Engineering Multi-Agent Systems I

PCS-5045

Escola Politécnica da USP
LTI – Laboratório de Técnicas Inteligentes

Agenda

- EP2 - Discussion
- Task Assignment
- Evaluation of MAS

EP2

<DISCUSSION>

Task Assignment

Follow-up: Task Scheduling

- Centralized Task Assignment
- Decentralized Task Assignment

Task Assignment

Centralized Task Assignment

- One central entity responsible for task assignment.
- Advantages: Global optimization, consistent decisions.
- Disadvantages: Scalability issues, single point of failure.

Task Assignment

Decentralized Task Assignment

- Agents decide on tasks based on local information.
- Advantages: Scalability, adaptability.
- Disadvantages: Can lead to suboptimal solutions, need for coordination mechanisms.

Task Assignment

Factors Influencing Choice

- System objectives (e.g., efficiency vs. resilience).
- Environment dynamism.
- Agent heterogeneity.
- Task characteristics.

Task Assignment

Task Allocation Metrics

- Time to completion.
- Resource utilization.
- Agent idle time.
- Task waiting time.

Task Assignment

FiTA (Fixed Task Assignment)

- Static task assignment mechanism.
- Tasks are pre-assigned to agents based on certain criteria.
- No real-time decision-making required during execution.

Task Assignment

FiTA Phases:

- Pre-analysis
- Assignment
- Execution
- Post-execution

Task Assignment

FiTA – Pre-analysis Phase:

- The system evaluates the capabilities, preferences, and constraints of the agents.
- The tasks are analyzed in terms of requirements, dependencies, priorities, etc.

Task Assignment

FiTA – Assignment Phase:

- Tasks are assigned to agents based on the analysis.
- Assignment is done statically, i.e., before runtime.
- A mapping is created that links tasks to specific agents.
- This can be done using different strategies such as matching capabilities, balancing the load, or meeting specific objectives.

Task Assignment

FiTA – Execution Phase:

- Agents execute their pre-assigned tasks.
- There may be mechanisms to handle exceptions or unexpected situations, but by design, FiTA does not allow for reassignment during execution.

Task Assignment

FiTA – Post-execution Phase:

- Evaluate the performance, log results, etc.
- Any learning or adjustments based on this run will apply to future instances, not the current one.

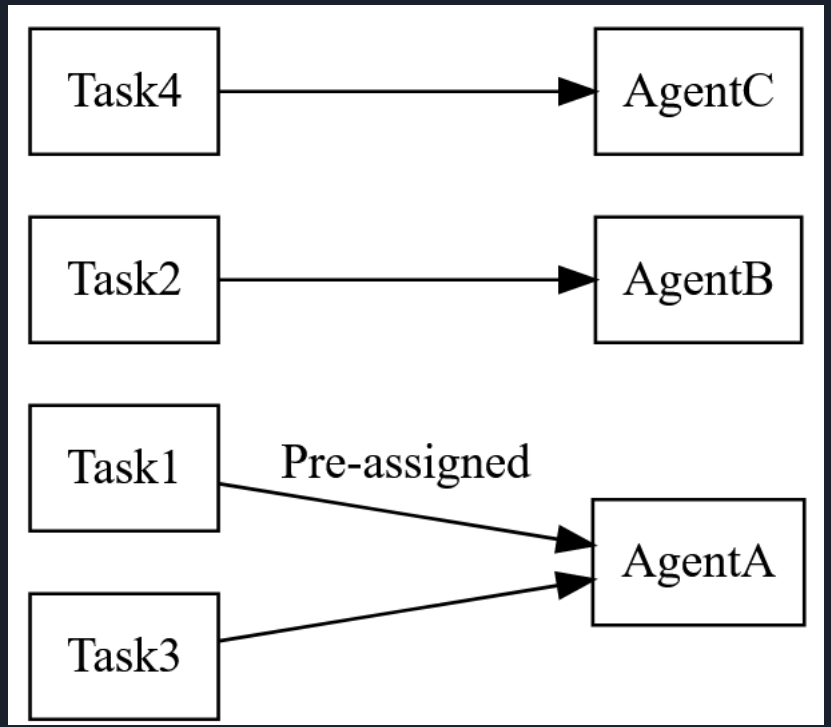
Task Assignment

FiTA – Pros and Cons:

- Pros: Predictability, Predictable, low overhead, can be optimized ahead of time.
- Cons: Inflexible, not adaptive to changing situations or unexpected occurrences (might lead to inefficiencies if the system's state changes).

Task Assignment

FiTA



Task Assignment

FiTA

```
# Tasks are pre-assigned to agents and sent directly without dynamic assignment
class AgentA(Agent):
    tasks = ["Task1", "Task3"]  # Pre-assigned tasks for AgentA

class AgentB(Agent):
    tasks = ["Task2"]

class AgentC(Agent):
    tasks = ["Task4"]
```

Task Assignment

DynCNET (Dynamic Contract Net)

- Dynamic task assignment mechanism based on the Contract Net Protocol.
- Agents bid for tasks based on local information and the coordinator assigns the task to the winning bid.

Task Assignment

DynCNET Phases:

- Task Announcement
- Bidding
- Bid Evaluation
- Task Assignment
- Execution
- Post-Execution

Task Assignment

DynCNET – Task Announcement Phase:

- A task is announced to potential agent contractors by the task manager or coordinator.
- The announcement includes all the information required to evaluate the task, such as requirements, deadlines, rewards, etc.

Task Assignment

DynCNET – Bidding Phase:

- Agents that are capable and willing to perform the task evaluate the announcement.
- Each agent generates a bid based on its evaluation.
- Bids are sent back to the coordinator.

Task Assignment

DynCNET – Bid Evaluation Phase:

- The coordinator evaluates all the received bids.
- Evaluation can be based on various criteria such as cost, time, quality, trustworthiness of the agent, etc.

Task Assignment

DynCNET – Task Assignment Phase:

- The coordinator assigns the task to the agent with the best bid.
- All agents are informed of the decision.
- The winning agent starts working on the task.

Task Assignment

DynCNET – Execution Phase:

- The assigned agent executes the task.
- If necessary, status updates and partial results may be communicated back to the coordinator or other stakeholders.

Task Assignment

DynCNET – Post-execution Phase:

- The results are returned to the coordinator.
- Payment, feedback, or other post-task processes are handled.
- The experience can be used to update the evaluation and bidding strategies for future tasks.

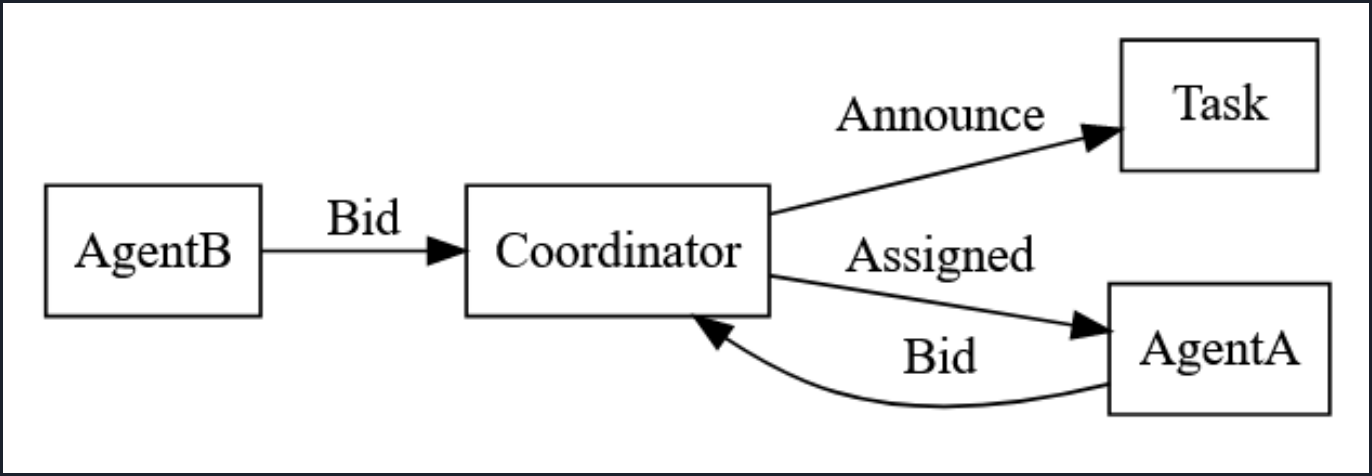
Task Assignment

DynCNET – Pros and Cons:

- Pros: Flexible, adaptive to changing situations, can lead to optimal or near-optimal solutions.
- Cons: Increased communication and computational overhead, complexity in bid evaluation and handling of dynamic situations.

Task Assignment

DynCNET



Task Assignment

DynCNET

```python
from spade.agent import Agent
from spade.message import Message

class CoordinatorAgent(Agent):
    async def announce_task(self, task):
        # Announce task to all agents
        # Collect bids
        # Assign task to agent with best bid

class WorkerAgent(Agent):
    async def bid_for_task(self, task):
        # Evaluate task based on local criteria
        # Send bid to coordinator
        bid_value = self.evaluate_task(task)
        bid_message = Message(to="coordinator@your_domain")
        bid_message.body = str(bid_value)
        await self.send(bid_message)

```

Task Assignment

FiTA vs DynCNET:

- Static vs. Dynamic.
- Predictability vs. Flexibility.
- Coordination overhead.
- Efficacy in different scenarios.

Evaluation of MAS

Motivation

- Why evaluate?
- The challenges of evaluating multiagent systems.

Evaluation of MAS

Evaluation Dimensions

- Performance: How well does the system achieve its objectives?
- Scalability: How does performance change as the number of agents/tasks increases?
- Robustness: How does the system handle failures or unforeseen scenarios?
- Efficiency: Resource consumption vs. output.

Evaluation of MAS

Performance Metrics

- Task completion time.
- Resource usage.
- Agent throughput.
- System responsiveness.

Evaluation of MAS

Performance Evaluation - SPADE

- Integrate performance monitoring tools or libraries with SPADE.
- Log agent activities, task start/end times, and other relevant events.
- Measure response time for key tasks.
- Compute throughput by analyzing logged data over specific intervals.

Evaluation of MAS

Performance Evaluation - SPADE

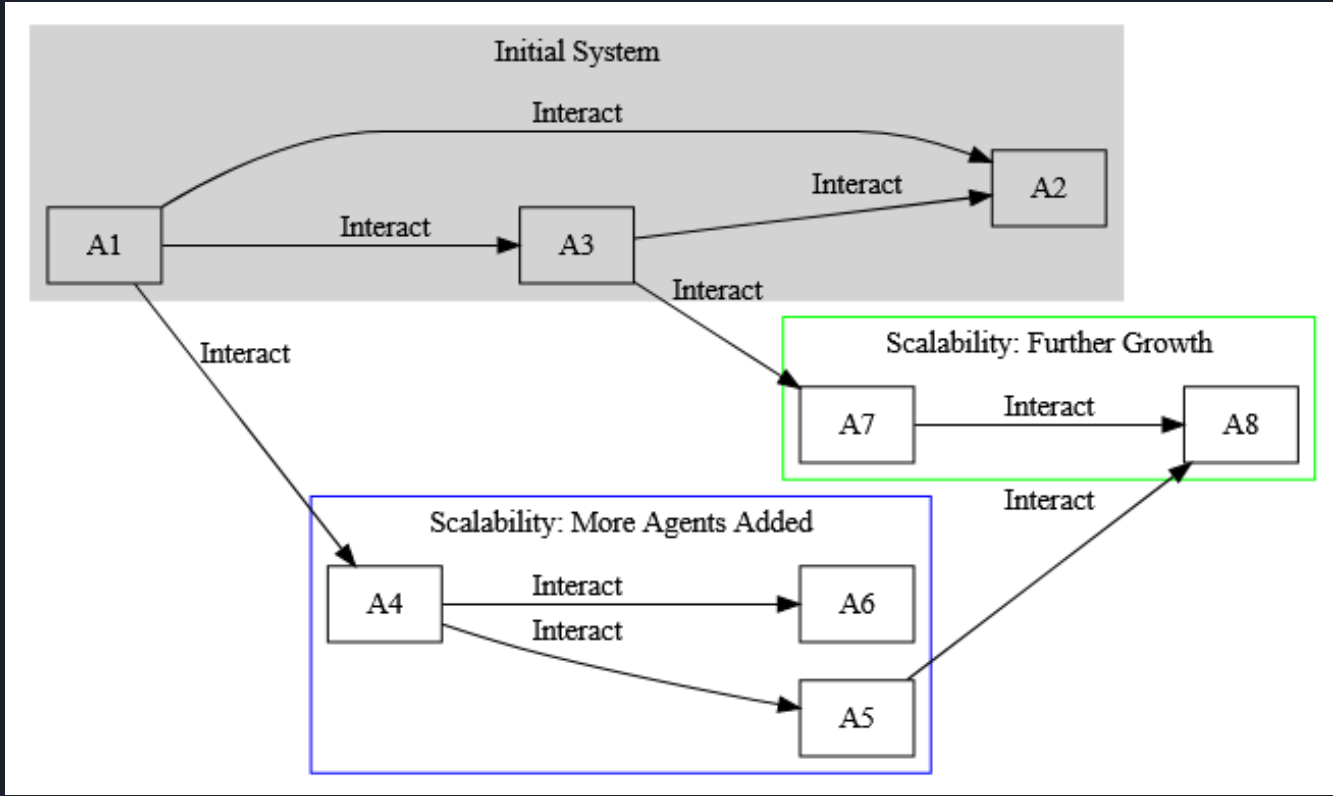
```python
from spade.agent import Agent
import time

class PerformanceAgent(Agent):
    def on_start(self):
        self.start_time = time.time()

    def on_task_complete(self):
        self.end_time = time.time()
        response_time = self.end_time - self.start_time
        # Log response_time for analysis
```

Evaluation of MAS

Scalability Analysis

- How to measure scalability.
- Importance of anticipating growth.

Evaluation of MAS



Evaluation of MAS

Scalability Analysis

- Identify Scalability Metrics: Determine metrics such as load capacity, growth rate, performance under load, etc.
- Perform Load Testing: Incrementally increase the load on the system and observe how it reacts.

Evaluation of MAS

Scalability Analysis

- Analyze Results: Assess how well the system sustains performance as it scales.
- Identify Bottlenecks: Determine any areas that hinder scalability.
- Implement Solutions: Make necessary adjustments to enhance scalability.

Evaluation of MAS

Robustness Evaluation

- Introducing failures or unexpected scenarios.
- Observing system recovery.
- Measuring downtime or degradation.

Evaluation of MAS

Robustness Evaluation

- Identify Potential Failures: Understand the areas where failures can occur.
- Simulate Failures: Use fault injection tools or techniques to simulate errors within SPADE agents.
- Analyze System Response: Evaluate how the system handles failures.
- Implement Mitigations: Add appropriate error handling, redundancy, or other measures to improve robustness.

Evaluation of MAS

Efficiency Metrics

- CPU and memory usage.
- Network bandwidth consumption.
- Cost per task or operation.

Evaluation of MAS

Efficiency Metrics

- Identify Efficiency Metrics: Select relevant metrics like CPU usage, memory consumption, etc.
- Monitor Resource Utilization: Use monitoring tools to track resource usage in SPADE agents.
- Analyze Utilization Patterns: Understand how resources are used over time and under different conditions.
- Optimize Resource Use: Make necessary adjustments to improve efficiency, such as optimizing code or reallocating resources.

Evaluation of MAS

Evaluation Methods

- Simulations: Controlled environments, repeatability.
- Real-world experiments: Ground truth, but can be complex.
- Theoretical analysis: Mathematical models, proofs.

Evaluation of MAS

Challenges in MAS Evaluation

- Non-deterministic behavior.
- Complexity of interactions.
- Dynamic environments.
- Ensuring repeatability.

Evaluation of MAS

Tools and Frameworks for Evaluating MAS

- MAS-specific simulation tools.
- Monitoring and logging tools.

Evaluation of MAS

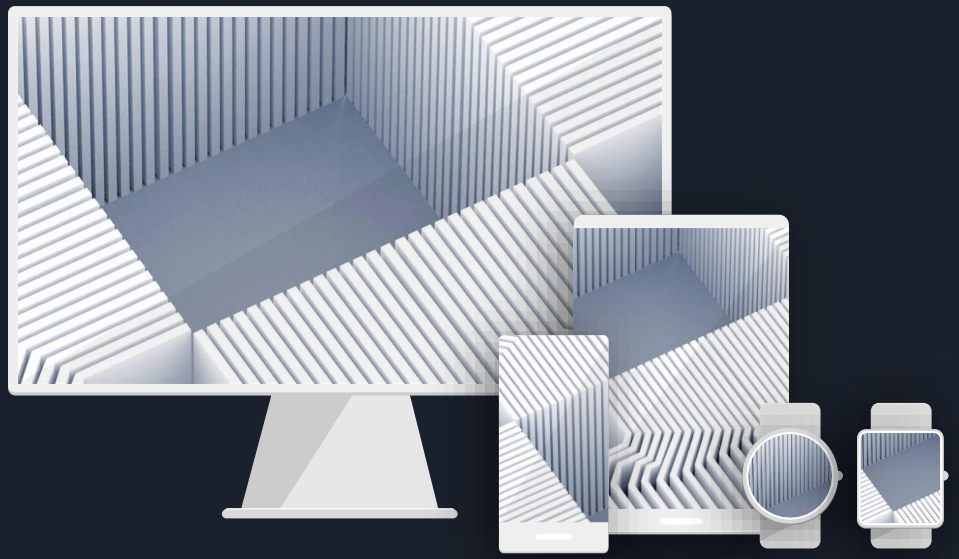
Best Practices

- Set clear objectives for each evaluation.
- Ensure repeatability.
- Consider both quantitative and qualitative metrics.

Thank you!

anarosa.brandao@usp.br

arthur.casals@usp.br



References

1. Michael Wooldridge. An introduction to multiagent systems. Baffins Lane, John Wiley and Sons, 2009 2nd ed.
2. Gerhard Weiss (Ed). Multiagent systems. Cambridge, 2nd edition MIT Press, 2013.