# Reliability analysis of mobile agent control system with multiple alternative plans

# Reliability Analysis of Mobile Agent Control System with Multiple Alternative Plans

Xia Wang[1*], Yang Xu[2], Jun Liu[3] and Keming Wang[1]

[1*]School of Computing and Artificial Intelligence, Southwest Jiaotong Univerisity, Chengdu, 600031, China.
[2]School of Mathematics, Southwest Jiaotong Univerisity, Chengdu, 600031, China.
[3]School of Computing, Ulster University, Belfast, Northern Ireland, BT15 1ED, UK.

*Corresponding author(s). E-mail(s): Kelly_Wang@my.swjtu.edu.cn;

**Abstract**

With the advancement of artificial intelligence technologies, mobile agents are becoming more commonly used in a variety of industries that require high reliability from their control systems. In an uncertain environment, if the mobile agent control system's state transition includes only one plan, the system will enter the fault state immediately after the plan fails. Therefore, multiple alternative plans can be provided during the system design process to improve system reliability. First, this paper studies and describes the factors associated with the proposed multiple alternative plans, namely the success rate and plan implementation cost. Second, a Policy Generation Algorithm (PGA) for identifying an appropriate execution sequence of those alternative plans is proposed. Furthermore, we propose a formal method-based pipeline framework for verifying the reliability of a mobile agent control system equipped with multiple alternative plans: we invoke the probabilistic model checking technique to create a Discrete-Time Markov Chain (DTMC) formal model of the mobile agent control system, convert the required properties into Probabilistic Computation Tree Logic (PCTL) formulae, and verify the model using the advanced probabilistic model checker PRISM. A case study is provided to demonstrate the applicability of the suggested methodological framework. The experimental results demonstrate that the proposed mobile agent control system with multiple

alternative plans can improve system reliability while also meeting the least expected operational cost under the alternative plan set.

**Keywords:** probability model checking, uncertain environment, control system, multiple alternative plans, decision tree

# 1 Introduction

An agent is an entity that can function continuously and autonomously in a given environment with characteristic such as residency, responsiveness, sociality, and initiative. The present mobile agent control system has been gradually turned into an automatic operation mode as artificial intelligence and computer technology have progressed. Several related studies have produced important findings. According to accident reports, there are still crucial subjects in current intelligent technology applications that cause failures or accidents [1]. It has been noted that the most essential element of using intelligent control systems in mobile agent is to secure the control system's safety and reliability while achieving intelligence. Although a large number of groups have started research to improve the safety of intelligent control systems, the current research has not proven to be sufficient to cover the actual requirements.

In a defined environment, the control system can be planned in such a way that the mobile agent can complete the goal task or state transfer by a set of actions [2, 3]. For example, through reinforcement learning, an agent can realize the automatic exploration of the environment, gain experience, and then use that experience to select the efficient route. In an uncertain environment, however, the performance ability will be substantially decreased [4–6].

In industry, especially in safety-critical fields, environmental uncertainty cannot be completely eliminated and ignored. There are many related studies on the feature selection of the uncertainty [7, 8]. Therefore, it is very important to improve the self-adaptability of mobile agents in uncertain environments and the fault tolerance of control systems.

A general system has a plan that can be used to implement transitions between two adjacent states. It is worth noting that a plan can be a planned path between two locations, a program that implements two state transitions, or a hardware device that implements a function. When the plan fails, the system falls into failure. Improving the plan's quality can effectively improve the reliability of the control system. However, in an uncertain environment, no one can guarantee that the plan will always be executed successfully and complete the task. As a result, it is crucial to consider alternatives to anomalous situations when designing a system. In this paper, we propose a control system design method with multiple alternative plans while considering the expected cost of execution. In addition, a Policy Generation Algorithm

(PGA) is designed based on decision tree analysis, which determines the execution sequence of alternative plans between states.

Designers can design control systems equipped with multiple alternative plans for implementing state transitions and the sequence in which the alternative plans are executed. In addition, it is very important to verify the reliability of the control system and required properties. Formal methods are the system development method based on precise mathematical semantics. Its primary function is to eliminate ambiguity in system design. The application of formal methods in the development of safety-critical systems is highly recommended in the safety standard specification EN50128/EN50129 [10]. Model checking is a technology that can be used to ensure that the system model satisfies all of the required properties, including the safety. In recent years, many industrial organizations have begun to adopt model checking [9]. Probabilistic model checking is widely used to verify the qualitative and quantitative properties of systems [11]. In order to analyse the reliability of the proposed control system with multiple alternative plans, the present work formalises the state transfer process of the control system based on the DTMC probabilistic model, represents the relevant and required properties using the PCTL logic formula, and verifies the system model using the probabilistic model checking technique.

In summary, the main contributions of this paper are as follows:

- Proposes to add multiple alternative plans for transferring between states in the mobile agent control system to the existing single plan design, which is meant to keep the system from rapidly entering a fault state owing to a single plan failure.
- Proposes a Policy Generation Algorithm (PGA) based on a decision tree analysis approach for selecting an appropriate plan execution sequence based on the relevant parameters of each plan. With multiple alternative plans, only one plan needs to be adopted at a time between states. Another alternative plan can be adopted in time after the failure of one plan. The different plan execution sequence has a significant impact on the expected cost of the control system. For a control system with multiple alternative plans, this method seeks to select the execution sequence with the lowest expected cost.
- Models the control system with multiple executable plans by the Discrete Time Markov Chain (DTMC), formalizes the specified properties using PCTL formulae, and verifies the DTMC model using the probabilistic model checker PRISM. The goal of this solution is to ensure ambiguity-free state transfer of control systems with multiple alternative plans based on the strict semantic characteristics of formal methods, and the reliability of the state transfer of the designed system can be analysed based on the results of probabilistic model checking.
- Compares and examines finally the control system with multiple alternative plans vs. a single plan, and the results demonstrate that the control system with multiple alternative plans is more reliable.

The rest of this paper is organized as follows. Section 2 shows a collection of related works. The background and preliminary knowledge are reviewed in Section 3. Section 4 introduces the methodological framework. Section 5 further illustrates the proposed framework with a case and discusses the verification results. Section 6 concludes the work and outlines the future work.

## 2 Related Works

This literature review focuses on the reliability of the mobile agent control system and formal verification of the control system model. The goal of mobile agent research is to successfully organize the agent's mobility so that it can go from one state to another along a predetermined plan. It can also be understood as the transition of the states, that is, the transition from the current state to the target state. In recent years, research on agent control has made great progress [12]. Lamini et al. suggested a new fitness function that considers the distance, the safety and the energy of robot path planning [13]. These methods include using the Genetic Algorithm (GA) with the improved crossover operator. To adapt the robot to a dynamic environment, Wang et al. proposed the learning-based methods that can be used to explore spatio-temporal information of the environment [14]. A real-time obstacle avoidance decision model based on Machine Learning (ML) techniques was proposed by Song et al. [15], as well as an improved Smooth Rapidly exploring Random Tree (S-RRT) algorithm.

In a static environment, the agent completes the state transition according to the established plan, which is of course the best choice [16, 17]. However, in an uncertain environment, exceptions may occur at any time, and the established plan may fail, resulting in failure of the control system [18]. The research on agents in uncertain environments is also very important and has attracted the attention of many researches [19–24]. For instance, Kim et al. developed an optimization approach to increase profit from reliability enhancement for a given target [25]. The ideal option reveals that not all subsystems are upgraded, and the gap between the severity of the failure and the rate of growth in the improvement cost determines the priority of subsystem improvement. Rungskunroch et al. developed a benchmark risk model which is a linear transform model based on posterior probability, and conducted the analysis through the decision tree and probability tree [26]. Musharraf et al. investigated how different properties of emergency scenarios influence people's choice of egress route subsequent to training [27]. Wu et al. proposed a novel approach to study the impacts of communication delays on the safety and availability of safety-critical DNCSs based on the formal language of timed colored Petri nets (CPNs) [28]. Farhadi et al. used the probability tree and Markov chains approach to determine the optimal number of spare parts. In this method, they can minimize the system's total cost and or system's total availability [29]. The majority of these studies aim to raise the quality of the plan in order to increase system reliability. It goes without

saying that raising the quality of the plan is crucial for raising system reliability. There are still drawbacks to this strategy, though, as there is never a guarantee that a plan won't be impacted by a changing environment.

Unlike a control system equipped with a single plan, a control system with multiple alternative plans can adopt alternatives immediately after one plan fails to continue the state transfer process until it moves to the next state or all plans have failed. We have yet to find a design solution that directly proposes using multiple alternative plans, but there is some similar work. Research similar to the equipping multiple alternative plans referred to in this paper includes redundant reserves of equipment components. To deal with the problem of redundant device assignment in operating conditions with uncertainty, Baladeh et al. developed a novel dynamic k-out-of-n system in which possible operating conditions are modelled using discrete scenarios [30, 31]. In addition, the execution sequence of alternative plans affects the expected cost of the overall control system. Thus, this paper adds the analysis of the expected cost to the proposed control system design equipped with multiple alternative plans and designs a policy generation algorithm to select an appropriate plan execution sequence.

Obviously, with the addition of executable plans, the complexity of the system increases. A significant part of safety research is focused on control system. Probabilistic model checking techniques can be used to model complex systems and verify that the designed system satisfies specified qualitative and quantitative properties. Numerous researchers have successfully used probabilistic model checking in a variety of domains. Zheng and Bolton et al. tried to address some deficiency by extending formal, task-based, verification methods with concepts from human reliability analysis and probabilistic/statistic model checking [32, 33]. Gao et al. evaluated the financial production risk by establishing the optimal cash reserve ratio based on probabilistic model checking [34]. Wan et al. proposed the method that reduces knowledge verification into probabilistic computation tree logic verification [35]. Zhang et al. developed a framework for probabilistic model checking on a layered Markov model to verify the safety and reliability requirement of robotics [5]. Feng et al. applied abstractions to stochastic two-player games based on Markov decision processes, depending on the sort of knowledge regarding uncertainties and imperfections in the operator autonomy interactions [36]. To improve the work efficiency and reduce delivery costs, Zhu et al. applied the probabilistic model checking method [37]. Wang et al. proposed a behavioural hierarchical analysis framework in smart home based on probabilistic model checking [38]. These fascinating studies demonstrate that probabilistic model checking has developed into one of the most efficient methods for system verification.

This paper suggests adding alternative plans for control systems based on previous research to enhance the reliability of mobile agents in uncertain environments. In addition, considering the cost incurred by increasing the executable plans, the cost expectations under various execution sequences are

analysed by a decision tree. We design an algorithm for generating execution policies to determine an appropriate sequence to carry out plans. The DTMC model is developed according to the execution sequence of the plans and is verified using probabilistic model checking technology. The system's reliability is assessed based on the verification results.

# 3 Preliminaries

## 3.1 Probability formula

In a mobile agent control system, a plan to implement state transfer of an agent represents a collections of actions. In an uncertain environment, when an agent executes a plan, there is a certain probability that the current plan will fail to perform the task due to many possible unexpected factors. If there is only one executable plan, the system will go into a failure state as soon as the current plan fails. Therefore, adding alternative plans can effectively reduce the probability of system failure. When an agent is about to move from one state to another, the control system executes one of the alternative plans. If the agent executes the current plan successfully and moves to the next state, the rest of the plans are not executed. If the environment is abnormal and the execution of the current plan fails, the system can activate a reserved plan and continue to complete the state transfer task. The system fails if all of the plans failed.

Under multiple alternative plans, the success probability of transition from one state to the next can be calculated with (1).

$$\begin{aligned} P_{suc}(S_i \to S_j) = P_1 + (1 - P_1) \cdot P_2 + ...+ \\ (1 - P_1) \cdot (1 - P_2)... \cdot (1 - P_{n-1}) \cdot P_n \end{aligned} \tag{1}$$

$P_{suc}(S_i \to S_j)$ denotes the probability of the agent move from state $i$ to state $j$. $P_n$ indicates the probability that execution Plan $n$ successfully completes the task. According to (1), the following two theorems can be deduced.

**Theorem 1** *The success probability of the state transition is only related to the number of executable plans and the probability of executing the plan to complete the task, not the sequence of execution.*

*Proof* From the standpoint of system failure, it can reach the next state as long as no system failure happens. The probability of implementing state transfer is the total probability value minus the probability of system failure. As a result, the probability of success can be represented by (2). □

$$P_{suc}(S_i \to S_j) = 1 - (1 - P_1) \cdot (1 - P_2) \cdot ... \cdot (1 - P_{n-1}) \cdot (1 - P_n) \quad (2)$$

From Theorem 1, it can be stated that the execution sequence of the alternative plans has no effect on the probability of the control system successfully completing the state transition. Therefore, if the number of plans is consistent, the total probability of state transition can be ignored when we perform cost analysis on the control system.

**Theorem 2** *A control system with multiple alternative plans has a greater probability of successfully completing a state transition than a control system with a single plan.*

*Proof* When there is only one plan, the system will enter the fault state as soon as the plan fails. When a system contains many plans, if one fails, another can be implemented, and the system will only fail and shut down if all plans fail. $P_{suc}(S_i \xrightarrow{n-1} S_j)$ represents the probability of transferring from $S_i$ to $S_j$ with (n-1) plans. $P_{suc}(S_i \xrightarrow{n-1} S_j) = 1 - (1 - P_1) \cdot (1 - P_2)... \cdot (1 - P_{n-1})$, $P_{suc}(S_i \xrightarrow{n} S_j)$ represents the probability of transferring from $S_i$ to $S_j$ with n plans plans. $P_{suc}(S_i \xrightarrow{n} S_j) = 1 - (1 - P_1) \cdot (1 - P_2)... \cdot (1 - P_{n-1}) \cdot (1 - P_n)$. $P_n$ represents the probability of success of plan n, and its domain is, $0 < P_n < 1$, therefore, $0 < 1 - P_n < 1$ is satisfied. $(1-P_1)\cdot(1-P_2)...\cdot(1-P_{n-1}) > (1-P_1)\cdot(1-P_2)...\cdot(1-P_{n-1})\cdot(1-P_n)$, and $1-(1-P_1)\cdot(1-P_2)...\cdot(1-P_{n-1}) < 1-(1-P_1)\cdot(1-P_2)...\cdot(1-P_{n-1})\cdot(1-P_n)$. Therefore, $P_{suc}(S_i \xrightarrow{n-1} S_j) < P_{suc}(S_i \xrightarrow{n} S_j)$. Consequently, adding alternative plans may improve the system's reliability. □

## 3.2 Discrete Time Markov Chain (DTMC)

When a stochastic process possesses the Markov property, the conditional probability distribution of its future state depends solely on the current state, given the current state and all previous states.

**Definition 1** (Markov process)[39]. Suppose the $\{X(t), t \in T\}$ is a stochastic process, **E** is the state space, if $\{t_1 < t_2... < t_n < t\}$, any $x_1, x_2, ..., x_n, x \in \mathbf{E}$, the conditional distribution function of the random variable $X(t)$ under the know variable $X(t_1) = x_1, ..., X(t_n) = x_n$ is only related to $X(t_n) = x_n$, but not related to $X(t_1) = x_1, ..., X(t_{n-1}) = x_{n-1}$.

DTMC is a special form of Markov process, which have discrete values in the general definition by implementing the time parameters and the state space.

**Definition 2** (DTMC)[40]. A DTMC is a tuple $\mathbb{D} = (S, s_0, P, L, AP)$.

where,

- $S$ is a finite set of states.
- $s_0$ denotes the initial state.
- $P$ is a probabilistic transition function, such that for every state $s \in S$, and $\Sigma_{s' \in S} P(s, s') = 1$.
- $L : S \to 2^{AP}$ is a labelling function that assigns a set of atomic propositions to each state $s \in S$.
- $AP$ is a finite set of atomic propositions.

### 3.3 Probability Computation Tree Logic (PCTL)

As for model checking for the DTMC model, Probability Computation Tree Logic (PCTL) is widely used to specify properties.

**Definition 3** (PCTL syntax). Given a set of atomic propositions AP, the PCTL formulae are defined by the following grammar [41]:

$$\varphi ::= p \mid \neg\varphi \mid \varphi \vee \phi \mid \mathbb{P}_{\bowtie m}(\psi)$$
$$\psi ::= X\varphi \mid \varphi \cup \phi \mid \varphi \cup^{\leq k} \phi$$

where:

- $p \in AP$ is a finite set of atomic propositions.
- $\varphi$ and $\phi$ are the state formulae and $\psi$ is path formulae interpreted over the states and paths of $M$, respectively.
- $\neg$ and $\vee$ are the boolean connectives that defined in the usual way. $\neg$ is for negative. $\vee$ denotes disjunction.
- $\mathbb{P}_{\bowtie m}(\psi)$ is a probabilistic operator where $\bowtie \in \{<, \leq, \geq, >\}$ and $m \in [0, 1]$ is a probability bound or threshold.
- $k \in \mathbb{N}^+$ is a positive integer number reflecting the maximum number of transitions needed to reach a certain system.
- $X, \cup$ are defined as 'next' and 'until', respectively.

A state $s$ in a DTMC model satisfies an atomic proposition $p$ if $p \in AP$. $s$ satisfies a state formula $\mathbb{P}_{\bowtie m}(\psi)$, denotes $s \models \mathbb{P}_{\bowtie m}(\psi)$, if the probability of taking a path starting from s and formula $\psi$ satisfies the bound $\bowtie m$. The path formula $\varphi \cup^{\leq k} \phi$ is true on a path if $\phi$ holds in the state at several time step $i \leq k$ and at all preceding states $\varphi$ holds. In addition, the equivalences $F\varphi \equiv true \cup \varphi$ are often used. $F$ denotes reachability, which means $\varphi$ eventually becomes true.

## 4 Methodological Framework

We recommend integrating several alternative plans while designing a mobile agent control system in uncertain environments. Adding alternative plans is one of the ways to avoid the mobile agent from entering the downtime state immediately after a plan fails. Its purpose is to increase the replacement plan, which can be used immediately after a plan fails. However, when there

are multiple alternatives and the characteristics and costs of each plan are different, choosing an appropriate execution sequence is very important to save execution costs.

In industrial applications, constructing a dependable system necessitate not just the system's reliability but also its cost. A plan's success rate and cost are expected to be proportional in this paper. To put it another way, the higher the success rate, the raise the cost. The system can only use one plan at a time, and another plan can be activated only if the current one fails. Note that once the plan is executed, whether it succeeds or fails, it will consume the corresponding cost. For example, if the system execute Plan 1 first, it takes 5 costs. If the Plan 1 is accomplished and the system enters the goal state, then the total cost of the task is 5. If the Plan 1 fails, Plan 2 with a cost of 7 needs to be executed, and if the Plan 2 succeeds, the total cost of consumption is 12. If the Plan 2 is adopted first, the cost of Plan 2 success is 7, and if the Plan 1 is adopted after the Plan 2 fails, the total cost is 12. As a result, different execution sequences of the plans will create diverse effects when there are several plans. For a more detailed explanation, we'll utilise Example 1.

**Example 1**: Fig. 1 illustrates an example that lists multiple plans to realize states' transition. In a simple system, there are two states, $S_0$ and $S_1$. Plan A, Plan B, and Plan C are the executable plans provided for accomplishing the transitions from $S_0$ to $S_1$. The cost of implementing Plan A is $C_a$, and the probability of successfully transiting from $S_0$ to $S_1$ by using Plan A is $P_a$. The cost of using Plan B to attain the S1 state is $C_b$, and the likelihood of success is $P_b$. Plan C has a $P_c$ probability of reaching state S1 and a $C_c$ cost for execution.

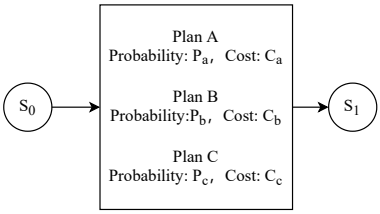


**Fig. 1** The example of states transaction.

It can be observed that there are two states, and the system needs to transfer from state $S_0$ to $S_1$. The designer provided three plans, and the probability of successfully reach $S_1$ and expense of executing each plan are differ. There is no need to execute the subsequent plan if the state $S_1$ is successfully attained after implementing a specific plan. The sequence of executing plans need to be confirmed before develop the mobile agent control system.

Combined with the description of Example 1, it can be learned that in designing a control system equipped with multiple alternative plans, the

parameters of each plan, such as the successful implementation rate and cost, should be counted first. Second, the designer needs to choose a proper plan execution sequence. Finally, the reliability of the control system based on multiple alternative plans needs to be analysed. We will analyse each part and propose an overall methodological framework.

## 4.1 Plan evaluation

Plan evaluation is to evaluate multiple alternative plans in the current state, and consider the probability and cost that a plan can successfully enter the next state.

Equipment failure, human reasons, dynamic obstacles, and other factors can all contribute to planning failure. There is a variety of studies that may be used to estimate the probability of a plan's success [42, 43, 51].

The cost of the plan needs to be evaluated in terms of budget based on the equipment, energy consumption, and time required for the plan implementation. The components vary from plan to plan, so the cost also varies. In general, the higher the cost, the higher the quality. A high-quality implementation plan has a higher probability of success and demands higher-quality components when building a plan. A low-quality plan with low-cost components, on the other hand, has a reduced likelihood of success. These work is not the main content of our work, we just need to know the probabilities and cost of each plan before designing the model of the control system.

## 4.2 Policy selection based on decision tree and PGA

A decision tree provides a graphical representation for comparing alternatives and allows decision analysis by combining uncertain and cost values for executable plans. The decision tree analysis approach can be used to determine the sequence of the executable plans while considering the reliability of system state transition. In this paper, we propose to use the analysis method of decision tree in policy selection. The states are represented by circles, while the state transition is represented by rectangles, which contain the probability of plan accomplishment and demand cost. Transitions are shown with arrows. See Example 1, for the state transition relationship in Fig. 1, the analysis based on decision tree is shown in Fig. 2. We define each of the execution sequences as a policy. The intention is to find an appropriate policy with the lowest Expected Cost Value (ECV).

The decision tree shows that the designer provides three executable plans for the state transition from $S_0$ to $S_1$. Taking the path marked in red in Fig. 2 as an example. At this time, Plan A is selected as the preferred plan, the probability of execution Plan A reaching $S_1$ is $P_a$, the cost is $C_a$, and the ECV is $C_a$. If Plan A fails, Plan B is adopted, the probability of success is $P_b$, the cost is $C_b$, and the ECV is $(C_a + (1 - P_a) \cdot C_b)$. It is worth noting that the expected value of a policy is equal to the sum of all possible expected values to

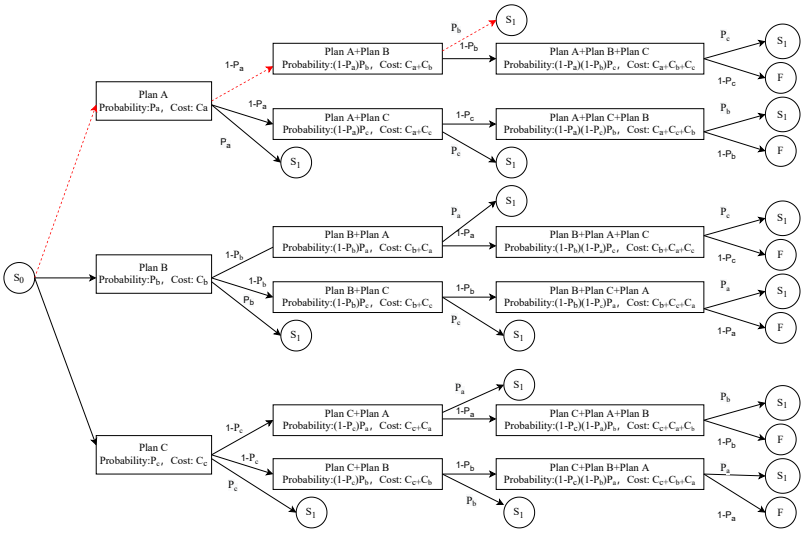


**Fig. 2** The example of states transition.

reach the final state. Use (3) to record the ECV of each policy, compare and select the execution sequence with the lowest ECV as an appropriate policy for system modelling.

$$\begin{aligned} ECV_{i\to j} = C_1 + (1-P_1)\cdot C_2 + ...+ \\ (1-P_1)\cdot(1-P_2)...\cdot(1-P_{n-1})\cdot C_n \end{aligned} \tag{3}$$

In addition, we propose a PGA algorithm to solve the ECV under each policy in the decision tree. First, the set of plans that can realize state transition included in the system, as well as the success rate and cost executions of each plan, are defined. Listing all possible plan execution sequences. A conceivable ordering is considered a policy. Calculating the cumulative expected cost value corresponding to all policies using (3). Finally, the resulting policies and ECV are sorted, and the results are outputted. The policy with the lowest cost expectation is the foundation for modeling the system. Algorithm 1 is the pseudocode for the PGA algorithm.

## 4.3 Probabilistic model checking

The goal of probabilistic model checking is to ensure that the probabilistic model meets the relevant qualitative and quantitative properties. It is to verify whether certain property is satisfied qualitatively, and to verify the probability that a certain property is satisfied quantitatively.

The probability symbolic model checker PRISM was created at the University of Birmingham and improved at the University of Oxford [11]. It is a useful tool for formal modelling and study of systems that behave

**Algorithm 1** Policy Generation Algorithm (PGA)

**REQUIRE** A set of plans, including plan's successes rate and costs.
**ENSURE** Expected cost value for each policy.

1 : List all possible plan execution sequences to get different execution policies;

2 : Apply (3) to sum ECV for all policies;

3 : Sort all policies in ascending order based on the expected cost value of each policy.

4 : Output the policy with the smallest ECV.

in a stochastic or probabilistic way. It has been used to examine systems in a variety of application domains, including e-business [44], robot control [45, 46], security protocol [47], etc [48]. The model in PRISM subsumes some well-known temporal logics by including PCTL, Continuous Stochastic Logic (CSL) [49], and Probabilistic Logic (PL) [50]. PCTL is a well-known temporal logic for probabilistic verification of DTMC model.

As mentioned previously, the mobile agent control system in an uncertain environment is designed with multiple alternative plans, and an appropriate plan execution sequence is obtained through the PGA algorithm. Based on the probabilistic model checking technology, we propose to build a DTMC formal model that represents the states transitions of the control system with multiple alternative plans. We list the properties to be verified and the corresponding PCTL logic formulae for verification. The system's reliability is assessed based on the collected results.

## 4.4 Framework

In conclusion, this paper provides a methodological framework integrating system model design and formal verification for mobile agent control systems working in uncertain environments.

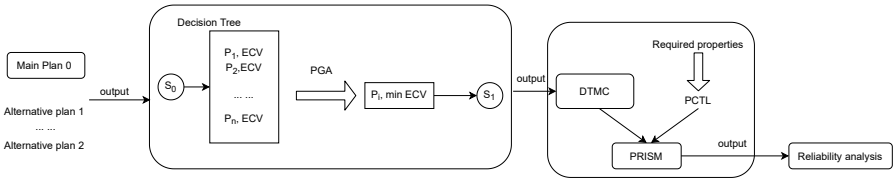


**Fig. 3** The verification framework of the mobile agent control system with multiple alternative plans.

The framework are depicted in the Fig. 3, which includes some steps. The first is the plan evaluation. Through the first stage, the execution cost of the proposed plan and the probability of successful completion of the task can be obtained. These data are used as the output of the first stage and input to

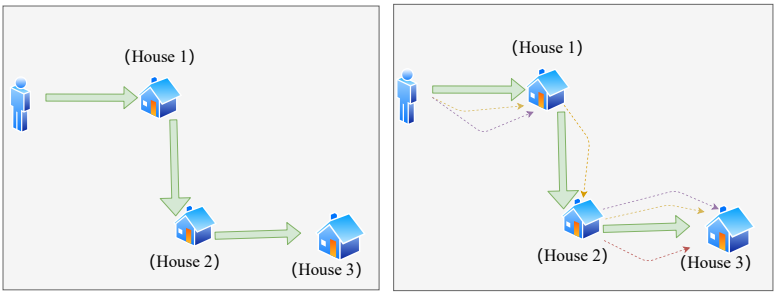


**Fig. 4** Schematic diagram of robot (agent) movement in a smart hotel. The diagram on the left represents the traditional planning model. The agent is in the initial position (marked as state $S_0$), starting from $S_0$, the agent needs to pass through House 1 (marked as $S_1$), House 2 (marked as $S_2$), and finally arrive at House 3 (marked as $S_3$). Green Arrows indicate realization paths (marked as specific plans). The diagram on the right adds a plan to implement the state transition. Dotted arrows of different colors indicate different plans.

the next stage. The second stage is the generated policy based on the PGA algorithm, that is, the analysis and comparison of the executable plans, and the selection of an appropriate execution sequence as the input of the third stage. The third stage is probabilistic model checking. The DTMC formal model is developed in accordance with the optimal execution sequence, the required properties formula are depicted by PCTL in accordance with the specification, and the system model is verified by the probabilistic model checker PRISM. Finally, the system's reliability is assessed based on the verification results.

# 5 Case Study

To better demonstrate the effectiveness of the proposed method in this paper, we use an example for further illustration. The rapid development of artificial intelligence has brought great convenience to people's daily life. In contemporary society, automated robots have been introduced in many fields to reduce manpower consumption and improve results. For instance, manufacturing industry, smart home, intelligent hotels. In the midst of an epidemic, the choice of robot waiters as a delivery tool for international hotels is certainly a great one. But how to guarantee the reliability and cost saving of the robot waiter (can be regard as an agent) movement is worthy of in-depth study.

Fig. 5 shows a simplified scenario. In a smart hotel, there is a robot waiter in charge of three houses with tasks including food delivery and item supply. At the initial state, the robot is in the lobby, waiting to deliver items to each room (the current state is labelled as $S_0$). The sequence in which the robot transports items each time is, first to House 1 (labelled as $S_1$), then to House 2 (labelled as $S_2$), and finally arrives House 3 (labelled as $S_3$). Arrows between states indicate plans to implement state transitions. A plan is a set of actions that can be used to achieve the agent's state transition. The image on the left is a traditional system design. A green arrow between every two locations

indicates a plan. For example, from the initial location to House 1, the agent can reach $S_1$ through the plan. We know that the environment is random, the current plan may fail during the implementation process. At this time, if the plan fails, the task fails, and the robot waiter is no longer able to migrate. The figure on the right shows our proposed method. That is, adding alternative plans to a plan pool. A plan may fail due to various factors such as hardware failure, software weakness, and environmental changes, resulting in system failure. As a result, more plans for state transitions can be considered while creating a mobile agent control system. When a plan fails, an alternative plan in the plan pool can be taken immediately to reduce the probability of task failure. Dashed arrows of different colors indicate different plans. For example, between House 1 and House 2 in addition to the green arrow, there is an orange dashed arrow, indicating that there are two plans can be used to achieve the robot waiter from House 1 to House 2.

In this scenario, the robot waiter is abstracted into an agent. In fact, the movement of the agent can also be considered as state transfer. To ensure that the agent may effectively transition from one state to another, it must engage appropriate activities. We assume that the agent will move from state $S_0$ to the goal state $S_3$. To begin with, it is obvious that the agent's goal is to reach state $S_3$, and the designer has to provide a control system that will allow the agent to achieve this goal safely and accurately. Designers should strive to increase the control system's reliability when working in an uncertain environment. It is impossible to guarantee that a particular plan will be able to finish the task while designing a system. As a result, the plans significantly impact the system's effective.

Once we have developed the plans, we need to confirm the probability of success of each plan and the cost of implementation. The probability that the plan will be implemented successfully is determined through statistical analysis employing a significant amount of pertinent data. The cost of the plan needs to be evaluated in terms of budget based on the equipment, energy consumption, and time required for the plan implementation. These (probabilities and costs) all require detailed evaluations of the specific plan, which are beyond the scope of this paper. The case study is mainly used to demonstrate the effectiveness of the proposed verification framework for a mobile agent control system with multiple alternatives. Therefore, some hypothetical values are used in this case study. As shown in Fig. 5. The circles represent the states, and the rectangles are the executable plans. The plan labels, the success rate, and the cost of adopting the plan are marked in rectangles. For example, three plans are provided for $S_0$ to $S_1$. Plan 1 has a probability of success of 0.7 and a cost of 7. Plan 2 has a probability of success of 0.8 and a cost of 12. Plan 3 has a probability of success of 0.9 and a cost of 15.

There are different success rates and cost for different plans. The probability of completing the state transition is increased when there are multiple executable plans. However, the system can only execute one plan at a time, and it only switches to another plan if the first one fails. The execution

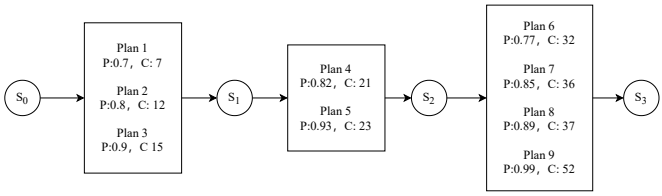


**Fig. 5** A case of state transition of mobile agent control system.

sequence of the plans has different impact on the cost of the system. For example, if Plan 1 is successful, the state transition only consumes 7 costs. If Plan 1 fails and Plan 2 succeeds, the state transition cost needs 19. If Plan 1 and Plan 2 fail and Plan 3 is adopted, the consumption cost is 34. Assuming Plan 3 is considered to be executed first, Plan 3 has a higher probability of successfully completing the task. As a result, it is less likely that other plans will need to be executed, and less likely that additional costs will be required. It is important to decide on a proper sequence of plan execution for the cost savings of mobile agent control in the long term.

## 5.1 Policy generation

The sequence in which plans are executed is referred to as a system policy. As previously stated, the decision tree is used to analyse several different execution sequences. In the mobile agent control system, the entire control process can be separated into three phases in this example, with $S_0$ to $S_1$ being the first, $S_1$ to $S_2$ being the second, and $S_2$ to $S_3$ being the third.

**$S_0$ to $S_1$**: There are three alternative plans for completing the states' transition. The decision tree is depicted in Fig. 6. At state $S_0$, there is a 0.7 probability of getting to state S1 if Plan 1 is used first. There is a 0.3 probability that Plan 1 will fail, thus requiring activation of either Plan 2 or Plan 3. Assuming that Plan 2 is selected as the second execution plan, the probability of a successful transfer to state $S_1$ is 0.8, and there is a 0.2 probability that Plan 3 will be activated due to the failure of Plan 2. The probability that Plan 3 will complete the transfer is 0.9. If Plan 3 fails, then the system enters a fault state.

The second policy is to adopt Plan 1 first in state $S_0$. There is a 0.3 probability that Plan 1 will fail, thereby activating Plan 3, at which point the probability of successfully transferring to state $S_1$ is 0.9, and there is a 0.1 probability that Plan 2 will be activated due to the failure of Plan 3. The probability that Plan 2 will successfully complete the transfer is 0.8. If Plan 2 also fails, then the system enters a fault state. The description is the same for the other policies.

The *PGA* algorithm is used to calculate the ECV of the various sequence combinations of the three plans, producing the results displayed in Tab. 1. The results indicate that there are six policies in total, with the policy's ECV from Plan 1 to Plan 2 to Plan 3 being the lowest. As a result, adopt this policy.

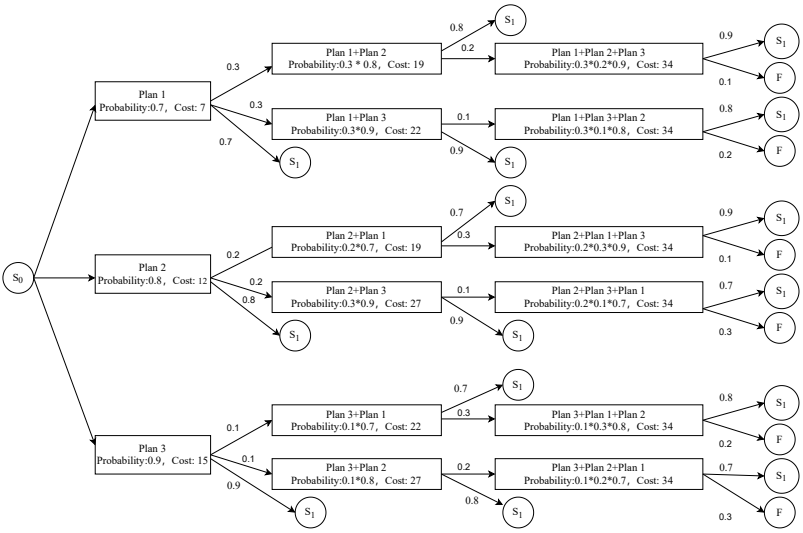


**Fig. 6** The decision tree depicts the state transition from $S_0$ to $S_1$.

**Table 1** Policy compare of the transition from $S_0$ to $S_1$

| Policy | ECV |
|---|---|
| Plan 1 → Plan 2 → Plan 3 | **11.50** |
| Plan 1 → Plan 3 → Plan 2 | 11.86 |
| Plan 2 → Plan 1 → Plan 3 | 14.30 |
| Plan 2 → Plan 3 → Plan 1 | 15.14 |
| Plan 3 → Plan 1 → Plan 2 | 16.06 |
| Plan 3 → Plan 2 → Plan 1 | 16.34 |

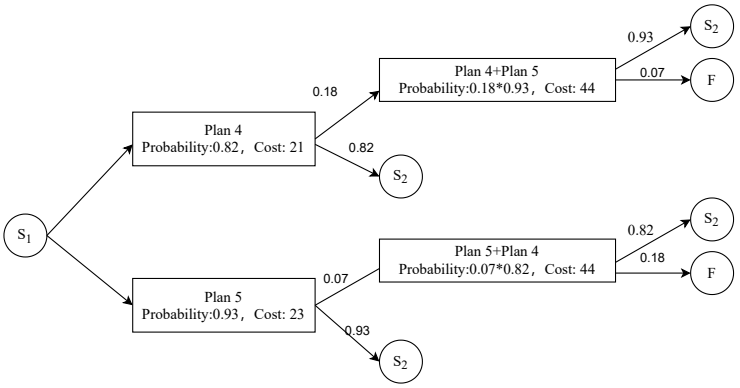


**Fig. 7** The decision tree depicts the state transition from $S_1$ to $S_2$.

**$S_1$ to $S_2$**: There are two plans for completing the states' transition. The decision tree is shown in Fig. 7.

In Tab. 2, the results of the policy generation are shown. The findings show two policies, with the ECV of the policy from Plan 5 to Plan 4 being the lowest. As a result, apply this policy from $S_1$ to $S_2$.

**Table 2** Policy compare of the transition from $S_1$ to $S_2$

| Policy | ECV |
|---|---|
| Plan 5 → Plan 4 | **24.47** |
| Plan 4 → Plan 5 | 25.14 |

**$S_2$ to $S_3$**: There are four plans for completing the states' transition. Note that the decision tree structure from state $S_2$ to $S_3$ is same as that of $S_0$ to $S_1$, except that there are 4 optional plans, and the constructed decision tree is more complicated. We don't provide a detailed decision tree due to space limitation. According to the method $PGA$, 24 policies are obtained. The top ten policies with the lowest ECV are listed in Tab. 3. The policy $Plan\ 6 \rightarrow Plan\ 8 \rightarrow Plan\ 7 \rightarrow Plan\ 9$ is chosen for modelling since it has the lowest ECV.

**Table 3** Policy compare of the transition from $S_2$ to $S_3$

| Policy | ECV |
|---|---|
| Plan 6 → Plan 8 → Plan 7 → Plan 9 | **41.62** |
| Plan 8 → Plan 6 → Plan 7 → Plan 9 | 41.63 |
| Plan 8 → Plan 7 → Plan 6 → Plan 9 | 41.69 |
| Plan 6 → Plan 7 → Plan 8 → Plan 9 | 41.75 |
| Plan 8 → Plan 7 → Plan 9 → Plan 6 | 41.82 |
| Plan 6 → Plan 8 → Plan 9 → Plan 7 | 41.83 |
| Plan 8 → Plan 6 → Plan 9 → Plan 7 | 41.84 |
| Plan 6 → Plan 7 → Plan 9 → Plan 8 | 42.09 |
| Plan 7 → Plan 6 → Plan 8 → Plan 9 | 42.27 |
| Plan 7 → Plan 8 → Plan 6 → Plan 9 | 42.28 |

## 5.2 DTMC model

As stated previously, the plan execution sequence from state $S_0$ to $S_1$ is $Plan\ 1 \rightarrow Plan\ 2 \rightarrow Plan\ 3$, from state $S_1$ to $S_2$ is $Plan\ 5 \rightarrow Plan\ 4$, and from state $S_2$ to $S_3$ is $Plan\ 6 \rightarrow Plan\ 8 \rightarrow Plan\ 7 \rightarrow Plan\ 9$. According to the obtained state and the sequence of plan execution, the corresponding DTMC model can be developed. The DTMC model's transition of states is depicted in Fig. 8.

$S_0$ is the initial state, while $S_1$, $S_2$, and $S_3$ are the states that must be accomplished in order. A status point marked with a $T$ represents a planned transition point. After a state fails, the next plan is adopted immediately. In this system, there are 6 planned switching points, i.e., $T_1$, $T_2$, $T_3$, $T_4$, $T_5$, and $T_6$. For example, $T_1$ represents Plan 1 failed and Plan 2 is executed. $F$ stands for system failure. The system fails when all plans fail during a transition.

The detailed description for the Fig. 8 is,

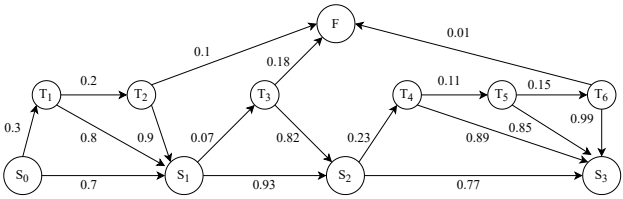


**Fig. 8** A case of state transition of mobile agent control system.

- Starting with $S_0$, Plan 1 is implemented first. Execution Plan 1 has a 0.7 probability of successfully transfer to state $S_1$, and it has a 0.3 probability of being pushed to execution Plan 2 if Plan 1 fails.
- When executing Plan 2, there is a possibility that 0.8 will transfer to state $S_1$, and 0.2 will fail, requiring Plan 3 to be executed.
- When running Plan 3, there's a chance of 0.9 that the agent will transfer to state $S_1$, and 0.1 chance that the agent will fail. If Plan 3 fails, the system becomes sluggish and shuts down.
- Plan 5 is implemented directly once the system enters state $S_1$. Plan 5 has been used to have a 0.93 possibility of successfully entering state $S_2$, and a 0.07 chance of failure to execute Plan 4.
- When Plan 4 is executed, there is a 0.82 probability of entering state $S_2$ and a 0.18 probability of fails, resulting the system failure.
- After the system enters state $S_2$, Plan 6 is first adopted. Plan 6 has been used to have a 0.77 chance of successfully entering state $S_3$, and a 0.23 chance of failure to execute Plan 8.
- When executing Plan 8, there is a possibility of 0.89 transition to state $S_3$, and 0.11 may fail, thus executing Plan 7.
- When executing Plan 7, there is a possibility of 0.85 transition to state $S_3$, and 0.15 may fail, thus executing Plan 9.
- When Plan 9 is executed, there is a 0.99 probability of entering state $S_3$ and a 0.01 probability of fails, resulting the whole system failure.

The PRISM code describing the DTMC model is as follows. The model has an initial state 0 and 9 states, 1 and 2 for the states in the control system, i.e., $S_1$ and $S_2$, respectively. 3 is the target state which includes $S_3$ state and fail. 4...9 represent planned transition states. $d : [0..2]$ denotes the goal value is 0, 1, 2. 0 is the initial value, 1 represents the system fails, and 2 denotes arrive the $S_3$. There are 11 sentences here. The left side of the "$\rightarrow$" represents the current state, and the right side represents the state and probability that may be entered in the next step. For example, $s = 0 \rightarrow 0.7 : (s' = 1) + 0.3 : (s' = 4)$ means that the agent is now in state $S_0$, the probability of entering $S_1$ is 0.7, and the probability of the current plan fails and adopting the next plan is 0.3.

*dtmc*
*module agent_movement*
$s : [0..9]$ *init* 0;
$d : [0..2]$ *init* 0;

$[\,]s = 0 \rightarrow 0.7 : (s' = 1) + 0.3 : (s' = 4);$
$[\,]s = 4 \rightarrow 0.8 : (s' = 1) + 0.2 : (s' = 5);$
$[\,]s = 5 \rightarrow 0.9 : (s' = 1) + 0.1 : (s' = 3) \wedge (d' = 1);$
$[\,]s = 1 \rightarrow 0.93 : (s' = 2) + 0.07 : (s' = 6);$
$[\,]s = 6 \rightarrow 0.82 : (s' = 2) + 0.18 : (s' = 3) \wedge (d' = 1);$
$[\,]s = 2 \rightarrow 0.77 : (s' = 3) \wedge (d' = 2) + 0.23 : (s' = 7);$
$[\,]s = 7 \rightarrow 0.89 : (s' = 3) \wedge (d' = 2) + 0.11 : (s' = 8);$
$[\,]s = 8 \rightarrow 0.85 : (s' = 3) \wedge (d' = 2) + 0.15 : (s' = 9);$
$[\,]s = 9 \rightarrow 0.99 : (s' = 3) \wedge (d' = 2) + 0.01 : (s' = 3) \wedge (d' = 1);$
$[\,]s = 3 \rightarrow (s' = 3);$
*end module*

## 5.3 Verification in PRISM

The experiments are performed on an Intel(R) Core(TM) i5-7200U CPU with 2.50GHz and 2.71 GHz. The PRISM we used for verification is PRISM 4.7 (GUI mode).

Our purpose is to enable mobile agent to complete state transfer with a high probability in uncertain environments, thus improving system reliability. In verification, the PCTL formula $P =?[F(s = 3) \wedge (d = x)]$ is presented to specify properties of the model. The formula means "what is the probability, from the InitState of the model, of terminal enter the target state and obtain a $d$ value". For example, suppose x=1 is considered. The $P =?[F(s = 3) \wedge (d = 1)]$ means "What is the probability that from the initial state to the system fails". In addition, we also listed other properties. As shown in the Tab. 4, No.2 indicates that the probability of reaching state $S_3$ from $S_0$ is 0.98144. No.3 denotes the probability of reaching state $S_1$ from $S_0$ is 0.98148. No.4 indicates the failure probability of the transition from $S_1$. No.5 indicates the accomplish probability of the transition from $S_1$ to $S_3$. The probability of reaching state $S_2$ from $S_1$ is depicted in No.6. No.7 indicates the failure probability of the transition from $S_2$. No.8 indicates the accomplish probability of the transition from $S_2$ to $S_3$.

**Table 4** Verification results in the PRISM

| No | Init State | Properties | Results |
|---|---|---|---|
| 1 | $S_0$ | $P =?[Fs = 3 \wedge d = 1]$ | 0.01856 |
| 2 | $S_0$ | $P =?[Fs = 3 \wedge d = 2]$ | 0.98144 |
| 3 | $S_0$ | $P =?[Fs = 1]$ | 0.99400 |
| 4 | $S_1$ | $P =?[Fs = 3 \wedge d = 1]$ | 0.01264 |
| 5 | $S_1$ | $P =?[Fs = 3 \wedge d = 2]$ | 0.98736 |
| 6 | $S_1$ | $P =?[Fs = 2]$ | 0.98740 |
| 7 | $S_2$ | $P =?[Fs = 3 \wedge d = 1]$ | 3.795E-5 |
| 8 | $S_2$ | $P =?[Fs = 3 \wedge d = 2]$ | 0.99996 |

Table. 5 shows the probabilities for different sample volume. The row 1 denotes the probability of the system fails, which contains the verification results and simulation results under different samples. The simulation method is adopted to explain that the verification result is consistent with real-life

conditions. The simulated samples represent distinct states, and the flows of several transitional paths represent the choices made by the agent. In this experiment, the number of samples used in the simulation procedure ranged from 500 to 10000. The findings show that as the number of simulation samples grows, the experimental result approaches the verification result.

**Table 5** Simulation results for the agent control system

| Room | Number of Simulation Samples | | | | | Verify Prob |
|---|---|---|---|---|---|---|
| | 500 | 1000 | 3000 | 5000 | 10000 | |
| 1 | 0.01400 | 0.01800 | 0.01767 | 0.01820 | **0.01860** | 0.01856 |
| 2 | 0.98600 | 0.98200 | 0.98233 | 0.98180 | **0.98140** | 0.98144 |

It is worth noting that if the probability and cost of a plan change, the final results may alter. As discussed in this paper, the Policy Generation Algorithm (PGA) determines the execution sequence of all alternative plans based on the probabilities and costs of all alternative plans that achieve a transition between two states. As a result, if the probability and cost of a plan change, the PGA algorithm's final execution sequence may change. Furthermore, a change in probability can affect the reliability of the execution state transition.

## 5.4 Discussion

In the case, we demonstrate the higher reliability of control systems with multiple executable plans using probabilistic model checking techniques. To better illustrate, we compare it to a single-plan mobile agent control system. Simplify the previous case of state transfer by selecting only the plan with highest probability of success among the executable plan. As mentioned in Fig. 9, each state transfer has only one executable plan, and the probability of the plan being successfully performed is the likelihood of the state completing the transfer. The probability of effectively transferring state $S_0$ to $S_1$ is 0.9, 0.93 for successfully transferring state $S_1$ to $S_2$, and 0.99 for successfully transferring state $S_2$ to $S_3$.

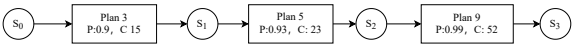


**Fig. 9** A case of state transition of mobile agent control system with a single plan.

The DTMC model of the control system with a single plan is,
*dtmc*
*module agent_movement*
$s : [0..3]$ *init* 0;
$d : [0..2]$ *init* 0;
$[\,]s = 0 \rightarrow 0.9 : (s' = 1) + 0.1 : (s' = 3) \wedge (d' = 1);$
$[\,]s = 1 \rightarrow 0.93 : (s' = 2) + 0.07 : (s' = 3) \wedge (d' = 1);$
$[\,]s = 2 \rightarrow 0.99 : (s' = 3) \wedge (d' = 2) + 0.01 : (s' = 3) \wedge (d' = 1);$

$[\,]s = 3 \rightarrow (s' = 3);$

*end module*

The DTMC model represents that the whole model has 4 states with two target values. In the initial state, there is 0.9 possibility of entering state $S_1$ and 0.1 possibility of system failure. When in state $S_1$, there is 0.93 probability of entering state $S_2$ and 0.07 probability of system failure. When in state $S_2$, there is 0.99 possibility of reaching the target state $S_3$ and a 0.01 possibility of system failure. The required properties and verification results in PRISM are shown in Tab. 6.

**Table 6** Verification results in the PRISM

| No | Init State | Properties | Results |
|---|---|---|---|
| 1 | $S_0$ | $P=?[Fs=3 \wedge d=1]$ | 0.17137 |
| 2 | $S_0$ | $P=?[Fs=3 \wedge d=2]$ | 0.82863 |
| 3 | $S_0$ | $P=?[Fs=1]$ | 0.90000 |
| 4 | $S_1$ | $P=?[Fs=3 \wedge d=1]$ | 0.07930 |
| 5 | $S_1$ | $P=?[Fs=3 \wedge d=2]$ | 0.92070 |
| 6 | $S_1$ | $P=?[Fs=2]$ | 0.93000 |
| 7 | $S_2$ | $P=?[Fs=3 \wedge d=1]$ | 0.01000 |
| 8 | $S_2$ | $P=?[Fs=3 \wedge d=2]$ | 0.99000 |

Figure. 10 shows the comparison of the results of the two control systems. MP represents the result of the control system with multiple executable plans and SP is the value of the control system with a single plan. Obviously, the control system with multiple executable plans has a higher probability of reaching the target state than a control system with a single plan and a lower probability of failure than a control system with a single plan.

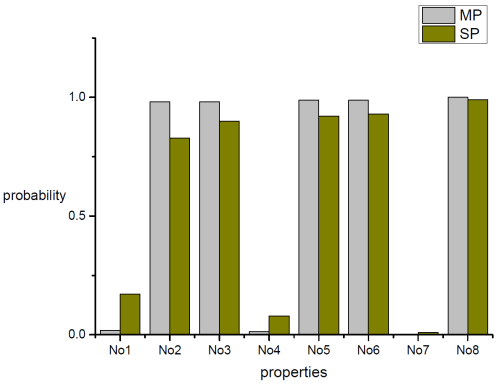


**Fig. 10** Diagram of comparison results.

On the other hand, plans incur costs once they are implemented. Therefore the expected cost value between states of a control system with a single plan is the cost of executing the current plan. The ECV of the whole system

is $\sum_{SP} = ECV_{S_0 \to S_1} + ECV_{S_1 \to S_2} + ECV_{S_2 \to S_3} = 90$. In the analysis of the control system with multiple executable plans, the total ECV generated by the policy chosen through the proposed PGA is $\sum_{MP} = ECV_{S_0 \to S_1} + ECV_{S_1 \to S_2} + ECV_{S_2 \to S_3} = 77.59$. It can be observed that the ECV of the control system with multiple alternative plans is lower than that of the control system with a single plan. Consequently, the scheme is more likely to be used in the development of mobile agent.

It is worth noting that the complexity of the PGA algorithm depends on the number of plans. First, a permutation is performed based on the input plans and perms function (a permutation generating function in Matlab), listing all possible plan execution sequences (i.e., policies). The number of statements executing the permutations is $n!$. Next, the expected cost value of all possible policies is calculated according to (3), and the number of statements executing the calculation process is $n!$. Finally, the results of the policies are sorted in ascending order according to the expected cost values based on the Quicksort algorithm, and the number of statements executing the sorting process is $nlog_2^n$. Thus, the complexity of the PGA algorithm is $O(n! + nlog_2^n)$.

# 6 Conclusion

One of the primary tasks of a mobile agent control system is to guarantee that the agent can safely move from one state (or location) to another until a task is completed. Moreover, each state transfer is achieved by a formulated behaviour plan. However, in an uncertain environment, the plan may fail due to some unexpected events. A control system with a single plan would immediately go to a failure or downtime state. In this paper, we proposed to add alternative plans to the design of mobile agent control systems to improve the reliability of agent in uncertain environments. We first introduced the concept of the mobile agent control system with multiple alternative plans, and defined two relevant parameters, i.e., the success rate of the plan and the cost of execution. Then, we proposed a PGA algorithm for selecting the most appropriate plan execution sequence. In order to analyse the reliability of the mobile agent control system with multiple alternatives, we also modelled the state transfer process into a DTMC model. The specified properties were formalized as PCTL formulae. Benefiting from the state-of-the-art probabilistic model checker PRISM, we have verified the model. A case study was given to show the application of the proposed methodological framework. Moreover, the control system equipped with single plan was compared with the control system equipped with multiple alternative plans. The results showed that the control system equipped with multiple alternative plans are more reliable than those with a single plan and have lower expected costs.

There are some aspects that can be explore further in the future. First, in this paper, we analysed the work of a single agent control system in an uncertain environment. In practice, a control system may need to control the behaviour of multiple agents at the same time and the interaction between

them also needs to be considered. Therefore, extending the proposed approach for dealing with multiple mobile agent system could be part of our future research. Second, this paper abstracted each implementation of the transfer between two states into a behaviour plan and did not analyse the specific plan contents. Each plan needs to be formulated for a specific case, and the independence between plans has to be ensured. Consequently, case-specific plan formulation also needs to be further explored.

**Acknowledgment** The authors thank the anonymous reviewer for the careful look and several valuable suggestions that helped us improve the presentation.

**Author Contributions** The conceptualization, writing and formal analysis was done by WX. The methodology and improving the language were carried out by LJ and WKM. XY participated throughout the preparation of the paper. All authors read and reviewed the version.

**Funding** This work was supported by the National Natural Science Foundation of China (No. 61976130, 62206227), the Natural Science Foundation of SiChuan (No. 2022NSFSC0464), and the Chengdu International Science Cooperation Project, China under Grant 2020-GH02-00064-HZ.

**Data Availability** Data sharing not applicable to this article as no datasets were generated or analysed during the current study.

## Declarations

**Conflict of Interest** The authors have no conflicts of interest and the paper is not been submitted to any other Journals.

**Ethical approval** This article does not contain any studies with human participants or animals performed by any authors.

## References

[1] Zhu H, Almukdad A, Iryo-Asano M, et al (2021) A novel agent-based framework for evaluating pedestrian safety at unsignalized mid-block crosswalks. Accident Analysis & Prevention, 159: 106288. doi: https://doi.org/https://doi.org/10.1016/j.aap.2021.106288

[2] Karthik S, Karthick M, Karthikeyan N, et al (2022) A multi-Mobile Agent and optimal itinerary planning-based data aggregation in Wireless Sensor Networks. Computer Communications, 184: 24-35. doi: https://doi.org/https://doi.org/10.1016/j.comcom.2021.11.019

[3] Kloock M and Alrifaee B (2023) Coordinated Cooperative Distributed Decision-Making Using Synchronization of Local Plans. IEEE Transactions

on Intelligent Vehicles.

[4] Zhou L, Tokekar P (2021) Multi-robot coordination and planning in uncertain and adversarial environments. Current Robotics Reports, 2(2): 147-157.

[5] Zhao X, Robu V, Flynn D, et al (2019) Probabilistic model checking of robots deployed in extreme environments. in: Proceedings of the AAAI Conference on Artificial Intelligence. vol 33, no 01, pp 8066-8074.

[6] Raeisi D, Jafarzadeh Ghoushchi S (2022) A robust fuzzy multi-objective location-routing problem for hazardous waste under uncertain conditions. Appl Intell : 1-21.

[7] Zhang P, Li T, Yuan Z, et al (2022) Heterogeneous feature selection based on neighborhood combination entropy[J]. IEEE T Neur Net Lear: 1-14.

[8] Zhang P, Li T, Yuan Z, et al (2023) A possibilistic information fusion-based unsupervised feature selection method using information quality measures. IEEE T Fuzzy Syst: 1-14.

[9] Zojaji Z, Ladani B T, Khalilian A (2016) Automated program repair using genetic programming and model checking. Appl Intell 45(4): 1066-1088.

[10] Donnarumma C, Fara P, Serra G, et al (2019) EN-50128 certification-oriented design of a safety-critical hard real-time kernel. in: 2019 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). IEEE: 314-317.

[11] Kwiatkowska M, Norman G, Parker D (2011) PRISM 4.0: Verification of probabilistic real-time systems. in: International conference on computer aided verification. Springer, Berlin, Heidelberg: 585-591.

[12] Zhang H, Lin W, Chen A (2018) Path planning for the mobile robot: A review. Symmetry 10(10): 450. doi: https://doi.org/https://doi.org/10.3390/sym10100450

[13] Lamini C, Benhlima S, Elbekri A (2018) Genetic algorithm based approach for autonomous mobile robot path planning. Procedia Computer Science 127: 180-189. doi: https://doi.org/https://doi.org/10.1016/j.procs.2018.01.113

[14] Wang B, Liu Z, Li Q, et al (2020) Mobile robot path planning in dynamic environments through globally guided reinforcement learning. IEEE Robot Autom Let 5(4): 6932-6939. doi: https://doi.org/10.1109/LRA.2020.3026638

[15] Song Q, Li S, Yang J, et al (2021) Intelligent Optimization Algorithm-Based Path Planning for a Mobile Robot. Comput Intel Neurosc: 1-17.

[16] Heintzman L, Williams R K (2020) Multi-agent intermittent interaction planning via sequential greedy selections over position samples. IEEE Robot Autom Let 6(2): 534-541. doi: https://doi.org/10.1109/LRA.2020.3047788

[17] Toumieh C and Lambert A (2022) Decentralized multi-agent planning using model predictive control and time-aware safe corridors. IEEE Robot Autom Let 7(4): 11110-11117. doi: https://doi.org/10.1109/LRA.2022.3196777

[18] Chen Y, Rosolia U, Ubellacker W, et al (2022) Interactive multi-modal motion planning with branch model predictive control. IEEE Robot Autom Let 7(2): 5365-5372. doi: https://doi.org/10.1109/LRA.2022.3156648

[19] Chen Y T, Zhang Z Y, and Huang J (2020) Dynamic task priority planning for Null-Space behavioral control of multi-agent systems. IEEE Access 8:149643-149651. doi: https://doi.org/https://doi.org/10.1109/ACCESS.2020.3016347

[20] Tahir H, Syed M N, and Baroudi U (2020) Heuristic approach for real-time multi-agent trajectory planning under uncertainty. IEEE Access, 8: 3812-3826. doi: https://doi.org/10.1109/ACCESS.2019.2962785

[21] Rens G and Moodley D A (2017) Hybrid POMDP-BDI agent architecture with online stochastic planning and plan caching. Cognitive Systems Research, 43: 1-20. doi: https://doi.org/https://doi.org/10.1016/j.cogsys.2016.12.002

[22] Torreno A, Onaindia E, Komenda A, et al (2018) Cooperative multi-agent planning: A survey. ACM Computing Surveys, 50(6): 1-34. doi: https://doi.org/https://doi.org/10.1145/3128584

[23] Schwung M and Lunze J (2022) Cooperative event-based control of mobile objects over an unreliable communication network. AT-Automatisierungstechnik, 70(2): 105-118.

[24] Wang X, Liu J, Nugent C, et al (2023) Mobile agent path planning under uncertain environment using reinforcement learning and probabilistic model checking. Knowl-Based Syst, 264:110355. doi: https://doi.org/https://doi.org/10.1016/j.knosys.2023.110355

[25] Kim K O, Zuo M J (2018) Optimal allocation of reliability improvement target based on the failure risk and improvement cost. Reliab Eng Syst Safe 180: 104-110. doi: https://doi.org/https://doi.org/10.1016/j.ress.2018.

06.024

[26] Rungskunroch P, Jack A, Kaewunruen S (2021) Benchmarking on railway safety performance using Bayesian inference, decision tree and petri-net techniques based on long-term accidental data sets. Reliab Eng Syst Safe 213: 107684. doi: https://doi.org/https://doi.org/10.1016/j.ress.2021.107684

[27] Musharraf M, Smith J, Khan F, et al (2020) Identifying route selection strategies in offshore emergency situations using decision trees. Reliab Eng Syst Safe 194: 106179. doi: https://doi.org/https://doi.org/10.1016/j.ress.2018.06.007

[28] Wu D, Liu J, Wang H, et al (2021) A CPN-based approach for studying impacts of communication delays on safety and availability of safety-critical distributed networked control systems. IEEE T Ind Inform 18(5): 3033-3042. doi: https://doi.org/10.1109/TII.2021.3109436

[29] Farhadi M, Shahrokhi M, Rahmati S H A (2022) Developing a supplier selection model based on Markov chain and probability tree for a k-out-of-N system with different quality of spare parts. Reliab Eng Syst Safe 222: 108387. doi: https://doi.org/https://doi.org/10.1016/j.ress.2022.108387

[30] Baladeh A E and Taghipour S (2022) Reliability optimization of dynamic k-out-of-n systems with competing failure modes. Reliab Eng Syst Safe 227: 108734. doi: https://doi.org/https://doi.org/10.1016/j.ress.2022.108734

[31] Xiahou T, Zheng Y X, Liu Y, et al (2023) Reliability modeling of modular k-out-of-n systems with functional dependency: A case study of radar transmitter systems.Reliab Eng Syst Safe 233: 109120.doi: https://doi.org/https://doi.org/10.1016/j.ress.2023.109120

[32] Zheng X, Bolton M L, Daly C, et al (2020) The development of a next-generation human reliability analysis: Systems analysis for formal pharmaceutical human reliability (SAFPH). Reliab Eng Syst Safe 202: 106927. doi: https://doi.org/https://doi.org/10.1016/j.ress.2020.106927

[33] Bolton M L, Zheng X, Kang E (2021) A formal method for including the probability of erroneous human task behavior in system analyses. Reliab Eng Syst Safe 213: 107764. doi: https://doi.org/https://doi.org/10.1016/j.ress.2021.107764

[34] Gao H, Mao S, Huang W, et al (2018) Applying probabilistic model checking to financial production risk evaluation and control: A case study of Alibabas Yue Bao. IEEE Transactions on Computational Social Systems, 2018, 5(3): 785-795. doi: https://doi.org/10.1109/TCSS.2018.2865217

[35] Wan W, Bentahar J, Hamza A B (2013) Model checking epistemicCprobabilistic logic using probabilistic interpreted systems. Knowl-Based Syst 50: 279-295. doi: https://doi.org/https://doi.org/10.1016/j.knosys.2013.06.017

[36] Feng L, Wiltsche C, Humphrey L, et al (2015) Controller synthesis for autonomous systems interacting with human operators. in: Proceedings of the acm/ieee sixth international conference on cyber-physical systems: 70-79.

[37] Zhu Y, Xue X, Zhang K, et al (2016) Applying probabilistic model checking to express delivery location selection and optimization. in:IEEE 13th International Conference on e-Business Engineering (ICEBE). IEEE: 32-39.

[38] Wang X, Liu J, Moore S J, et al (2023) A behavioural hierarchical analysis framework in a smart home: Integrating HMM and probabilistic model checking. Inform Fusion 95: 275-292.

[39] Ethier S N, Kurtz T G (2009) Markov processes: characterization and convergence. John Wiley & Sons.

[40] Sultan K, Bentahar J, El-Menshawy M (2014) Model checking probabilistic social commitments for intelligent agent communication. Appl Soft Comput 22: 397-409. doi: https://doi.org/https://doi.org/10.1016/j.asoc.2014.04.014

[41] Ciesinski F, GröBer M (2004) On probabilistic computation tree logic. Validation of Stochastic Systems. Springer, Berlin, Heidelberg: 147-188.

[42] Guo Y, Sheng S, Phillips C, et al (2020) A methodology for reliability assessment and prognosis of bearing axial cracking in wind turbine gearboxes. Renew Sust Energ Rev 127: 109888. doi: https://doi.org/https://doi.org/10.1016/j.rser.2020.109888

[43] Pietrantuono R, Popov P, Russo S (2020) Reliability assessment of service-based software under operational profile uncertainty. Reliab Eng Syst Safe 204: 107193. doi: https://doi.org/https://doi.org/10.1016/j.ress.2020.107193

[44] Li M, Li B, Huai J P (2012) Reliability-aware automatic composition approach for web services. Science China Information Sciences 55(4): 921-937.

[45] Kress-Gazit H, Lahijanian M, Raman V (2018) Synthesis for robots: Guarantees and feedback for robot behavior. Annual Review of Control, Robotics, and Autonomous Systems 1: 211-236. doi: https://doi.org/https:

//doi.org/10.1146/annurev-control-060117-104838

[46] Luckcuck M, Farrell M, Dennis L A, et al (2019) Formal specification and verification of autonomous robotic systems: A survey. ACM Computing Surveys (CSUR) 52(5): 1-41. doi: https://doi.org/https://doi.org/10.1145/3342355

[47] Saraswat A, Abhishek K, Ghalib M R, et al (2022) Towards Energy Efficient Approx Cache-coherence Protocol Verified using Model Checker. Comput Electr Eng 97: 107482. doi: https://doi.org/https://doi.org/10.1016/j.compeleceng.2021.107482

[48] Kwiatkowska M, Norman G, Parker D (2002) PRISM: Probabilistic symbolic model checker. in: International Conference on Modelling Techniques and Tools for Computer Performance Evaluation. Springer, Berlin, Heidelberg: 200-204.

[49] Desharnais J, Panangaden P (2003) Continuous stochastic logic characterizes bisimulation of continuous-time Markov processes. The Journal of Logic and Algebraic Programming 56(1-2): 99-115. doi: https://doi.org/https://doi.org/10.1016/S1567-8326(02)00068-1

[50] Ng R, Subrahmanian V S (1992) Probabilistic logic programming. Inform Comput 101(2): 150-201. doi: https://doi.org/https://doi.org/10.1016/0890-5401(92)90061-J

[51] Rebello S, Yu H, Ma L (2018) An integrated approach for system functional reliability assessment using Dynamic Bayesian Network and Hidden Markov Model. Reliab Eng Syst Safe 180: 124-135. doi: https://doi.org/https://doi.org/10.1016/j.ress.2018.07.002