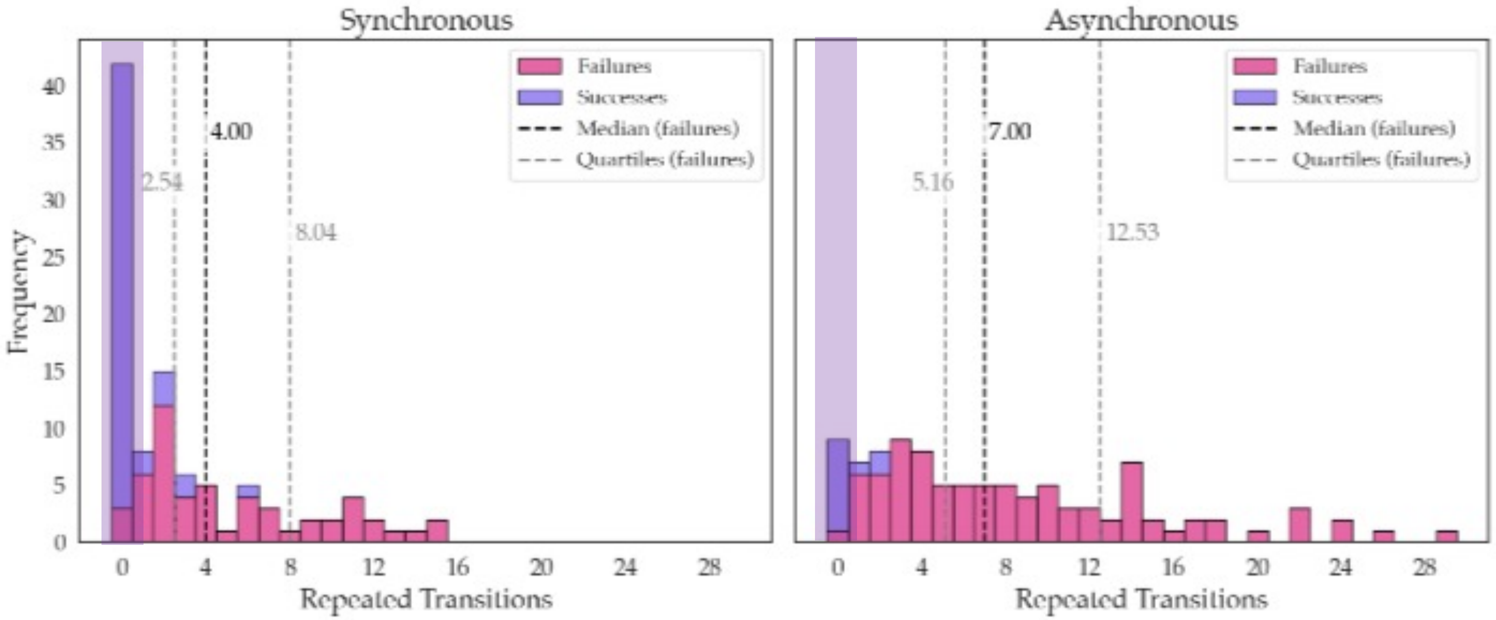
$$M = < S, A, T, R >$$



Experiments & Results

Failure Mode Analysis

Finding 5. Asynchronous recovery is worse than synchronous recovery



While we designed the synchronous and asynchronous datasets to test different capabilities of LLM agents, we mainly observe similar transition failures in both settings

Trial ...

Why Cooking?

Trial ...

Agent Structure

Orchestrator (Planner)
→ 3 * Action Agents (Actor)

timestep 0

- 주문 들어옴

- Orchestrator

1) 주문에 포함된 메뉴 확인, DB에서 해당 메뉴의 recipe 확인
(이때 orchestrator의 planning 능력을 측정해야하는 bench이기에 DB 내 recipe는 자연어 줄글로 구성)

2) Recipe 기반 SERVE까지 전체적인 planning 수행
(이때 multi-agent 상황임에 유의해서 planning 해달라 요구, 기본적으로 3-agent 상황 가정)
(planning은 각 timestep 별로 각 agent의 action을 action_library를 기반으로 지정하는 것)
(이때 orchestrator가 생성한 plan은 코드 내 queue 형태로 저장)

timestep n (n ≥ 1)

- Action Agent

해당 timestep n에 orchestrator에게서 부과된 업무 수행
동선 내 collision은 고려/FAIL 대상 X, 이미 다른 agent에 의해 occupy 된 기구로 navigate 되지만 않으면 됨

- Orchestrator

Orchestrator는 기본적으로 timestep 0 이후로는 동작하지 않음.

timestep 0 이후로 orchestrator가 trigger되는 조건은 크게 세 가지:
1) 새로운 주문, 주문 변경, 주문 취소 등 동적 변경 사항 감지
2) Action Agent들 중 하나라도 FAIL 감지
3) Orchestrator가 작성한 schedule을 모두 수행했는데 아직 SERVE되지 않은 메뉴가 있음

Termination

- 모든 메뉴가 SERVE됨.
- 최대 timestep 수 초과 (매우 크게 둘 것!!)

Trial ...

Scenario

Level 1.

주문 하나만 처리, 주문 하나에 메뉴 하나만 포함.

Level 2.

주문 하나만 처리, 주문 하나에 메뉴 여러개 포함.

Level 3.

주문 여러개 처리, 조리 과정 중 주문 취소, 변경 사항, 주문 추가 등 동적 상황 발생 가능

모든 level scenario에 손님 개인 custom은 가능 (고기 굽기 정도, 특정 재료 제외 등 정량적인 timestep 및 객관적으로 이행 여부가 판별 가능한 것) → 이후 올바르게 수행되었는지 eval 단계에서 확인

감사합니다