2025 5th International Conference on Artificial Intelligence and Industrial Technology Applications (AIITA)

Research on the Application of Rule Inference Technology in Intelligent Edge-Terminal Scenarios

Yuwei Chen[1,a]
[1]China Academy of Industrial Internet
Beijing, China
[a]e-mail: chenyuwei@china-aii.com

Jiehao Chen[2,b]
[2]China Academy of Industrial Internet
Beijing, China
[b]e-mail: chenjiehao@china-aii.com

Jie Shi[3,c*]
[3]China Academy of Industrial Internet
Beijing, China
*Corresponding author e-mail: [c]shijie@china-aii.com

***Abstract*—Currently, the general mode of Edge-Terminal collaboration is that the terminal executes lightweight tasks, the edge side collects data from terminal devices for centralized computing, and the two ends conduct data interaction. Due to the limited computing power and memory of terminal devices and the complex deployment environment, there are usually problems such as inefficient deployment and upgrade of end-side programs and difficult management. This paper proposes a rule inference application mode for Edge-Terminal collaborative scenarios. The rule inference engine can run simultaneously on both the edge side and the end side, which can separate business logic from application programs, support the collaborative linkage of data at both ends, and improve the deployment efficiency of end-side devices and the autonomous decision-making ability of both ends.**

Keywords-Rule inference; Expert system; Intelligent computing; Rete algorithm

I. INTRODUCTION

The rule inference engine evolved from the expert system and is able to parse the input data in accordance with a set of predefined rules and make corresponding decisions[1]. The rule inference engine can separate business rules from program code, support the expansion and modification of rules, adapt to different application scenarios, and has good scalability and maintainability[2].

Rule inference technology is widely used in fields such as the Internet of Things (IoT) and intelligent decision-making. For example, in the research on the correlation between IoT traffic data and the automation rules of smart home devices [3], Yujiao Sun analyzed the network traffic of the automation rules of smart home devices, combined with the characteristics of the actions of smart home devices, and used feature matching to find the corresponding device actions in the traffic of automation rules[3]. Jiawen Chen conducted research on intelligent decision - making algorithms for the downlink of measurement and control systems based on rule - based rule inference, genetic algorithms, BP neural networks, and DDPG reinforcement learning[4]. It was confirmed that the adaptability and anti-interference ability of measurement and control systems in complex interference environments can be improved by integrating anti-interference intelligent decision - making technology.

For intelligent Edge-Terminal scenarios, the memory and computing power of terminal devices are limited, and there is no development environment. Developers compile and package the developed programs and deploy them to the terminal for operation. The compiled and packaged programs are difficult to modify. Therefore, every time when faced with the scenario where the business logic needs to be changed, it is necessary to repackage and redeploy, resulting in low iteration and upgrade efficiency[5]. Moreover, due to resource limitations, it is difficult to train and deploy large models on the terminal side. How to achieve fast reasoning and data interaction with the edge side has long been a key problem to be solved in Edge-Terminal scenarios collaborative scenarios. This paper, facing the field of intelligent decision-making, proposes an application mode of rule inference technology in intelligent Edge-Terminal collaborative scenarios, which can quickly draw inference conclusions, improve deployment efficiency, and meet the diverse needs of multiple terminals in different task roles. It has good research significance and application prospects in the field of Edge-Terminal collaborative intelligent decision - making.

II. INTRODUCTION TO RULE INFERENCE TECHNOLOGY

A. Composition of the Rule Inference Engine

The rule inference engine is a kind of software system that conducts reasoning and makes decisions based on rules and factual data, aiming to achieve specific objectives. This paper adopts the Drools rule inference engine, it is a rule engine based on forward inference, which is mainly used for rule management and execution, fact parsing as well as workflow control. It supports dynamic rule updates and can adjust rules in real time without restarting the application program. The Drools rule inference engine mainly consists of a working memory, a rule base, and an inference engine[6], as shown in Figure 1.

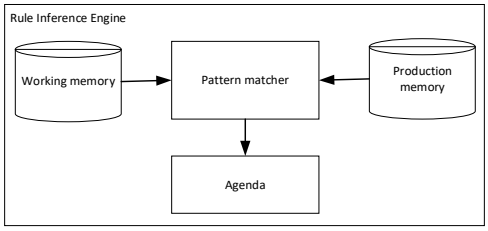

Figure 1. Composition of the Drools rule inference engine

979-8-3315-0976-7/25/$31.00 ©2025 IEEE

Working Memory: It is a collection of facts and is used to store the fact data involved in inference process. The fact data can be input by the users or generated during the inference process.

Rule Base: It is a knowledge set in the form of a collection of rule files, which contains the knowledge required for performing reasoning. The rules in the rule base are composed of conditions and actions, including business rules, decision-making rules, calculation rules, etc. Generally, they are in the form of "IF condition, THEN action", specifying the triggering premise for executing the rule and the specific actions or output results to be executed [7].

Inference Engine: It is the core component of the rule inference engine and is responsible for completing the inference process. It includes components such as a pattern matcher, an agenda, and an execution engine[8]. The pattern matcher is used to compare the rules in the rule base with the facts in the working memory and find out the rules that match the facts. The agenda is used to store the activated rules and arrange them in the order of rules such as priority. The execution engine is responsible for executing the rules in the agenda and updating the facts in the working memory according to the results of the rules. Moreover, the results after execution can also automatically trigger a new round of the inference process.

B. Principle of the Rete Algorithm

The Drools rule inference engine has the Rete algorithm built-in. This algorithm is a forward rule fast matching algorithm, and its matching speed is independent of the number of rules. The core principle of this algorithm is to reduce the amount of pattern matching calculations between a large number of rules and objects by constructing an efficient matching network [9]. The schematic diagram of the Rete algorithm network matching is shown in Figure 2.

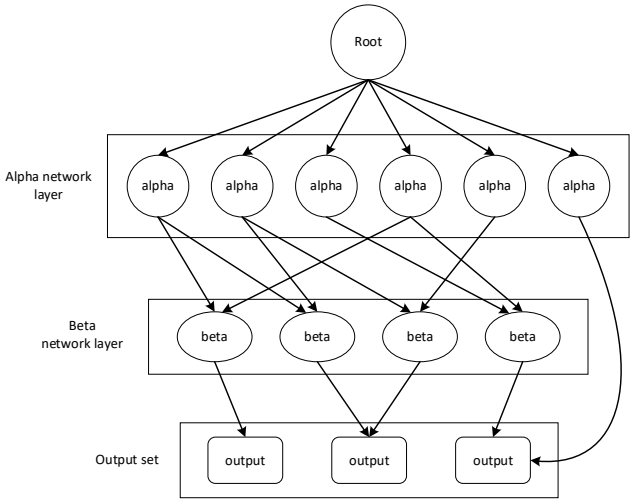

Figure 2. Rete algorithm matching network

The Rete matching network consists of a root node, an Alpha network, a Beta network, and an output set[10]. The root node is a virtual root node, into which the factual data is input. The Alpha network is a pattern matching network composed of rules and is used to record the rule matching conditions. Each alpha node in the Alpha network only allows one input. If there are different patterns in different rules, one node can be shared to reduce redundant calculations. The Beta network is a connection network, which consists of beta nodes. Each beta node can have no more than two inputs and is responsible for checking the constraint relationships among different patterns of rules to avoid matching conflicts caused by the same variables. The output node, also known as the output node, indicates that the corresponding rule has been activated after matching. The Rete algorithm improves the matching efficiency through the following characteristics:

(1) State preservation. Whenever there is a change in the fact base, the states after rule matching are preserved in the alpha and beta nodes. When the fact base changes next time, most of the results do not need to be recalculated, thus avoiding a large amount of redundant calculations.

(2) Node sharing. Different rules containing the same patterns can share the same node, which can further reduce the amount of calculation and thereby improve the matching efficiency[11].

(3) Network construction. The Rete algorithm forms a complex network structure through the connection of various nodes to conduct rapid matching of facts and rules.

C. Execution Process of the Rule Inference Engine

The execution process of the rule inference engine[12] is illustrated in Figure 3:

(1) Input the facts into the working memory.

(2) Use the pattern matcher to compare the rules in the rule base with the facts in the working memory to determine the rules that can match the facts, that is, activate the rules.

(3) If there are conflicts during the execution of rules, that is, multiple rules simultaneously meet the matching conditions, then all the conflicting rules will be put into the conflict set.

(4) Put all the activated rules into the agenda in sequence to resolve conflicts.

(5) Use the execution engine to execute the rules in the agenda. Repeat steps (2) to (5) until all the rules in the agenda have been executed.

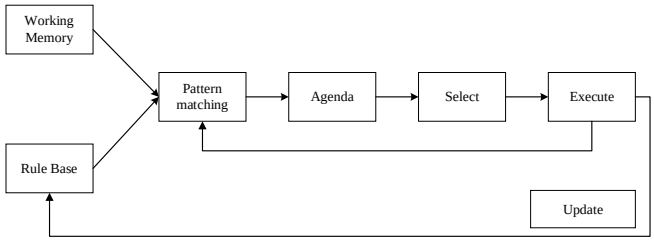

Figure 3. Execution process of the rule inference engine

III. RULE INFERENCE METHOD IN INTELLIGENT EDGE-TERMINAL SCENARIOS

A. Rule Representation Method

Rules mainly consist of conditions and consequences, and the name and attribute parts can also be added to further

maintain and manage the rules. In the rules of Drools, the rule name must exist and be unique; the rule attributes are optional items, which can be used to select the execution mode of the rules, such as no-loop (set not to loop), salience (set the priority of the rule), etc. The condition part of the rule, also known as LHS, can contain multiple conditions, and when there is no condition, it is defaulted to true; the result part of the rule, also known as RHS, represents the action part after the conditions are met[10]. The condition is preceded by the keyword "when", and the result is preceded by the keyword "then". The condition must be before the result. The rule description starts with "rule" and ends with "end". The writing form of a rule is as follows:

```
rule "name"
    when
        LHS
    then
        RHS
end
```

B. Intelligent Edge-Terminal Rule Inference Architecture

This paper proposes an Edge-Terminal architecture based on the rule inference engine, as illustrated in Figure 4. The rule inference engine can run simultaneously on both the edge side and the end side to realize the decoupling of business rules from program code. The edge side supports the formulation and management of rules, can send rule files to the specified intelligent terminal, and interacts with the end side for instructions and inference results. The operation steps of the intelligent Edge-Terminal collaborative architecture are as follows:

(1) Customize rules on the edge side and generate a rule base file in the specified location.

(2) Send the specified rule base file to the specified intelligent terminal through the rule release module.

(3) The edge-side rule inference engine matches the incoming fact data with the rules in the rule base file to obtain inference conclusions or specific instructions and transmits them to the intelligent terminal.

(4) The intelligent terminal rule inference engine receives the instructions transmitted in step (3) or the fact data of the local program of the intelligent terminal, matches and executes the rules based on the rule base file, obtains the inference result, and can transmit the necessary result data back to the edge side.

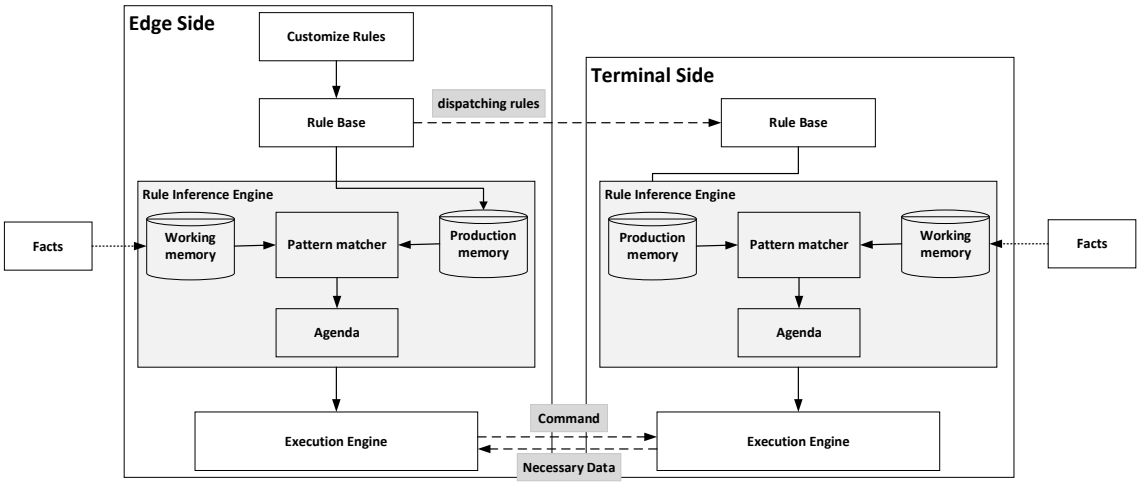

Figure 4. Intelligent Edge-Terminal rule inference architecture

Compared with the traditional software operation, the architecture mode proposed in this paper has the following advantages:

(1) The Drools rule inference engine has the characteristics of cross-platform and hot deployment and can be deployed on both the edge side and the end side. It does not need to be repeatedly compiled and deployed, which can significantly improve the flexibility of both ends.

(2) The Rete algorithm built into the Drools rule inference engine has the characteristics of state preservation and node sharing, and the matching speed is independent of the number of rules. Therefore, in scenarios with complex business and many branches, the software matching and running efficiency can be greatly improved.

(3) Both ends can perform autonomous inference and decision-making. Different end-side devices can locally save different rule files to meet the customization needs of various terminal scenarios. Moreover, it is not necessary to transmit all results back to the edge side for inference and decision-making, reducing communication time and improving the autonomous decision-making and flexibility of both ends.

(4) The edge side can uniformly customize rules and perform rule distribution and data transmission, which can realize the unified management of the end side by the edge side and improve the collaborative linkage of both ends.

C. Execution and Invocation Process of the Drools Rule Inference Engine

The core function of the rule inference engine is rule execution. Users or external systems can provide data through the API and obtain the results obtained by the Drools rule engine by matching the corresponding rules[13]. The execute method of the ApiConfigController is the core method of this function. The specific invocation process of rule execution is shown in Figure 5.

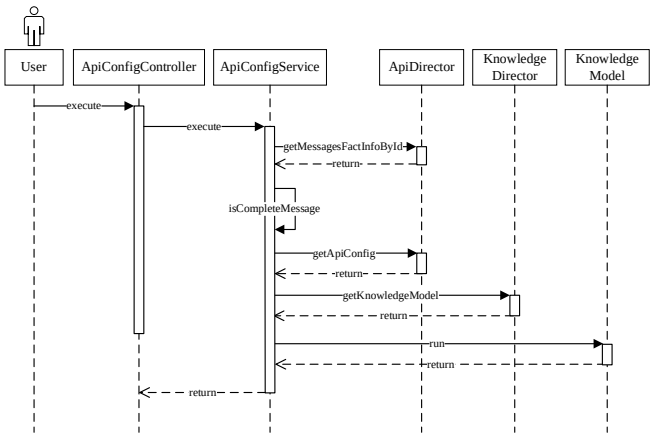

Figure 5. Execution and invocation process of the Drools rule inference engine

(1) The user initiates a request to execute the rules under a certain interface and calls the execute method of the control layer.

(2) The execute method of the control layer calls the execute method of the business layer.

(3) The execute method first calls the getMessageFactInfoById method of the ApiDirector to obtain the rule factor information configured in the interface to be executed, and then calls the isCompleteMessage method to verify whether the required items of the incoming rule factors are complete.

(4) Call the getApiConfg method of the ApiDirector to obtain the corresponding interface information according to the incoming parameters.

(5) Call the getKnowledgeModel method of the KnowledgeDirector with the interface information obtained in (4) to acquire the specific model information of the KnowledgeModel of the published knowledge package configured in maintenance interface.

(6) Call the run method in the corresponding KnowledgeModel, pass in the data information in the message, and execute the rules.

(7) Return the parameters level by level according to the calling sequence, and display and return the result information to the front-end page.

IV. CONCLUSIONS

Based on the characteristics of the Drools rule inference engine and the advantages of the Rete algorithm, this paper proposes a rule inference architecture in intelligent Edge-Terminal application scenarios. It can realize the unified customization and distribution of rules on the edge side and the unified management of intelligent terminals. In addition, the rule inference engine can be deployed on both the edge side and the end side, can independently perform rule inference at the same time, and supports the transmission of data such as instructions and results. This intelligent Edge-Terminal collaborative rule-based inference architecture model has characteristics such as high efficiency, flexibility, and easy deployment. Compared with the traditional Edge-Terminal collaboration mode, it increases the autonomous decision-making and collaborative linkage of both ends and can adapt to more intelligent decision-making application scenarios.

REFERENCES

[1] Y. C. Chen, Design and Implementation of a Business Rule Management Platform Based on the Drools Rule Engine [D]. Xidian University, 2021. DOI:10.27389/d.cnki.gxadu.2021.002535.

[2] K. Zhao, Research on Data Pre-deployment Based on Rule Inference Engine in Intelligent Environment [D]. Chongqing University, 2021. DOI:10.27670/d.cnki.gcqdu.2021.000366.

[3] Y.J.Sun, Research on Automatic Rule Reasoning Technology for Internet of Things Smart Homes [D]. Xidian University, 2023. DOI: 10.27389/d.cnki.gxadu.2023.003567.

[4] J.W.Chen, Research and Implementation of Intelligent Decision - making Technology for the Downlink of Measurement and Control Systems [D]. University of Electronic Science and Technology of China, 2023. DOI: 10.27005/d.cnki.gdzku.2023.004425.

[5] H.Q.Tong. Research on the Optimization of Edge-side Inference Models Based on Deep Learning [D]. Shenyang Aerospace University, 2023. DOI:10.27324/d.cnki.gshkc.2023.000260.

[6] Y.Fu, Research on Distributed Rule Engine Technology for Large-scale Internet of Things Scenarios [D]. Guangzhou University, 2022. DOI:10.27040/d.cnki.ggzdu.2022.001114.

[7] P. Z. Ren and Z. Wang. Research on Rule Design of Human-Machine Command and Countermeasure System [J]. *Command Control & Simulation*, 2023, 45(2):130-136.

[8] W. X. Chen. Research on Active Perception and Processing Technology of Key Events in Manufacturing Internet of Things for Production Process [D]. Guizhou University, 2016.

[9] Y. Liu, Research on Programming Automation Based on Expert System and Efficient Algorithms [D]. Jiangxi Normal University, 2023. DOI:10.27178/d.cnki.gjxsu.2023.000783.

[10] X. Zhou, Design and Implementation of a Clinical Decision Support System Based on the Drools Rule Engine [D]. East China Normal University, 2021. DOI:10.27149/d.cnki.ghdsu.2021.002365.

[11] Y. Yang, X. D. Shi, S. Song, Y. H. Huo, L. D. Chen. Alert Correlation Analysis Based on Rete Rule Inference [J]. Journal of Beijing University of Posts and Telecommunications, 2020, 43(2): 23-28. DOI:10.13190/j.jbupt.2019-113.

[12] Wang and L. Huang, 2024. Knowledge-based Research on Automation Engines. In: 2024 IEEE 10th International Conference on Edge Computing and Scalable Cloud (EdgeCom). Shanghai. pp.37-41.

[13] Z. H. Yang, G. Li, L. Jiao and J. Zhang, 2019. Design and Implementation of Rule Inference Engine in ESTSWEU. In: Journal of Physics. Shanghai. doi:10.1088/1742-6596/1176/2/022047.