Rule Based Action Plans Generation in A Multi Agent System- An Algorithmic Approach

Debasmita Das¹, Shibakali Gupta²
¹PG Scholar, ²Assistant Professor, Dept of CSE, University Institute of Technology, Burdwan, India

Abstract— Multi agent system has been one of the most challenging parts in the field of artificial intelligence. Multiple agents are applied to do definite works using own intelligence. Communication is a medium through which agents can contact each other and perform at their best. In this paper we have formed multi agent communicative rule based architecture algorithmically. The framework depicts how the agents in an efficient way, can conduct their target sequentially and can provide ultimate results.

Keywords—multi agent, super agent, sub agent, knowledge base, rule base, action agent

I. INTRODUCTION

Multi agent system and its communication are the main concern presented in our paper. The system consists of separate several modules where each module is algorithm dependent and responsible for accessing a part involving the work. We have described these modules in a précised way. Solving the tasks by forming knowledge base and rule base is the key area projected here [1], [2], [3]. We hope, the entire design of the multi agent system will emphasize the agent environment in a broader way. We have arranged the paper as follows: it provides the background, gives the design architecture and its description including algorithms, then provides execution steps, example, table and graph and finally concludes.

II. BACKGROUND

Agent system is considered as a featured area of Artificial Intelligence, which has been used to handle easier to complex problems [4], [5]. An agent is an entity like a computer program. It can adopt any changes made in the environment and known as autonomous [6]. Multi agent system is composed of multiple autonomous entities containing variety of information [7], [8], [9]. The ideas evolved are based on soft computing, commerce area, as well as in logical science [10]. Several agents can exist at the same time called as multiple agents and share common resources communicating with each other [11], [12]. The combination of these numbers of agents is known as multi agent system (MAS). Our key focus is to emphasise the utilization of this system in an innovative way to solve simple to complicated problems by agent intelligence.

III. FRAMEWORK DESIGN AND BRIEF DESCRIPTION

Fig. 1. Multi agent communication architecture

A. Super Agent

Super agent is the head of all sub agents. The entire algorithm runs according to the super agent’s instruction and permission. It coordinates among the sub agents as well as other involving agents and takes the ultimate decisions.
B. Sub Agent

A super agent is connected to a number of sub agents. The sub agents are delivered by a number of tasks. The most efficient sub agent finally conducts the target task.

C. User

User prepares the tasks, and sends to super agent for its result. User is requested to give a feedback after getting the result, whether it is satisfactory or not. This feedback is stored for future purpose.

D. Knowledge Base

Each sub agent creates their own knowledge base as task requirements. Note that the knowledge of sub agents is not sharable. The knowledge base is divided in three modules:

1. The first module is oriented to database access, where data records are stored in a database and collected during task computation.
2. The second module is oriented to previous records, where the sub agent can check whether it has any stored feedback and task experience of previously executed task or not.
3. The third module is connected to Wikipedia book where the necessary information is collected.

Thus the knowledge base is formed and can be viewed according to the following specifications:

1. Task requirements (t<sub>r</sub>): it is the key elements needed to compute the task.
2. Task process method (t<sub>m</sub>): it is the steps through which the tasks to be solved.
3. Task output (t<sub>o</sub>): it is the output of the task processed past.
4. Task failure (t<sub>f</sub>): it is whether the previous task was failed.
5. Reason of failure (t<sub>r</sub>): it is the cause that has failed the past task.
6. Overcome of failure (t<sub>o</sub>): it is the method by which the failure can be overcome.
7. Task success (t<sub>s</sub>): it is whether the previous task succeeded.
8. Task feedback (t<sub>b</sub>): it is whether the user inputs a satisfied feedback after getting the task output. If the task is new, it doesn’t contain any feedback.
9. Task advantage (t<sub>a</sub>): it is the importance of a task.
10. Task disadvantage (t<sub>d</sub>): it is any shortcomings of a task.

E. Work Agent

Each work agent is connected to its respective sub agent and performs the capability and optimization checking.

F. Rule Base Agent

The knowledge base forms rule base dividing into three modules according to the following points:

1. Methods of the task given
2. Avoiding the failure causes of the task
3. Avoiding the negative points of the task

Rule base agent extracts these features from knowledge base and yields rules formation.

G. Action Agent

Action agent retrieves the rules formed by rule base agent and then applies them to the task execution.

H. Capability and Optimization (CO) Agent

The CO agent connects to each sub agent for their efficiency test known as capability and optimization computing. The sub agents hire their respective work agents for the checking. The work agents find the derivative of the given target task with respect to the knowledge base specifications.

\[
\frac{d}{dx}(\text{task}) = t_r + t_m + t_o + t_f + t_r + t_s + t_o + t_d
\]  \hspace{1cm} (1)

Now, each derivative is assigned to a priority number as:

<table>
<thead>
<tr>
<th>Task requirements:</th>
<th>If has: +1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>If no: 0</td>
</tr>
<tr>
<td>Task process method:</td>
<td>If has: +1</td>
</tr>
<tr>
<td></td>
<td>If no: 0</td>
</tr>
<tr>
<td>Task output:</td>
<td>If has: +1</td>
</tr>
<tr>
<td></td>
<td>If no: 0</td>
</tr>
<tr>
<td>Task failure:</td>
<td>If has: -1</td>
</tr>
<tr>
<td></td>
<td>If no: 0</td>
</tr>
<tr>
<td>Reason of failure:</td>
<td>If has: +1</td>
</tr>
<tr>
<td></td>
<td>If no: 0</td>
</tr>
<tr>
<td>Overcome of failure</td>
<td>If has: +1</td>
</tr>
<tr>
<td></td>
<td>If no: 0</td>
</tr>
<tr>
<td>Task success:</td>
<td>If has: +1</td>
</tr>
<tr>
<td></td>
<td>If no: 0</td>
</tr>
<tr>
<td>Task feedback:</td>
<td>If has: +1</td>
</tr>
<tr>
<td></td>
<td>If no: 0</td>
</tr>
</tbody>
</table>

If there is a task feedback, then again the feedback derivatives as:

\[
\frac{d}{dx}(\text{feedback}) = f_s + f_d
\]  \hspace{1cm} (2)
Where $f_s$ = satisfactory and assigned to +2, $f_u$ = unsatisfactory and assigned to -2

Task advantage:  
If has: +1  
If no: 0

Task disadvantage:  
If has: +1  
If no: 0

After the priority assignment, the summation of the prior derivatives will be taken. The sub agents, that produce the maximum value of total priority for their work agents, are capable and optimized. The result is sent to the CO agent and it records it in a log book according to the sub agents’ rank.

If there is more than one capable and optimized sub agents that has the same priority, then the following conditions will be followed:

1. The sub agents will be checked whether they are busy in doing any other task or free.
2. If more than one sub agent is free, then sequence wise the task will be delivered to the first sub agent.
3. If the first sub agent fails, then the next ranked sub agent will be considered.

IV. PROPOSED ALGORITHM

The algorithms drawn from the concept are given as follows:

J. Initiation Algorithm

This algorithm describes the starting of the task procedure between user and super agent.

ALGORITHM (INITIATION)

S-1: U $\leftarrow$ user  
S-2: S $\leftarrow$ super agent  
S-3: user prepares the tasks  
S-4: U sends tasks to S  
S-5: return U

K. Task Analysis Algorithm

This algorithm describes how a task is handled sequence wise.

S-1: Su $\leftarrow$ sub agent  
S-2: S $\leftarrow$ super agent  
S-3: CO $\leftarrow$ capability and optimizing agent  
S-4: task sent to Su by S

L. Knowledge Base Algorithm

This algorithm describes how knowledge is created from the target task.

ALGORITHM (KNOWLEDGE BASE)

S-1: Su $\leftarrow$ sub agent  
S-2: KB $\leftarrow$ knowledge base  
S-3: S $\leftarrow$ super agent  
S-4: Su creates KB  
S-5: KB composed of three modules  
S-6: Kb treated as requirements  
S-7: Su informs S  
S-8: return Su

M. Capability and optimization Algorithm

This algorithm describes the capability and optimization process.

ALGORITHM (CAPABILITY AND OPTIMIZATION)

S-1: Su $\leftarrow$ sub agent  
S-2: WA $\leftarrow$ work agent  
S-3: Su delivers task to WA  
S-4: task derivates with respect to KB specifications  
S-5: priority number assigned to derivatives according to priority rule  
S-6: priority summation collected  
S-7: Su are recorded to log book in ascending order of their summation  
S-8: maximum priority summed Su are considered as most capable and optimized  
S-9: return Su

N. Rule Base Algorithm

This algorithm describes how the rules are formed to task completion.

ALGORITHM (RULE BASE)

S-1: RA $\leftarrow$ rule base agent  
S-2: RB $\leftarrow$ rule base  
S-3: KB $\leftarrow$ knowledge base  
S-4: RA extracts rules from KB  
S-5: RB formed by RA  
S-6: return RA
O. Action Agent Algorithm

This algorithm describes the working procedure of action agent.

ALGORITHM (ACTION AGENT)

S-1: AA ← action agent
S-2: RB ← rule base
S-3: AA retrieves rules from RB
S-4: task executed via test
S-5: return task

V. TASK EXECUTION

The task execution can be described stepwise and sequentially as follows:

1. A user delivers a task or a number of tasks to the super agent.
2. Super agent delivers the task to its respective participating sub agents.
3. The sub agents gather their requirements by creating their own knowledge base.
4. After collecting requirements, the sub agents inform the super agent.
5. The super agent delivers the responsibility of capability and optimization checking to CO to find the most efficient sub agent.
6. The CO agent communicates with the sub agent and the sub agents call their respective work agents for CO checking.
7. The CO agent after computation forms a log table where the sub agents are arranged rank wise.
8. CO agent informs super agent.
9. Super agent delivers the task according to the selected sub agent.
10. The chosen sub agent calls the action agent which extracts and forms rules from rule base agent.
11. The action agent fetching the rules performs the test.
12. Results are delivered to super agent.
13. User, after receiving the output provides a feedback to super agent.

VI. EXAMPLE

Let us consider a simple example:
The task is to produce the pay slip that is gross salary and net salary of the employees of a bank.

Suppose there are four participating sub agents:
1. Sub agent 1
2. Sub agent 2
3. Sub agent 3
4. Sub agent 4

The four sub agents prepare their task requirements as follows:

1. Name of the employee
2. The unique id of each employee
3. The salary structure (basic)
4. The dearness allowance (da)
5. The allowance of house rent (hra)
6. The allowance claimed in medical domain (ma)
7. The insurance (in)
8. The provident fund (pf)
9. The process method to calculate gross and net salary

Knowledge Base Creation

From requirements list, the first eight data are collected from database. Let, for sub agent 2, the database is as:

TABLE I

<table>
<thead>
<tr>
<th>NAME RECORD</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mailini Roy</td>
</tr>
<tr>
<td>Sutapa Biswas</td>
</tr>
<tr>
<td>Maan Singha</td>
</tr>
<tr>
<td>Bedi</td>
</tr>
<tr>
<td>Chandrakanta Ghosh</td>
</tr>
<tr>
<td>Ashish Kumar</td>
</tr>
</tbody>
</table>

TABLE II

<table>
<thead>
<tr>
<th>ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>E1</td>
</tr>
<tr>
<td>E2</td>
</tr>
<tr>
<td>E3</td>
</tr>
<tr>
<td>E4</td>
</tr>
<tr>
<td>E5</td>
</tr>
</tbody>
</table>
The last or ninth data is fetched from Wikipedia book as:

Gross salary = basic + (hra * basic) / 100 + (da * basic) / 100 + (ma * basic) / 100

Net salary = gross salary - (pf + insurance)

Let, sub agent 2 is a fresher for this task. So, it doesn’t have any past working experience or feedback.

The knowledge base is created for sub agent 2. Likewise, each of the rest three sub agents forms their own knowledge base.

**Capability and Optimization Checking**

Suppose, for sub agent 2, according to the equation (1), it gives,

\( \frac{df}{dx(yazhip)} = f_1 + f_2 + f_3 + f_4 + f_5 + f_6 + f_7 + f_8 + f_9 + f_{10} \)

Now, if priority is assigned to each parameter according to the conditions and the summation is taken, it becomes,

\[ = 1+1+0+0+0+0+1+1 \]
\[ = 4 \]

Suppose, for sub agent 1, the priority summation provides,

\[ 1+1+1+ (-1) +1+0+0+ (-2) +1+1 \]
\[ = 3 \]

Now, let the sub agents are recorded ranked wise according to their summation as:

**TABLE IX**

<table>
<thead>
<tr>
<th>Rank</th>
<th>Sub agent</th>
</tr>
</thead>
<tbody>
<tr>
<td>I</td>
<td>2,3</td>
</tr>
<tr>
<td>II</td>
<td>1</td>
</tr>
<tr>
<td>III</td>
<td>4</td>
</tr>
</tbody>
</table>

So, we get sub agent 2 and sub agent 3 as the most capable and optimized and recorded for future reference. Now, we will check which of the two sub agents is free. If we get both of the agents free, then sequence wise the task is first delivered to sub agent 2. If it fails, then sub agent 3 will get a chance.
Rule Base Creation

Suppose, one of the sub agents has the reason of the past task failure as it couldn’t access the data of hra. So, forming the rules, there will be a loop against the checking that whether the sub agent has all the requirements or not. Thus, the rule base created combining method way and related conditions.

Execution

Suppose the action agent executes the task of sub agent 2.

If, the data are as:
- Employee id is E1
- Employee name is “MAILINI ROY”
- Basic is 4000
- hra is 200
- da is 300
- Md is 600
- pf is 900
- Insurance is 400

Then the gross salary= 48,000 and net salary= 46,700.

VII. TABLE

A dependency table can be constructed from the architecture. The dependent modules are action agent, CO agent, rule base agent and work agent. Their dependencies have been shown on the basis of secondary and primary occurrence.

<table>
<thead>
<tr>
<th>Type</th>
<th>Dependent Relation</th>
<th>Secondarily depends on</th>
<th>Primarily depends on</th>
</tr>
</thead>
<tbody>
<tr>
<td>agent</td>
<td>1. Action agent, CO agent, Rule base agent</td>
<td>Sub agent</td>
<td>Super agent</td>
</tr>
<tr>
<td></td>
<td>2. Work agent</td>
<td>Sub agent</td>
<td>CO agent, super agent</td>
</tr>
</tbody>
</table>

VIII. GRAPH

Fig. 2. Dependency Graph

The graph is generated according to the consequences from the dependency table. The modules which are primarily or secondarily dependent on sub agent or super agent as stated in the table are graphically presented.

IX. CONCLUSION

We have shown how, in a compact and algorithmic way, multi agent communication can takes place without any complexities. The table and the graph help to understand the scenario in a comfortable way. Agents using their own intelligence are intended to solve the given tasks. We can use this architecture for other multi agent properties, as cooperation, coordination and negotiation. The same goal can be achieved through multi agent system involving all their key characteristics later by forming a multi agent society. This society will involve a group of multi agent system not only communicating, additionally coordinating, cooperating and negotiating. This society provides a small world consisting of multi functioned agents concurrently performing tasks. Our future work is intended based on these properties.

REFERENCES


