STRIPS Planning with Modular Behavior Selection Networks for Smart Home Agents

Kyon-Mo Yang
Dept. of Computer Science
Yonsei University
Seoul, Korea
kmyang@sclab.yonsei.ac.kr

Sung-Bea Cho
Dept. of Computer Science
Yonsei University
Seoul, Korea
sbcho@yonsei.ac.kr

Abstract—A smart home has highly advanced automatic systems for lighting, temperature control, multi-media and many other functions. In the field of intelligent service agents in smart home, the service agent should collect the information using sensors at home such as cameras, temperature sensors and light sensors, and generate agent behaviors appropriate to the user’s request. This paper presents a smart home system with the intelligent agent using modular behavior networks and STRIPS planning system. Behavior selection networks, one of the well-known behavior based methods suitable for goal oriented tasks, are designed in two types of modules: The networks with specific goal for the user’s request, and the networks with common goal to achieve a sub-goal in the specific networks. This modular approach can facilitate reuse and reduce the computation of activation levels without changing the structure. In order to solve complex problems in situations that requires sequential inference, we propose a hybrid system, a goal-oriented STRIPS planning system in behavior selection networks, to achieve global goals efficiently in a smart home. The proposed method is applied to the smart home implemented using Unity3D and verified the usefulness by various scenarios. Experimental results confirm the reduction in computation of activity on action nodes.

Keywords—Smart home, Intelligent agent, Behavior selection network, STRIPS, Planning

I. INTRODUCTION

The advances in the smart home have implied that service agent support people in their daily activities. In the smart home, the intelligent agents monitor current situation using sensors and control the device through actions. Useful and suitable action representations, with accompanying planning techniques are crucial for the task performance of many agent systems, and thus a core issue of research on intelligent agents. However, the planning techniques have their own weakness of low flexibility in complex environments. Reactive techniques can generate behaviors quickly based on environmental conditions [1]. Even though the techniques work well in complex and various domains, it is difficult to generate behaviors robustly when consistency or stability is insufficient. These limitations of both approaches have required a hybrid behavior generation system, a combination of the deliberative techniques and the reactive techniques.

In this paper, we propose a hybrid technique based on behavior selection networks [2] and STRIPS planning for smart home. It is known for its robust performance even though there is insufficient environmental information. The behavior selection networks can generate behaviors automatically with achieving goals, but it is hard to perform perfectly in complex problems that some plans are required to solve since it is designed to run reactively. The proposed method has the contribution that it overcomes this limitation by combining the modular behavior selection networks with STRIPS planning that generates the sequence by considering current situation and global-goals. In addition, we implement the smart home environment using Unity3D and verify the usability of this method.

The rest of the paper is organized as follows. Section 2 presents related works for planning systems and intelligent agent. Section 3 describes in details the hybrid behavior network module. Section 4 reports the experiments conducted to show the usefulness of the method.

II. RELATED WORK

Researchers of intelligent service agents studied how to generate behaviors in smart environments for providing services. They proposed behavior generation structures which generate human-like behaviors in service environments.

A. Intelligent agent

In recent years, the role of intelligent agent has been proliferated in the intelligent services such as web, home, education, and so on. These agents have been used to make users use the function more conveniently by providing the service. Yoon et al. proposed hierarchical intention-response model for users by a home service agent [3]. Tong et al. proposed a three-layer agent-based web service workflow model [4]. This agent’s purpose is that users only need to focus on what they want to achieve rather than how to achieve. Giraffa et al. used the intelligent agents in tutoring system [5]. Tutoring agents are entities whose ultimate purpose is to communicate with the student in order to efficiently fulfill their respective tutoring function, as part of the pedagogical mission of the system. Various fields of research have used the agents and researchers have focused on how to generate the behaviors of agents for responding to the user’s request, which is important problem.

B. Classical planning systems

The classical approach that most planners use today describes states and operators in a restricted language known as the STRIPS language [6]. This language consists of states,
action and goals. The states are represented by conjunction of function-free ground literals, that is, predicates applied to constant symbols, possibly negated. The language selects an action considering states to achieve the goals. STRIPS operators consist of three components for actions. The action description is what an agent actually returns to the environment in order to do tasks. The precondition is a conjunction of atoms that say what must be true before the operator can be applied. The effect of an operator is a conjunction of literals that describe how the situation changes when the operator is applied. Although this classical planning with STRIPS language is a robust technique to achieve a goal, it is difficult to apply to the problem in the reactive situation.

C. Behavior selection networks

The behavior selection network (BSN) proposed by Maes generates behaviors autonomously by attaching goals to the behavior based system [2]. The network was inspired by animal behaviors observed by robotics researchers and ecologists. The behavior network generates the most suitable behavior for an environment using sensory information and established goals. The basic components of the BSN are the action nodes from which the BSN has to choose. Each action node is provided a structure précising its relation with other components. Tyrrell proposed modular BSN to avoid a deadlock between actions [7]. The modular BSN fell into a deadlock, because the network is designed with various goals and an action is conducted for various goals. Tyrrell designed the network for only one goal. Decugis proposed the environment node in BSN [8]. When the behavior is generated using BSN, the system cannot know that which behavior of action node is conducted in real word. To solve this problem, environment nodes should check if the behavior is conducted or not in real word. Dorer proposed the method using resource information [9]. The MBSN does not select multi actions at a time, because the decision method in MBSN is comparison of the activation level among candidates of action nodes. If the action nodes have same activation level, the MBSN selects an action node randomly. The Dorere' method selected the action more flexibly through the conduction of the candidates that do not use same resource. Although BSN is improved by using these methods, it is not easy to apply to complex problems.

D. Hybrid system

Classical planning systems have limitation that they belong to given environments and problems and require a lot of preprocessing steps. Moreover, reactive systems like BSN are not easy to apply to complex problems. A hybrid system has been applied to generate flexible behaviors [10].

The hybrid systems have applied to the various domains to improve the performance. Table 1 shows the previous studies on hybrid system. Min et al. proposed the goal oriented BSN system to generate behavior of delivery service robot [11]. The system used basic behavior sequences plan and BSN. The sequences can be changed by user’s input and the robot conducts behavior over the sequences. Mendoca et al. developed an autonomous navigation system using fuzzy cognitive maps [12]. They used the model to represent the robot’s dynamic behavior in both reactive and deliberative layers. Larue et al. proposed a hybrid system to mimic a human mind [13]. This architecture consists of three layers: reactive, algorithmic, and reflective. The reactive layer is assigned to agents within sensor, effector and knowledge. Algorithmic and reflective layers modify the reactive organization by short-term goal and long-term goal, respectively. By using this method, it can get flexible cognitive results. Vila applied the hybrid system to the network management problem. This system had two different sets of agents [14]. M-agents aim to monitoring the main tasks as the reactive agent. P-agent decided the best way to achieve a maximum network performance as the deliberative agent. Schroeder et al. proposed a hybrid diagnosis agent system based on logic programming [15]. The system contains a deliberative layer of knowledge base and inference machine of a state-of-the-art diagnoser, as well as a reactive layer for communication and control.

<table>
<thead>
<tr>
<th>Authors</th>
<th>Methods</th>
<th>Domain</th>
</tr>
</thead>
<tbody>
<tr>
<td>Min et al. [11]</td>
<td>BSN and predefined sequences</td>
<td>Mobile service robot</td>
</tr>
<tr>
<td>Mendonace et al. [12]</td>
<td>Fuzzy cognitive map based reinforcement learning</td>
<td>Mobile service robot</td>
</tr>
<tr>
<td>Larue et al. [13]</td>
<td>Conceptual map and goal oriented planning</td>
<td>Cognitive human mind</td>
</tr>
<tr>
<td>Vila et al. [14]</td>
<td>Rule-based method and best way planning</td>
<td>Network management</td>
</tr>
<tr>
<td>Schroeder et al. [15]</td>
<td>Rule-based method and knowledge base</td>
<td>Diagnosis agent system</td>
</tr>
</tbody>
</table>

Table 1 shows the hybrid systems which used low-level controls for reactive actions managed by high-level controls. The proposed system is based on modular behavior selection networks as reactive approaches and the modules controlled by STRIPS planning language with considering the global goals in order to overcome limitations of the conventional reactive methods.

III. HYBRID ARCHITECTURE

In this paper, the smart home service agent is designed to respond to the user’s request. The system classifies the type of the user’s requests as single service and complex service. The single service requests are defined as one type of device control in smart home environments. The complex service requests are defined as set of single services. To generate behaviors of agent for providing the service, the system uses STRIPS language module and BSN module.
A. System architecture

Figure 1 shows the system architecture. The system consists of three modules: Request analysis module, STRIPS planning module, and behavior selection network module.

The request analysis module infers type of the user’s request. Yoon et al. defined that the user intention classifies two types [3]. The low-level intention can directly aware the purpose and the high-level intention needed to infer the user’s purpose. We define the requirement of smart home service. The low-level requirement called a single service requirement is that the user wants to control one type of device by the command of the direct meaning. The high-level requirements called a complex service requirement involve the control of the various devices by the command. Depending on the type of user’s requirements, the system uses different procedures.

The STRIPS planning module aims to make sequences. To respond to the complex service requirements, the system needs to generate the sequences. These sequences consist of the set of the single services that map to the goal of BSN. The system controls the execution sequence of the single services using STRIPS language. The sequence queue pushes the sequence and executes the job until the job is accomplished.

The Behavior selection network module selects a behavior to provide services. The module consists of the set of service BSN and common BSN. The service BSNs are designed for specific services. The goal of these BSNs is to control a device. The common BSNs are not used for specific services but for conducting sub-goal in services modules.

B. Modular behavior selection network design

- Definition 1. Maes’ BSN \( B_M = \{E, A, G\} \)

where \( E = \{e_1, e_2, ..., e_n\} \) is the set of environments, \( A = \{a_1, a_2, ..., a_n\} \) is the set of action nodes, and \( G = \{g_1, g_2, ..., g_n\} \) is the set of goals. Equation 1 is the amount of a computation of activation energy [2]. The computation is defined as the number of comparisons between nodes.

\[
AC = \#N \cdot (\#E + \#G + \#P + \#S + \#C) \tag{1}
\]

where \#N, \#E, \#G, \#P, \#S, and \#C are the numbers of nodes, environments, goals, predecessor links, successor links, and conflict links, respectively.

- Definition 2. Tyrrell’ modular BSNs \( B_T = \{E, A, g\} \)

Tyrrell modified the MBSN to avoid the deadlock of action node when generating the behaviors. The BSN was designed to conduct the only one goal. Here is the computation of activation energy in Tyrrell’ BSN. In Equation 2, we assume that \#N, \#E, \#P, \#S, and \#C are proportional to \#G. In Tyrrell’ method, the MBSN is modularized for having one goal.

\[
AC’ = \left( \frac{\#N \cdot (\#E + \#G + \#S + \#C)}{\#G} \right) \cdot \#G
\]

\[
= \frac{\#N \cdot (\#E + \#P + \#S + \#C)}{\#G} \quad \tag{2}
\]

We propose the extended modular design method using two types: Service BSN and common BSN.

- Definition 3. Extended modular BSN \( B_E = \{B_s, B_c\} \)

where \( B_s \) is a set of specific BSNs and \( B_c \) is a set of common BSNs.

- Definition 4. Service BSNs \( B_s = \{E_s, A_s, A_c, g_s\} \)

The service BSNs aim to providing a specific service. These BSNs have two types of the action nodes. \( A_s \) defines actions to conduct the service and \( A_c \) presents the common actions that are performed on the various BSNs.
• Definition 5. Common BSNs $B_c = \{E_c, A_c, g_c\}$

The common BSNs are like sub-BSN in the service BSNs. The BSN is conducted, when select the common action in service BSNs. In each service BSNs $B_s$, the intersection of the environments $E_s$ is represented as $E_s = E_{s1} \cap E_{s2} \ldots \cap E_{sn} = \{e_{s1}, e_{s2}, \ldots, e_{sn}\}$. Similarly, the intersection of the action nodes $A_s$ is represented as $A_s = A_{s1} \cap A_{s2} \ldots \cap A_{sn} = \{a_{s1}, a_{s2}, \ldots, a_{sn}\}$. The set of $g_c$ maps to the set of $A_c$.

$$AC'' = \left(\frac{(N-NC+\#GC) \cdot (\#E+\#P+\#S+\#C) \cdot (\#EC+\#PC+\#SC+\#CC) \cdot \#GC}{\#G}\right)$$

where $\#NC$, $\#EC$, $\#GC$, $\#PC$, $\#SC$, and $\#CC$ are the numbers of nodes, environments, goals, successor, predecessor, and conflict linker in common BSN, respectively.

It is possible to reduce the complexity of computation of activation level and reuse the BSN modules through the proposed method.

Table 2. Condition list of action nodes in TV management BSN

<table>
<thead>
<tr>
<th>Action node</th>
<th>Precondition</th>
<th>Add</th>
<th>Delete</th>
</tr>
</thead>
<tbody>
<tr>
<td>Turn_on TV4</td>
<td>TV4_Off, No_Prob., User_loc.4</td>
<td>TV4_On, Action_act</td>
<td>TV_Manag.</td>
</tr>
<tr>
<td>Turn_off TV4</td>
<td>TV4_On, No_Prob.</td>
<td>TV4_Off, Action_act</td>
<td>TV_Manag., User_loc.6</td>
</tr>
<tr>
<td>Turn_on TV6</td>
<td>TV6_Off, No_Prob., User_loc.6</td>
<td>TV6_On, Action_act</td>
<td>TV_Manag.</td>
</tr>
<tr>
<td>Turn_off TV6</td>
<td>TV6_On, No_Prob.</td>
<td>TV6_Off, Action_act</td>
<td>TV_Manag., User_loc.6</td>
</tr>
<tr>
<td>Turn_on TV7</td>
<td>TV7_Off, No_Prob., User_loc.7</td>
<td>TV7_On, Action_act</td>
<td>TV_Manag.</td>
</tr>
<tr>
<td>Turn_off TV7</td>
<td>TV7_On, No_Prob.</td>
<td>TV7_Off, Action_act</td>
<td>TV_Manag., User_loc.7</td>
</tr>
</tbody>
</table>

We design three service BSNs and one common BSN. TV management, radio management, and light management are service BSNs. These BSNs control each device. “Solve Problem” BSN is a common BSN. This BSN solves the problem that the actions of the service BSNs occur.

Table 2 shows the details of the condition list of action nodes in the “TV management” BSN. In this BSN, the “Solve_Prob.” action node is the common actions. When selecting this action, the problem solve BSN is conducted as a common BSN. In addition, the “Identify_Prob.” action node is environment node.

By using these environment nodes, the system can know the result of action in real world. In the implemented system, the smart home service simulator sends the result of the selected action node to BSN module. Through this method the system can check the result in real world. Figure 3 and 4 show the “TV management” BSN and the ‘Solve problem’ BSN, respectively.

C. Planning design using STRIPS language

The STRIPS planning system does not plan the sequences of all primitive behaviors or trajectories, but plans the sequences of sub-goals to control behavior network modules. The system should be controlled explicitly to achieve the global goal through the sequence of several independent behavior modules as sub-goals. The planning system performs global tasks with several sub-goals correctly in complex environments but the behavior modules only deal with current situations and one corresponding sub-goal.

The implemented planning system using STRIPS language is to generate the sequences with global task. This language constructs plans from scratch, based on primitive action descriptions and planners using pre-defined Plan Decompositions Hierarchies [16]. We design actions and plan decompositions. In traditional STRIPS fashion, actions have preconditions, or facts that must be true for the action to occur, and effect, or facts that are true in the world after the action.
occurs. They directly relate to what is changed in the world state: When an action is executed, the preconditions are removed from the world state and the effects are added. For instance, “Control(device)” action is defined as follows.

**Action: Control(device)**

**Mode:** Management device

**Precondition:** AND (At (device), Info (device, state))

**Effect:** Finish (device)

The control action aims to manage a device. The precondition of this action is “At(device)” and “Info(device, state)”. Function of “At(device)” represents the specific name of a device and Function of “Info(device, state)” represents the state of the device. The effect of this action is “Finish(device)”. For example, if the system needs to perform management TV, the function defined “Control(TV)” and this function can execute, when device is TV and the state of TV is on. When the function is executed, the TV management is finished.

Pre-defined actions are also organized in plan decompositions, whose detail is how one plan can be executed by performing a sequence of component action. Plan decomposition is a two-level hierarchy, the first level consisting of a series of action function, and the second level consisting of their respective BSNs or functions. Here is an example of plan decomposition:

**Plan: Saving Energy**

Acquire the state of device →
Management device →
Finish Saving Energy

Figure 5 shows the flowchart of saving energy. In this figure, the step of ‘Acquire the state of device’ is conducted using “Get info(device)” function. The step of ‘Management device’ is conducted using “Control(device)” and “Pass(device)” function. When the precondition of “Finish()” is satisfied, it is the complete of generating the sequence. In this way, the system generates the service sequences appropriate to the current situation.

The generated sequences that consist of the set of BSNs are pushed in the sequence queue. The algorithm for performing the sequences is showed as Figure 6. In this algorithm, the system performs BSNs according to the sequence. If a single service BSN completes the goal, the system deletes this BSN from the sequence queue. Then, the system gets the next BSN from the sequence queue and conducts it. If it does not have next BSN, the system sends to user the message of complete of service. If the error or exception is occurred in this process of execution, the system uses the pre-defined exception handling function.

**Input:**

Complex service sequence $S_c = \{B_{s1}, B_{s2}, ..., B_{sn}\}$

**Output:**

Execution completion of sequence of complex service

**Repeat:**

If (Execution of single service BSN $B_{si} = \text{Yes}$)

Delete $B_{si}$ in complex sequence queue

If (Next single service BSN $B_{si} = \text{No}$)

Execution completion of sequence of complex service

Else

Execution of next single service BSN $B_{si+1}$

Else

Exception handling

**IV. EXPERIMENTS**

**A. Computation of activation energy of BSNs**

In this section, we verify that the proposed method has lower computation than other methods.

Figure 7 shows the result of comparing computation of several methods. Maes’ BSN, Tyrrell’ BSN, and the proposed
BSN represent to MBSN, TBSN, and PBSN, respectively. We assume that \( \#P, \#S, \text{and} \#C \) are five and \( \#NC, \#EC, \#GC, \#PC, \#SC, \text{and} \#CC \) are two. The rage of the number of environments is five to thirty, and the range of the number of nodes is five to thirty.

As a result, MBSN has more computation and TBSN is the following. At the number of environments and nodes is thirty, the computations of MBSN, TBSN, and PBSM are 1500, 270, and 217, respectively. In addition, the computation is more affected by the number of nodes than that of environments.

### B. System usability test

![Home environment implemented using Unity3D](image)

**Fig. 8.** Home environment implemented using Unity3D

![Intelligent agent using MSAgent and MFC](image)

**Fig. 9.** Intelligent agent using MSAgent and MFC

This section verifies the usability of the proposed system. The smart home environment is implemented using Unity3D tool. Intelligent agent is implemented using MS Agent and MFC. The smart home agent can perform services that manage TV, radio, light and so on. In addition, the saving energy service as the complex service conducts the management of all TV, radio, and light according to the current state of devices. We design various scenarios, and here is one of them.

On Monday, at 9:00 a.m., the user oversleeps and hurries to prepare to go to office. He goes out of home without turning off the TV in the living room and the light in the kitchen. He sends message of saving energy to the agent. The agent analyzes his message and decides the complex message. The agent makes a sequence using the STRIPS planning system and performs the BSN modules according to the sequence.
Because the scenario is the complex request, the system uses two BSNs: “Light management” BSN and “TV management”. Figure 10 illustrates the change of activation levels to respond to saving energy command in the scenario. In this figure, the action node is selected four times. When \( t \) is seven, the system selects the “TV4_Off” action and “Identify_Pro” action is selected at \( t = 14 \). Then, “L5_Off” action and “Identify_Pro” action in “Light management” BSN are selected. The system identifies the problem that whether the action is performed or not in real word using each “Identify_Pro.” action.

V. CONCLUDING REMARKS

In this paper, we have proposed a smart home agent and a method of generating behaviors in the agent. This smart home service agent responds to the user’s request using STRIPS planning and BSN modules. This hybrid system overcomes the limitation of classical planning and reactive system. In addition, the designed method of two types of BSN module has lower computation than conventional methods. The system classifies the user’s requests to the single services or the complex services that are sets of single services and responds appropriate to the services. We design three BSNs: Management of TV, radio, and light. Also we design one STRIPS plan: Saving energy. The smart home environment is implemented using Unity3D and is verified the usability of the system. For future works, the system will be improved through design to more BSN modules and STRIPS plans.

REFERENCES