A Simple Modal Logic Approach to Decision Process

JULIO CLEMPNER
National Polytechnic Institute
Center for Applied Science and
High Technology Research
Center for Computing Research
Av. Juan de Dios Batiz s/n, Edificio CIC
Col. Nueva Industrial Vallejo, 07738
Mexico City
MEXICO

JESUS MEDEL
National Polytechnic Institute
Center for Applied Science and
High Technology Research
Center for Computing Research
Av. Juan de Dios Batiz s/n, Edificio CIC
Col. Nueva Industrial Vallejo, 07738
Mexico City
MEXICO

Abstract: In this paper we introduce a new modeling paradigm for developing decision process representation associating to any Decision Process Petri net (DPPN) a Kripke structure (KS). The principal characteristics of this model is its ability to represent and analyze the shortest-path properties of a decision process. In this sense, we use a Lypunov-like function as a state-value function for path planning, obtaining as a result new characterizations for final decision points. We show that the dynamics of the DPPN can be captured by a KS and, some dynamic properties of a DPPN can be stated in temporal logics. The temporal logic is constructed according to the Lypunov-like function syntax and semantics. Moreover, we consider some results and discuss possible directions for further research.

Key–Words: Modal Logic, Decision Process, Lyapunov Theory

1 Introduction

The analysis, design and development of large systems is a very difficult task. Especially, systems consisting of concurrent processes involving continuous-time Markov process as well as it associated stochastic process. In this sense, the need to avoid design errors becomes more and more important. The fact is that the size of the systems is rapidly growing. If we suppose that the number of errors in a system is proportional to its size, it becomes clear that the larger the system is, more problematic will be its design, and less will be the probability that it will correctly work.

For this reason, the application of formal methods in the design of decision process systems is more and more frequently discussed. There are some approaches that are independent of the kind of system that is to be verified. However, in most cases, the kind of system determines the kinds of properties to be verified, and this in turn makes one or another verification formalism more or less suited. We will consider different ways how formal methods can be applied to guide the design of complex systems. In particular, we distinguish between two main approaches: Decision Process Petri Nets and Temporal logics.

The Decision Process Petri Nets (DPPN) is a powerful modeling framework, which combines advantages of the place-transitions Petri Nets (PN) with the expressive power of the decision process ([5], [6]). It extends the place-transitions Petri net theoretic approach including the Markov decision processes, using a utility function as a tool for trajectory planning. On the one hand, place-transitions Petri nets are used for process representation, taking advantage of the well-known properties of this approach namely, formal semantic and graphical display, giving a specific and unambiguous description of the behavior of the process. On the other hand, Markov decision processes have become a standard model for decision theoretic planning problems, having as key drawbacks the exponential nature of the dynamic policy construction algorithms. Although, both perspectives are integrated in a DPPN they work in different execution levels. That is, the operation of the place-transitions Petri net is not modified and the utility function is used exclusively for establishing a trajectory tracking in the place-transitions Petri net. The DPPN is able to express the liveness, safety, persistence and fairness properties of the PN.

Temporal logics are usually defined in terms of Kripke structures ($\mathcal{KS}$). A Kripke structure can be represented by a bipartite graph having two classes of nodes, and no edges can relate nodes from the same set. The reachable states of the system are represented by places nodes and the action of the system as transitions nodes. It also contains a labeling of the states of the system with properties that hold in each state. Therefore, the dynamics of a discrete system can be captured by a Kripke structure and, some dynamic properties of a discrete system can be stated in temporal logics. In this sense, [**?**], [8] have investigated the temporal logics language corresponding to fintestateautomaton and place-transitions Petri Nets.

Kripke structures are closely related to Decision Process Petri nets, however there are fundamental differences. A DPPN represents systems that understand tokens as inputs and respond to these by performing actions in form of state transitions leaded by a utility function. Nevertheless, a Kripke structure is obtained just for a system's description, it exemplifies all the possible situations that can occur in performing the actions of a system. This means that a Kripke structure has no possibility to give reasons for explaining why the system is in a specific state, or it moves to a different state. In addition, for complicated systems specifications expressing the specification directly as a DPPN can be too complicated. This is one of the reasons modal logics are more widely used as a specification formalisms than Petri nets.

The motivation for this work is that the existing logics for decision process do not have the right expressive power to deal with decision making. On the other side, decision making play a fundamental role when modeling applications related with game theory.

The aim of this paper is to associate to any Decision Process Petri net (DPPN) a Kripke structure ($\mathcal{KS}$). We extended the logic Kripke structure in order to specify steady-state, transient and path-based measures extended with a Lyapunov-like function, which is a state-value function.

This approach allows for the specification of new measures that have not yet been addressed in the performability literature. In this sense, a Lyapunov-like functions can be used as state-value functions and optimal cost-to-target functions. As a result of calculating a Lyapunov-like function, a discrete vector field can be built for tracking the actions over the net. Each applied optimal action produces a monotonic progress (of the optimal cost-to-target value) toward an equilibrium point. In this sense, if the function decreases with each action taken, then it approaches an infimum/minimum (converges asymptotically or reaches a constant).

The principal characteristics of this model is its ability to represent and analyze the shortest-path properties of a decision process. In this sense, we use a Lypunov-like function as a state-value function for path planning, obtaining as a result new characterizations for final decision points. We show that the dynamics of the DPPN can be captured by a $\mathcal{KS}$ and, some dynamic properties of a DPPN can be stated in temporal logics. The temporal logic is constructed according to the Lypunov-like function syntax and semantics. Moreover, we consider some results and discuss possible directions for further research.

In this paper we introduce a new modeling paradigm for developing decision process representation. It extends the Kripke structure theoretic approach. $\mathcal{KS}$s are used for process representation taking advantage of the formal syntax and semantics, and the graphical display. Markov decision process is utilized as a tool for path planning via a Lyapunov-like function. As result, we obtain new characterizations for final decision points (optimum point). We propose a temporal logic called Decision Process Logic (DPL) constructed according to a state-value function syntax and semantics for $\mathcal{DK}$ allowing the specification of new properties that have not yet been addressed in the literature.

The rest of the paper is organized as follows. The next section presents the formulation of the problem including the decision Kripke structure and the proposed Decision Process Logic. Finally, some concluding remarks and future work are provided in Section 3.

2 Formulation

The aim of this section is to associate to any Decision Process Petri net ([5], [6]) a Kripke structure ($\mathcal{KS}$).

Definition 1 *Let $DPPN = \{P, Q, F, W, M_0, \pi, U\}$ be a Decision Process Petri net. The Kripke structure of a DPPN is a tuple $\mathcal{KS} = (S, s_0, R, M_0, L, \pi, U)$ where:*

- *S is a finite set of states,*
- *s_0 is the initial state,*
- *R is the set of triples (s, r, s') such that the transition $r \in R$ is enabled at state s and s' is is obtained by executing r at state s,*
- *$L : S \to 2^{\Gamma}$ is a function which associates with each state a set of atomic propositions Γ.*

The Kripke structure $\mathcal{KS} = (S, s_0, R, M_0, L, \pi, U)$ satisfies recursively the following properties:

1. $M_0 \in S$

2. If $M \in S$ and $M\,[q\rangle\, M'$ such that $M' = U(M)$ then $M' \in S$ and $(M, M') \in R$

3. S and R have no other elements

The executions of the $\mathcal{KS}$ are infinite sequences $M_0 M_1 M_2 ...$ of states in S leaded by a state-value function U where M_0 is the initial state and $(M_i, M_{i+1}) \in R$ holds for $i \geq 0$.

We have clear that the existing approaches for decision process and Kripke structure have characteristics in common. The goal now is to integrate these two approaches even more introducing a logic (linear temporal logic) to reason about decision process. Our approach includes a Lyapunov (steady-state) operator $\mathcal{S}$ for a path ([1], [2], [3], [4], [7]). The steady-state operator is related with the trajectory-tracking value of residing in a particular set of states.

Formulas are built from the set of atomic proposition Γ and are closed under the application of Boolean connectives, the unary temporal connective $\mathcal{X}$ (next), the binary temporal connective $\mathcal{U}$ (until) and, the Lyapunov operator $\mathcal{S}$ satisfying the following properties:

1. the atomic proposition $a \in \Gamma$ is a formula,

2. if φ and ψ are formulas then $(\neg\varphi)$ and $(\varphi \wedge \psi)$ are formulas,

3. if φ is a formula then $\mathcal{S}\varphi$ is a formula,

4. if φ and ψ are formulas then $\mathcal{X}\varphi$ and $\psi\mathcal{U}\varphi$ are formulas.

Boolean connectives are derived in the usual way, i.e. $false = \neg true$, $\varphi \vee \psi = \neg(\neg\varphi \wedge \neg\psi)$, and $\varphi \Longrightarrow \psi = \neg\varphi \vee \psi$.

Formulas keep the usual interpretation, with the added understanding that: the formula $\mathcal{S}\varphi$ asserts that the state-value function U for the paths that satisfy φ approaches to an infimum or attains the minimum, i.e. $U(s^{\triangle}) = 0$ or $U(s^{\triangle}) = C$.

The logic is interpreted under *computations.* A computation is a finite or infinite sequence $\kappa = K(0)K(1)...$ of sets of atomic proposition of Γ. Intuitively, $K(i)$ is the set of propositions that hold in the computation after i steps. A formula ψ is true in a computation κ and a point i, i.e. $\kappa, i \models \psi$, under the following conditions:

1. $\kappa, i \models a$ for $a \in \Gamma$ iff $a \in K(i)$.

2. $\kappa, i \models \varphi \wedge \psi$ iff $\kappa, i \models \varphi$ and $\kappa, i \models \psi$

3. $\kappa, i \models \neg\psi$ iff not $\kappa, i \models \psi$

4. $\kappa, i \models \mathcal{X}\psi$ iff there exists a point $i+1$ in the computation and $\kappa, i+1 \models \psi$

5. $\kappa, i \models \varphi\mathcal{U}\psi$ iff for some $j \geq i$, we have $\kappa, j \models \psi$ and for all l, $i \leq l < j$, we have $\kappa, l \models \varphi$

6. $\kappa, i \models \mathcal{S}\psi$ iff there exists a point $j \geq i$ in the computation such that $U(j) \to 0$ (approaches to an infimum) or $U(j) = C$ (the minimum is attained).

Interpretation. A $\mathcal{KS}$ represents a set of allowable action sequence. In particular, represents a set of action sequence that starts in an initial state and follows the transitions conditions obeying a state-value function U. At run-time, a $\mathcal{KS}$ can interpreted in the following manner. At every discrete time step, a decision maker is at one of the states, and it selects the next action to take. ....?

A *path* is an ordered sequence $s_i, s_{i+1}, ...$ of states generated by U over the Kripke structure $\mathcal{KS}$ such that $s_{i+1} = U(s_i)$ and $(s_i, s_{i+1}) \in R$ holds for $i \geq 0$.

Property 2 *The relation* $\leq_U \subseteq S \times S$ *defined by* $s_i \leq_U s_{i+1} \Longrightarrow U(s_{i+1}) \leq U(s_i)$*, is an ordering on* $\mathcal{KS}$*, i.e. its reflexive, transitive and antisymmetric.*

Proof. If we consider $\equiv_U$ be the equivalence relation on S induced by U as $\forall s, t \in S : s \equiv_U t \Longleftrightarrow U(s) = U(t)$. Then the equivalence class $(S/\equiv_U) = \{\Lambda(s) | s \in S\}$ is a poset isomorphic to a subset of $\mathbb{R}$. Thus, $S/\equiv_U$ is linearly ordered and, consequently, it is a lattice. The structure $S/\equiv_U$ is indeed trivial: all elements in S giving the same value under U are identified in this quotient set. On the other hand, let us consider the relation $\leq_U$as $\forall s, t \in S : s \leq_U t \Longleftrightarrow U(t) \leq U(s)$.This relation is reflexive and transitive, and it is antisymmetric because U is a Lyapunov-like function strictly decreasing and therefore it is one-to-one. ■

The sequence $s_i, s_{i+1}, ...$ of states of a path over the Kripke structure $\mathcal{KS}$ are functions $v : \mathbb{N} \to S$. We use v^i to denote the suffix of the path $v = s_0, s_1, ...$ starting at the index i, such that $v^i = s_i, s_{i+1}, ...$ We use $v^{(i)}$ to denote the ith state of the path v. The *set of paths* starting from a given state s over the Kripke structure $\mathcal{KS}$ is $\Upsilon(s) = \{v | v^{(0)} = s \wedge \forall i \in \mathbb{N} : (v^{(i)}, v^{(i+1)}) \in R\}$. Consequently, the set of all paths over S is $\Upsilon(S) = \bigcup_{s \in S} \Upsilon(s)$.

A path sequence $s_0, s_1, ...$ of states over the Kripke structure $\mathcal{KS}$ is called *fair* if for every set of states $F_k \in F$ $(0 \leq k < |F|)$ there are infinitely many states $v^{(i)}$ in the path that belong to F_k. The *set of fairness paths* starting from a given state s over the Kripke structure $\mathcal{KS}$ is $\Phi(s) = \{v \in \Upsilon(s) | \forall F_k \in F, \forall i \in \mathbb{N}\, \exists j \in \mathbb{N} : v^{(i+j)} \in F_k\}$. Consequently, the set of all fair paths over S

is $\Phi(S) = \bigcup_{s\in S} \Phi(s)$. Clearly, we have that $\Phi(s) \subseteq \Upsilon(s)$. For $F = \varnothing$ the requirement that every set F_k is visited infinitely often is vacuously true, and thus any path is fair. From now on, we will assume all formulas to be interpreted over fair paths. Therefore, the Kripke structure of a $DPPN$ is a tuple $\mathcal{KS} = (S, s_0, R, M_0, L, F, \pi, U)$ where $F \subseteq 2^S$ is a set of fairness constraints.

A *run* ρ is a sequence $s_0, s_1, ...$ of states generated by U over the Kripke structure $\mathcal{KS}$ such $(s_i, s_{i+1}) \in R$ holds for $i \geq 0$. The *language* of $\mathcal{KS}$ is defined as $\mathcal{L}(\mathcal{KS}) = \{v \in \Phi \mid v \text{ has a run } \rho{=}s_0, ... \text{ in } S\}$.

Remark 3 *In Petri nets the states of the system are the markings. The atomic propositions are predicates on the possible markings of the net (having usually one atomic proposition per place). Predicates are of the form $\varphi(p, m)$ where p is a place of the net and $m \in \mathbb{N}_+$, and it is read 'the number of token of p is greater then or equal to c', holding if at marking M we have that $M(p) \geq m$. The markings satisfying the atomic proposition p are those that put a token in p. In a Petri nets a computation is a sequence of sets of places. As a consequence, a computation is a sequence of markings. It is important to note that computations are the sequences of markings obtained from the runs ρ leaded by the state-value function U of the Petri net by removing the intermediate transitions.*

Definition 4 *A Petri net satisfies a formula φ if there is a run ρ leaded by the state-value function U that satisfies φ.*

Remark 5 *Intuitively, a Lyapunov-like function can be considered as state-value function. In our case, an optimal discrete problem, the trajectory-tracking values are calculated using a discrete Lyapunov-like function. Every time a discrete vector field of possible transitions is calculated over the decision process. Each applied optimal transition (selected via some 'criterion', e.g.* $\min(\cdot)$ *) decreases the optimal value, ensuring that the optimal course of action is followed and establishing a preference relation. In this sense, the criterion change the asymptotic behavior of the Lyapunov-like function by an optimal trajectory-tracking value. It is important to note, that the process finished when the equilibrium point is reached. This point determines a significant difference with the Bellman's equation.*

Then, we can conclude the following theorem of existence.

Theorem 6 *Let $\mathcal{KS} = (S, s_0, R, M_0, L, F, \pi, U)$ be a Kripke structure. Then, if there is a run $\rho = s_0, s_1, ...s_n$ leaded by the state-value function U that satisfies φ then the $\mathcal{KS}$ is bounded by the state s_n of the system.*

Proof. Let us suppose that the $\mathcal{KS}$ is not finite. Then s_n is never reached. Therefore, it is possible to evolve in time n and to reduce the trajectory function value over s_n. However, the Lyapunov-like trajectory function converges to zero when $n \to \infty$ (or reached a minimum) i.e., $U_n = 0$ or $U_n = C$. ■

Property 7 *Every Kripke structure $\mathcal{KS} = (S, s_0, R, M_0, L, F, \pi, U)$ is bounded by a state s_n.*

Proof. Straightforward by the previous theorem. ■

Definition 8 *An equilibrium (steady-state) point with respect a Kripke structure $\mathcal{KS} = (S, s_0, R, M_0, L, F, \pi, U)$ is the last state of the net.*

Remark 9 *An equilibrium point with respect a Decision Process Petri net $DPPN = \{P, Q, F, W, M_0, \pi, U\}$ is a place $p^* \in P$ such that $M_l(p^*) = C < \infty$, $\forall l \geq k$, and p^* is a sink.*

Definition 10 *A final decision state $s_f \in S$ with respect a Kripke structure $\mathcal{KS} = (S, s_0, R, M_0, L, F, \pi, U)$ is a state $s \in S$ where the infimum is asymptotically approached (or the minimum is attained), i.e. $U(s) = 0$ or $U(s) = C$.*

Definition 11 *An optimum state $s^{\triangle} \in S$ with respect a Kripke structure $\mathcal{KS} = (S, s_0, R, M_0, L, F, \pi, U)$ is a final decision state $s_f \in S$ where the best choice is selected 'according to some criteria'.*

Corollary 12 *Every Kripke structure $\mathcal{KS} = (S, s_0, R, M_0, L, F, \pi, U)$ has a final decision state.*

It easy to show the correspondence between a steady-state, final decision state and optimum state ([5], [6]).

$(s, s') \in R$ is a total transition relation that means that the state s' can be reached from the state s, i.e., it is a possible successor state of s. There may be more than one successor state, and usually conditions over the transitions (probabilities, etc.) are given to select a particular next state.

For the semantics of our logics, we must reason about successor and predecessor states of a particular state. These are formally defined as given below.

For any $s \in S$ let the *successors* of s be: $s' \in suc(s)$ iff $s \neq s'$, $(s, s') \in R$ and $\forall s_* : (s, s_*) \in R \vee (s_*, s') \in R \implies (s_* = s') \vee (s_* = s)$. For any $s \in S$ let the *predecessor* of s be: $s' \in pre(s)$ iff

$s' \neq s$, $(s', s) \in R$ and $\forall s_* : (s', s_*) \in R \vee (s_*, s) \in R \Longrightarrow (s_* = s) \vee (s_* = s')$.

We assume that every state has an out degree in $\mathcal{KS}$, i.e. a number of immediate successors. The *maximal elements* are those with no predecessors, i.e. state with null inner degree in $\mathcal{KS}$. The *minimal elements* are those with no successors, i.e. state with null outer degree in $\mathcal{KS}$.

Let us define the *upper distance* d^+ as follows:

$$\begin{gathered} d^+(s, s') = 1 \quad \Longleftrightarrow s' \in suc(s) \\ d^+(s, s') = 1 + r \Longleftrightarrow \\ \exists s_* : d^+(s, s_*) = r \& d^+(s_*, s') = 1 \end{gathered}$$

Similarly, the *lower distance* d^- is

$$\begin{gathered} d^-(s, s') = 1 \quad \Longleftrightarrow s' \in pre(s) \\ d^-(s, s') = 1 + r \Longleftrightarrow \\ \exists s_* : d^-(s, s_*) = r \& d^-(s_*, s') = 1 \end{gathered}$$

Thus $d^+(s, s') = d^-(s', s)$.

The *upper height* of a node s is $h^+(s) = Max\{d^+(s', s) | s'$ is maximal$\}$. The *lower height* of a node s is $h^-(s) = Max\{d^-(s', s) | s'$ is minimal$\}$.

We mainly use the above definitions for the transition relations of Kripke structures to determine successor and predecessor states.

We say that a state $s' \in S$ is *reachable* from a state $s \in S$, $rs(s, s')$, if there exists a path $\upsilon \in \Phi(s)$ of states of the Kripke structure $\mathcal{KS}$ leading from s to s' by U, *i.e.*

$$\begin{gathered} rs(s, s') = \{\upsilon | \upsilon^{(0)} = s \wedge \upsilon^{(n)} = s' \wedge \forall i \in \mathbb{N} : \\ s_{i+1} = U(s_i) \wedge (\upsilon^{(i)}, \upsilon^{(i+1)}) \in R\}. \end{gathered}$$

3 Conclusions and future work

We presented an approach for decision Kripke structures. It contribute to bridging the gap between classical Kripke structures and the decision process. We have proposed Decision Process Logic for property specification of Decision Kripke structures. The expressive power and the mathematical formality of the Decision Process Logic is also useful for capturing the dynamics and describing the dynamic properties of a DPPN. We have implemented a model checking system which is optimized by use of a Lyapunov-like utility function for path tracking. In this sense, there are a number of questions relating classical model checking that may in the future be addressed satisfactorily within this framework. We are currently working in the generalization to game theory.

References

[1] A. Aziz, K. Sanwal, V. Singhal, and R. Brayton. Model checking continuous time Markov chains. *ACM Transactions on Computational Logic*, 1, 1, 162-170, 2000.

[2] C. Baier, J.-P. Katoen, and H. Hermanns. Approximate symbolic model checking of continuous-time Markov chains. *In Concurrency Theory*, LNCS 1664, 146-162, Springer-Verlag, 1999.

[3] C. Baier, B.R. Haverkort, H. Hermanns, and J.-P. Katoen. Model checking continuous-time Markov chains by transient analysis. *In Computer Aided Verification*, LNCS 1855, 358-372, Springer-Verlag, 2000.

[4] C. Baier, B. R. Haverkort, H. Hermanns, and J.-P. Katoen. On the Logical Characterization of Performability Properties. In ICALP '00: *Proceedings of the 27th International Colloquium on Automata, Languages and Programming*, 780-792, London, UK, 2000. Springer-Verlag, 2000.

[5] J. Clempner. Colored Decision Process Petri Nets: Modeling, Analysis and Stability. *International Journal of Applied Mathematics and Computer Science*, 15, 3, 405-420, 2005.

[6] J. Clempner. Towards Modeling The Shortest-Path Problem and Games with Petri Nets. *Proc. of The Doctoral Consortium at the ICATPN*, 1-12, 2006.

[7] J.-P. Katoen, M.Z. Kwiatkowska, G. Norman, and D. Parker. Faster and symbolic CTMC model checking. *In Process Algebra and Probabilistic Methods*, LNCS 2165, 23-38, Springer-Verlag, 2001.

[8] M. Vardi. An Automata-Theoretic Approach to Linear temporal Logic. *In Logics for Concurrency: Structure versus Automata*. LNCS 1043, 238–265, Springer-Verlag, 1996.