RESUMEN DE TESIS DOCTORAL

Setting Decision Process Optimization into Stochastic vs. Petri Nets Contexts

Cortando con Procesos de Decisión Estocásticos respecto al contexto de las Redes de Petri

Graduated: Julio Clempner
Center for Computing Research (CIC), National Polytechnic Institute
Av. Juan de Dios Batiz s/n, Edificio CIC, Col. Nueva Industrial Vallejo, 07738 Mexico City, Mexico
Center for Applied Science and High Technology Research (CICATA), National Polytechnic Institute
Legaria 69 Col. Irrigación, 11500 Mexico City, Mexico
e-mail:julio@k-itech.com

Graduated on: November 24, 2006

Advisor: Jesús Medel
Center for Computing Research, National Polytechnic Institute
Av. Juan de Dios Batiz s/n, Edificio CIC, Col. Nueva Industrial Vallejo, 07738 Mexico City, Mexico
e-mail:jjmedelj@cic.ipn.mx

Co-Advisor: Alin Cârsteanu
Center for Research and Advanced Studies (Cinvestav), National Polytechnic Institute
Av. IPN 2508, C.P. 07360, Col. San Pedro Zacatenco, Mexico City, Mexico
e-mail: alin@math.cinvestav.mx

Abstract

In this work we introduce a new modeling paradigm for developing decision process representation for shortest-path problem and games. Whereas in previous work, attention was restricted to tracking the net using as a utility function Bellman's equation, this work uses a Lyapunov-like function. In this sense, we are changing the traditional cost function by a trajectory-tracking function which is also an optimal cost-to-target function for tracking the net. This makes a significant difference in the conceptualization of the problem domain, allowing the replacement of the Nash equilibrium point by the Lyapunov equilibrium point in shortest-path game theory. Two different formal theoretic approaches are employed to represent the problem domain: i) Markov decision process and, ii) place-transitions Petri Nets having as a feature a Markov decision process, called Decision Process Petri nets (DPPN). The main point of this paper is its ability to represent the system-dynamic and trajectory-dynamic properties of a decision process. Within the system-dynamic properties framework we prove new notions of equilibrium and stability. In the trajectory-dynamic properties framework, we optimize the trajectory function value used for path planning via a Lyapunov-like function, obtaining as a result new characterizations for final decision points (optimum points) and stability. We show that the system-dynamic and Lyapunov trajectory-dynamic properties of equilibrium, stability and final decision points (optimum points) meet under certain restrictions. Moreover, we generalize the problem to join with game theory. We show that the Lyapunov equilibrium point coincides with the Nash equilibrium point under certain restrictions. As a consequence, all the properties of equilibrium and stability are preserved in game theory under certain restrictions. This is the most important contribution of this work. The potential of this approach remains in its formal proof simplicity for the existence of an equilibrium point. To the best of our knowledge the approach seems to be new in decision process, game theory and Petri Nets.

Keywords: shortest-path problem, shortest-path game, stability, Lyapunov, Markov decision process, Petri nets.

Resumen

En este trabajo se introduce un paradigma nuevo de modelado para representar procesos de decisión relacionados con el problema de la trayectoria más corta y teoría de juegos. Mientras que trabajos anteriores han restringido su atención a recorrer la red utilizando la ecuación de Bellman como función de utilidad, en este trabajo se utiliza una función de tipo Lyapunov. En este sentido, se está cambiando la función de costo tradicional por una función de trayectoria y costo a objetivo óptima. Esto genera una diferencia significativa en la manera que el dominio del problema es conceptuado permitiendo el cambio del punto de equilibrio de Nash por el punto de equilibrio de Lyapunov en teoría de juegos. Se utilizan dos aproximaciones teóricas diferentes para representar el dominio del problema: i) procesos de decisión de Markov, y ii) redes de Petri lugar-transición teniendo como característica un proceso de decisión de Markov. El punto principal del escenario propuesto es la habilidad de representar las propiedades de la dinámica del sistema y la dinámica de las trayectorias de un proceso de decisión. Dentro del marco de las propiedades dinámicas del sistema se muestran nuevas características de equilibrio y estabilidad. Dentro del marco de las propiedades de dinámicas por trayectoria del sistema se optimiza la función para calcular la trayectoria de planeación con una función del tipo Lyapunov, obteniendo como resultado una caracterización nueva para puntos finales de decisión (puntos óptimos) y estabilidad. Además, se muestra que las propiedades dinámicas del sistema y las propiedades dinámicas por trayectoria del sistema de equilibrio, estabilidad y puntos finales de decisión (puntos óptimos) convergen bajo ciertas restricciones. Inclusive, se generaliza el problema para desembocar en teoría de juegos. En ese contexto, se muestra que el punto de equilibrio de Lyapunov coincide con el punto de equilibrio de Nash bajo ciertas restricciones. Como consecuencia todas las propiedades de equilibrio, estabilidad y punto final de decisión persisten en teoría de juegos. Esta es la contribución más importante de este trabajo. La potencialidad de esta aproximación está en la simplicidad de la prueba formal para la existencia de un punto de equilibrio. Hasta lo que nuestro conocimiento alcanza este trabajo parece ser nuevo en procesos de decisión, teoría de juegos y redes de Petri.

Palabras clave: problemas de la trayectoria más corta, juegos con trayectoria más corta, estabilidad, Lyapunov, procesos de decisión de Harkov, redes de Petri.

1 Introduction

Markov decision processes can be used to analyze shortest-path and minimum cost-to-target problems, in which a natural form of termination ensures that expected future costs are bounded, at least under some classes of policies. The stochastic shortest-path problem ([1], [2], [5], [10], [23], [41], [44]) is a generalization of the deterministic shortest-path problem through which at each node there exists a probability distribution over all possible successor nodes. Given a starting node and a selection of distributions we wish that the path would lead to a final point with probability one, and that it would have minimum expected length. Note that the case where at each node the probability distribution of transitions associates a unit probability to exactly one successor, is precisely the case of the deterministic shortest-path problem.

In this sense, finite action-state and action-transient Markov decision processes with positive cost functions were first formulated and studied by Eaton and Zadeh [12]. They called this a *problem of pursuit*, which consists of intercepting in minimum expected time a target that moves randomly among a finite number of states. In the study, they established the idea of proper policy and supposed that at each state, except for the final state, and the set of controls is finite. Pallu de la Barriere [35] supported and improved their results. Derman [10] also extended these results under the title of *first passage problems*, observing that the finite-state Markovian decision problem is a particular case. Veinott [43] obtained similar results to those of Eaton and Zadeh [12] proving that the dynamic programming mapping is a contraction under the assumption that all stationary policies are proper (transient). Kushner [23] enhanced the results of Eaton and Zadeh [12] allowing the set of controls to be infinite at each state, and restricting the state space with a compactness assumption. Whittle [44], under the name *transient programming*, supported the results obtained by Veinott [43]. He extended the problem presented by Veinott [43] to infinite state and control spaces under uniform boundedness conditions on the expected termination time.

Bertsekas and Tsistsiklis ([1], [2], [3]), denoting the problem as *stochastic shortest-path problem*, improved the results of Eaton and Zadeh [12], Veinott [43] and Whittle [44], by weakening the condition that all policies be transient. They established that every stationary deterministic policy can have an associated value function that is unbounded above. Bertsekas and Tsistsiklis [4], in a subsequent work, strengthened their previous result by relaxing the condition that the set of actions available in each state be finite. They assumed that the set of actions available in each state is compact, the transition kernel is continuous over the set of actions available in each state and the cost function is semi-continuous (over the set of actions available in each state) and bounded. Hinderer and Waldmann ([17], [18]) improved the result presented by Mandl [26], Veinott [43] and Rieder [38] for finite Markovian decision processes with an absorbing set. They were interested in the critical discount factor, defined as the smallest number such that for all discount factors β smaller than this value, the limit ν of the n-stage β-discounted optimal value function exists and is finite for each choice of the one-stage reward function. Pliska [36] assumed that the cost function is bounded and that all policies are transient, additionally to the already mentioned assumptions of compact action space, continuous transition kernel, and lower semi-continuous cost function. It is important to note that this work was the first one to extend the problem to Borel states and action spaces. Hernandez-Lerma et.al [15] expanded the results of Pliska by weakening the condition that the cost function was bounded and supposing that the cost function is dominated by a given function.

The optimal stopping problem is directly related with the stochastic shortest-path problem and was investigated by Dynkin [11], and Grigelionis and Shiryaev [13], and considered extensively in the literature by others ([10], [23], [40], [44]). It is a special type of transient Markov decision process where a state-dependent cost is incurred only when invoking a stopping action which leads the system to the destination (finish); all costs are zero before stopping. For the optimal stopping problems, the associated value function of the policy (under which the stopping action is never taken) is equal to zero at all states, however it is not transient.

Shortest-path games are usually conceptualized as two-player zero-sum, games. On the one hand, the "minimizing" player seeks to drive a finite-state dynamic system to reach a terminal state along an expected least cost path. On the other hand, the “maximizer” player seeks to maximize the expected total cost interfering with the “minimizer’s” progress. In playing the game, the players implement actions simultaneously at each state, with full knowledge of the state of the system, but without knowledge of each other's current decision.

Shapley [39] provided the first work on shortest path games. In his paper, two players are successively faced with matrix-games of mixed strategies, where both the immediate cost and transition probabilities to new matrix-games are influenced by the decisions of the players. In this conceptualization, the state of the system is the matrix-game currently being played. Kushner and Chamberlain's [21] took into account undiscounted, pursuit, stochastic games. They assumed that the state space is finite with a final state corresponding to the evader being trapped and they considered pure strategies over compact action spaces. Under these considerations, they proved that there exists an equilibrium cost vector for the game which can be found through value iteration. Van der Wal [42] explored a particular case of Kushner and Chamberlain [21] research. He produced error bounds for the updates in value iteration, considering restrictive assumptions about the capability of the pursuer to capture the evader. Kumar and Shiau [22], for the case of non-negative additive cost, proved the existence of an extended real equilibrium cost vector in non-Markov randomized policies. They showed that the minimizing player can achieve the equilibrium using a stationary Markov randomized policy. In addition, for the case where the state space is finite, the maximizing player can play ε-optimally using stationary randomized policies.

Patek and Bertsekas ([32], [33]) analyzed the case of two players, where one player seeks to drive the system to termination along a least-cost path and the other seeks to prevent termination altogether. They did not assume non-negativity of the costs, the analysis is much more complicated than the corresponding analysis of Kushner and Chamberlain's [21] and generalize (to the case of two players) those for stochastic shortest path problems [4]. Patek and Bertsekas proposed alternative assumptions which guarantee that, at least under optimal policies, the terminal state is reached with probability one. They considered undiscounted additive cost games without averaging, admitting that there are policies for the “minimizer” which allow the “maximizer” to prolong the game indefinitely at infinite cost to the “minimizer”. Under assumptions which generalize deterministic shortest-path problems, they established (i) the existence of a real-valued equilibrium cost vector achievable with stationary policies for the opposing players and (ii) the convergence of value iteration and policy iteration to the unique solution of Bellman's equation. The results of Patek and Bertsekas did imply the results of Shapley [39], as well as those of Kushner and Chamberlain [21]. Because of their assumptions relating to termination, they were able to derive stronger conclusions than those made by Kumar and Shiau [22] for the case of a finite state space. In a subsequent work, Patek [34] reexamined the stochastic shortest-path formulation in the context of Markov decision processes with an exponential utility function.

Whereas previous efforts have restricted attention to track the net using Bellman's equation as a utility function, this paper uses a Lyapunov-like function as a tool for path planning ([6], [7], [8], [9]). Two different formal theoretic approaches are employed to represent the problem domain: i) Markov decision process and, ii) place-transitions Petri Nets having as a feature a Markov decision process, called Decision Process Petri nets (DPPN). The main point of this paper is its ability to represent the system-dynamic and the trajectory-dynamic properties of a decision process application. We will identify the system-dynamic properties as those characteristics related only with the global system behavior, and we will identify the trajectory-dynamic properties as those characteristics related with the trajectory function at each state that depends on a probabilistic routing policy.

Within the system-dynamic properties framework we show notions of stability. In this sense, we call equilibrium point to the state in a MDP/DPPN that does not change, and it is the last state in the net.

In the trajectory-dynamic properties framework we define the trajectory function as a Lyapunov-like function. By an appropriate selection of the Lyapunov-like function, under certain desired criteria, it is possible to optimize the trajectory. By optimizing the trajectory we understand that it is maximum or minimum reward (in a certain sense). In addition, we use the notions of stability in the sense of Lyapunov to characterize the stability properties of the MDP/DPPN. The core idea of our approach uses a non-negative trajectory function that converges in decreasing form to a (set of) final decision states. It is important to point out that the value of the trajectory function associated with the MDP/DPPN implicitly determines a set of policies, not just a single policy (in case of having several decisions states that could be reached). We call "optimum point" the best choice selected from a number of possible final decision states that may be reached (to select the optimum point, the decision process chooses the strategy that optimizes the reward).

As a result, we extend the system-dynamic framework including the trajectory-dynamic properties. We show that the system-dynamic and the trajectory-dynamic properties of equilibrium, stability and optimum-point conditions converge under certain restrictions: if the MDP/DPPN is finite then we have that a final decision state is an equilibrium point.

The paper is structured in the following manner. The next section discusses the motivation of the work. Section 3 presents the Markov decision model, and all the structural assumptions are introduced there. Section 4 presents a detailed analysis of the equilibrium, stability and optimum-point conditions for the system-dynamic and the trajectory-dynamic parts in terms of the DPPN. Thereafter, we introduce the main results of the paper, giving a detailed analysis of the definitions to join game theory and introduce the Lyapunov-Nash equilibrium point properties, in sections 5 and 6. Finally, in section 7 some concluding remarks are outlined.

2 Motivation

The definition of system stability has attracted the attention of many past and present mathematicians and physicists including Torricelli, Laplace, Lagrange and others. However, it became a transparent criterion with the publication of the work of Lyapunov in 1892 [20]. The main idea of Lyapunov is attained in the following interpretation: given an isolated physical system, if the change of the energy E for every possible state s is negative, with the exception of the equilibrium point s^*, then the energy will decrease until it finally reaches the minimum at s^*. Intuitively, this concept of stability means that a system perturbed from its equilibrium point will always return to it.

In this paper we consider dynamical systems in which the time variable changes discretely, and the system is governed by ordinary difference equations. Let us consider systems of first-order difference equations given by

$$s_{n+1} = f(s_n, a_n),\ s_{n_0} = s_0, n \in \mathrm{N}_+^{n_0} \tag{1}$$

where s_i with $i \in \mathrm{N}$ are the state variable of the system, s_0 is the initial state, a_i with $i \in \mathrm{N}$ are the action of the system, $\mathrm{N}_+^{n_0} = \{n_0, n_0+1, ..., n_0+k, ...\}, n_0 \geq 0$. The system is specified by the state transition function f, which is always assumed as a one-to-one function for any fixed a and $n \in \mathrm{N}$, continuous in all its arguments.

Lyapunov defined a scalar function L, called a Lyapunov-like function, inspired by a classical energy function, which has four important properties that are sufficient for establishing the domain of attraction of a stable equilibrium point: a) $\exists s^*$ such that $L(s^*) = 0$, b) $L(s) > 0$ for $\forall s \neq s^*$, c) $L(s) \to \infty$ when $s \to \infty$, and d) $\Delta L = L(s_{i+1}) - L(s_i) < 0,\ \ \forall i\ \ s_i \neq s^*$. The condition (a) requires the equilibrium point to have zero potential by means of a translation to the origin, b) means that the Lyapunov-like function to be semi-positive defined, c) means that there is no s^* reachable from some s, and d) means that the Lyapunov-like function has a minimum at the equilibrium point.

A system is stable ([24], [25]) if for a given set of initial states the state of the system ensures: i) to reach a given set of states and stay there perpetually or, ii) to go to a given set of states infinitely often. The conventional notions of stability in the sense of Lyapunov and asymptotic stability can be used to characterize the stability properties of discrete event systems. An important advantage of the Lyapunov approach is that it does not require high computational complexity but the difficulty lies in specifying the Lyapunov-like function for a given problem.

At this point, it is important to note that the Lyapunov-like function L is not unique, however the energy function of a system is only one of its kind. A system whose energy E decreases on the average, but not necessarily at each instance, is stable but E is not a Lyapunov-like function.

Lyapunov-like functions [20] can be used as trajectory-tracking 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).

From what we have stated before, we can deduce the following geometric interpretation of distance:

a) $L(s)$ is a measure of the distance from the starting state s_0 to any state s in the state space. This is straightforward from the fact that $\exists s^*$ such that $L(s^*)=0$ and $L(s)>0$ for $\forall s \neq s^*$.

b) The distance from the stating state s_0 to any state s_n in the state space decreases, when $n \to \infty$. It is because $L(s_{i+1})-L(s_i)<0 \ \forall_i \ s_i \neq s^*$.

A Lyapunov-like function can be considered as a distance function denoting the length from the initial state to the equilibrium point. However, it is not necessarily optimal, it usually makes a monotonic convergence to the equilibrium point. It is important to note that the Lyapunov-like function is constructed to respect the constraints imposed by the difference equation of the system. In contrast, a Euclidean metric does not take into account these factors. For that reason, the Lyapunov-like function offers a better understanding of the concept of the distance required to converge to an equilibrium point in a discrete dynamical system.

By applying the computed actions, a kind of discrete vector field can be imagined over the search graph. Each applied optimal action yields a reduction in the optimal cost-to-target value, until the equilibrium point is reached. Then, the cost-to-go values can be considered as a discrete Lyapunov function.

In our case, an optimal discrete problem, the cost-to-target values are calculated using a discrete Lyapunov-like function. Every time a discrete vector field of possible actions is calculated over the decision process. Each applied optimal action (selected via some 'criteria') decreases the optimal value, ensuring that the optimal course of action is followed and establishing a preference relation. In this sense, the criteria change the asymptotic behavior of the Lyapunov-like function by an optimal trajectory-tracking value.

Usually, the criterion in optimization problems is related with the choice of whether to minimize or maximize the optimal action. If the problem is related with energy transformations, as is classically the case in control theory, then the criterion of minimization is applied. However, if the dilemma involves a reward, typical in game theory, then maximization is considered. In this work we will arbitrary consider the criterion of minimization.

The Lyapunov-like function can be employed as a trajectory-tracking function through the use of an operator, which represents the criterion that selects the optimal action that forces the function to decreases and approaches an infimum/minimum. It forces the function to make a monotonic progress toward the equilibrium point. The Lyapunov-like function can be defined for example as

$$L^*(s_{n+1}) = \min_{a^* \in A} L(f(s_n, a_n^*)) \tag{2}$$

which means that the optimal action is chosen to reach the infimum/minimum. The function L^* works as a guide leading the system optimally from its initial state to the equilibrium point.

There exist different methods to calculate a trajectory (given a trajectory-tracking function) for tracking the net. The cost function established by the algorithm of Dijkstra working backward from the equilibrium point result in an optimal function for tracking the net. In this sense, if we consider the cost as the number of states in the net that would be traversed from the initial state to the equilibrium point. The algorithm of Dijkstra will backward-produce the shortest path in terms of the number of states needed to reach the equilibrium point.

In Dijktra's algorithm it is supposed that every edge Z in a state transition graph has associated a nonnegative cost. The cost is interpreted as the expense of applying the related action. It is usually symbolized as $c(s,a)$ representing the cost of apply action a from state s. The total cost is given by the sum of the $c(s,a)$ along the trajectory from the initial state to the equilibrium point. The optimal cost C^* is given by the least cumulative cost calculated by summing the $c(s,a)$ over all possible trajectories from s_0 to s^*.

By induction the optimal cost is calculated as following. The cost at the initial state s_0 is 0. The next state is selected by considering the optimal condition that no other state can be reached with a lower cost. Continuing with the induction, let us suppose that every state has been appropriately selected by the optimal cost condition. At the end of the process, Dijkstra's algorithm produces the shortest paths in a graph in terms of the cost needed to reach the equilibrium point s^*. The complexity time of Dijkstra's algorithm is $O(|X|\log|X|+|Z|)$, where $|X|$ and $|Z|$ are the numbers of vertices and edges respectively in a state transition graph representation.

In fact, we can make use of backward search algorithms like: 1) Breadth First with complexity $O(|X|+|Z|)$; 2) Depth First with complexity $O(|X|+|Z|)$; 3) A^*: search algorithm, which is an extension of Dijkstra's algorithm, attempting to reduce the total number of states by incorporating a heuristic function to improve the cost; and others similar, where the distance corresponds to the cost given by every state reached when the net is tracked back from the equilibrium point to the initial state.

3 Decision Model

The aim of this section is to introduce the decision model and all the structural assumptions related with the Markov model ([14], [16], [37]).

Notation 3.1 *As usual, let* $\mathbb{R}$ *be the set of real number and let* $\mathbb{N}^*$ *be the set of non-negative integers.*

Definition 3.1 *A Markov Decision Process is a 5-tuple*

$$MDP = \{S, A, \Upsilon, Q, U\} \tag{3}$$

where:

- *S is a countable set of feasible states,* $S \subset \mathbb{N}^*$, *endowed with discrete topology*[1].
- *A is the set of actions, which is a metric space. For each* $s \in S$, $A(s) \subset A$ *is the non-empty set of admissible actions at state* $s \in S$. *Without loss of generality we may take* $A = \bigcup_{s\in S} A(s)$.
- $\Upsilon = \{(s,a) \mid s \in S, a \in A(s)\}$ *is the set of admissible state-action pairs, which is a measurable subset of* $S \times A$.
- $Q = \lfloor q_{ij|k} \rfloor$ *is an array of probabilities, where* $q_{ij|k} \equiv P(s_i \mid s_j, a_k)$ *representing the probability associated with the transition from state* s_i *to state* s_j *under an action* $a_k \in A(s_i)$. *Note that for any fixed k,* $Q|_k$ *is a stochastic matrix.*
- $U : S \to \mathbb{R}_+$ *is a trajectory function, associating to each state a real value. Note that U is a function bounded*

[1] Note that the existence of a topology on *S* is trivial, since *S* is countable. We introduce it for definition compatibilities.

from below. (e.g., it is convenient to use $\|U\| = \sup_{s\in S} U(s)$).

Interpretation: The control model (3) represents a discrete-time controlled stochastic system that is observed at time $n \in \mathsf{N}^*$ Denoting by s_n and a_k the state of the system and action applied at time *n*, respectively, the interpretation of the *MDP* dynamics is as follows. At each discrete time $n \in \mathsf{N}^*$ the state of the system $s_n = s \in S$ is observed. For every action $a_n = a \in A(s)$, the probability of the system to find itself in the next state s_{n+1} at time $n+1$ is $P(s_{n+1} \mid s_n = s, a_k = a)$. Considering the previous states of the trajectory (path, orbit) $(s_0, s_1, ..., s_n)$ the value of the trajectory function *U* is obtained, and then the next state s_{n+1} is selected according to *U* applying some 'criteria'. This is the Markov property of the decision process in (3).

For each $n \in \mathsf{N}^*$ the cross product $H_n = \Upsilon^n \times S$ is the set of admissible histories up to time *n*. The vector $h_n = (s_0, a_0, ..., s_{n-1}, a_{n-1}, s_n) \in H_n$ denotes the history of the process at time *n*. Considering the previous states of the trajectory $(s_0, s_1, ..., s_n)$, and for every action $a_n \in A(s_i)$, the probability of the system to find itself in state $s_j \in S$ is $q_{ij|k}$. A policy π is a (possibly randomized) measurable rule for choosing actions, which depends on the current state. The policy $\pi_{k|i} \equiv P(a_k \mid s_i)$ represents the probability measure associated with the occurrence of an action a_n from state s_i. The set of all policies is denoted by Π.

We define a process over *S* as a finite or infinite sequence of elements of *S*. If $p = (s_0, s_1 ..., s_n)$ is a finite process, we say that s_n is the end state of *p*, and we denote it $last(p) = s_n$. For completeness, $first(p) = s_0$ denote the state in which *p* starts. Let us define the sample space $\Omega = (S \times A)^\infty$, i.e. Ω represents the set of infinite processes over *S*. Let us define the random variables $X_n : \Omega \to S$ for each $n \in \mathsf{N}$, so that we have: $X_n(\omega) = x_n$ for $\omega = (x_0, a_0, x_1, ...)$.

Let (Ω, F) be a measurable space with *F* a σ-algebra of subsets of the previously defined sample space Ω. We define a probabilistic process over *S* as a pair (S, P), where P is a probability measure on *F*. If there is an element $s_0 \in S$ such that $X_0 = s_0$, we say that s_0 is the initial state of the probabilistic process (S, P). Let $p = (s_0, ..., s_n)$ be a finite process.

We define the likelihood of *p* by $\mathsf{P}(p)$. Intuitively, $\mathsf{P}(p)$ is the probability measure of *p* to occur in an execution of the system. Be aware however that the likelihood function does not define a probability measure on the set of finite processes, since it does not sum to 1.

Let (S, P) be a probabilistic process, and let $p = (s_0, ..., s_0)$ be a finite process over *S* with $\mathsf{P}(p) > 0$. Let us consider the mapping $g : p \to \Omega$ defined by: $g(s_0, s_1, ..., s_n, X_{n+1}, X_{n+2}, ...) = (s_n, X_{n+1}, X_{n+2}, ...)$.

The mapping *g* let us define a probability measure P on (Ω, F) as follows: $\forall A \in F,\ \mathsf{P}(A) = \mathsf{P}(g^{-1}(A) \mid p)$, where $\mathsf{P}(\cdot \mid p)$ is the probability conditional on *p*. We call the new probabilistic process (S, P) the probabilistic future of process *p*. We denote by the symbol E the expectation under probability P. By construction, $s_n = last(p)$ is the initial state of the probabilistic future of *p*.

Definition 3.2 *Two given processes p and p' represent a Path of the following type:*

1) *OR if one has associated a better probability* P *to occur at the same time,*
2) *AND if having associated any probability* P *they occur at the same time,*
3) *Concur if they have associated the same probability* P *to occur at the same time.*

From the previous definition we have the following remark.

Remark 3.1 *In a Concur-Path, we have* $last(p) = last(p')$ *and therefore we also have* $\mathrm{P}(p) = \mathrm{P}(p')$.

Consider an arbitrary $s_j \in S$ and for each fixed action $a_k \in A$ we look at the previous states s_i of the state s_j, denoted by $s_{\eta_{jk}} = \{s_h : h \in \eta_{jk}\}$ where $\eta_{jk} = \{h : (s_h, a_k, s_j)\}$, that materialize the concurrent state-action pair $(s_h, a_k) \in \Upsilon$ and form the sum

$$\sum_{h \in \eta_{jk}} \pi_{k|h} q_{hj|k} U_h^{(\pi_{k|h})} \tag{4}$$

Notation 3.2 *With the intention to facilitate the notation we will represent the trajectory function U as follows:*

1) $U_i \equiv U(s_i)$ *representations of the value of U at state* s_i.

2) $U_i \equiv U_i^{(\pi)}$ *for an arbitrary policy* π.

Continuing with all the a_k's we form the vector indexed by the sequence k identified by $(k_0, k_1, ..., k_f)$ as follows:

$$\left[\sum_{h \in \eta_{jk_0}} \pi_{k_0|h} q_{hj|k_0} U_h, \sum_{h \in \eta_{jk_1}} \pi_{k_1|h} q_{hj|k_1} U_h, ..., \sum_{h \in \eta_{jk_f}} \pi_{k_f|h} q_{hj|k_f} U_h \right] \tag{5}$$

the index sequence k is the set $\lambda = \{k : a_k \in (s_h, a_k, s_j)$, and s_h running over the set $s_{\eta_{jk}}\}$, and $f = \#(\lambda)$ is the number of actions to state s_j.

Intuitively, the vector (5) represents all the possible trajectories through the actions a_k where $(k_0, k_1, ..., k_f)$ to a state s_j for a fixed *j*.

Continuing the construction of the definition of the trajectory function *U*, let us introduce the following definition.

Definition 3.3 *Let* $MDP = \{S, A, \Upsilon, Q, U\}$ *be a Markov Decision Process, let* $(s_0, s_1, ..., s_n)$ *be a realized trajectory of the system and let* $L : \mathrm{R}^n \to \mathrm{R}_+$ *be a continuous map. Then L is a Lyapunov-like function [20] if it satisfies the following properties:*

1) $\exists s^*$ *such that* $L(s^*) = 0$,

2) $L(s) > 0$ *for* $\forall s \neq s^*$,

3) $L(s) \to \infty$ *when* $s \to \infty$,

4) $\Delta L = L(s_{i+1}) - L(s_i) < 0 \quad \forall i \quad s_i \neq s^*$.

From the previous definition we have the following remark.

Remark 3.2 *In definition 3.3 point 3 we state that* $L(s) \to \infty$ *when* $s \to \infty$ *meaning that there is no* s^* *reachable from some s.*

Then, formally we define the trajectory function *U* as follows:

Definition 3.4 *For the discrete time* $n \in \mathrm{N}^*$ *the trajectory function U with respect a Markov Decision Process* $MDP = \{S, A, \Upsilon, Q, U\}$ *is represented by*

$$U_j = \begin{cases} U_0 & \text{if } \; n = 0 \\ L(\alpha) & \text{if } \; n > 0 \end{cases} \tag{6}$$

where

$$\alpha = \left[\sum_{h \in \eta_{jk_0}} \pi_{k_0|h} q_{hj|k_0} U_h, \sum_{h \in \eta_{jk_1}} \pi_{k_1|h} q_{hj|k_1} U_h, \ldots, \sum_{h \in \eta_{jk_f}} \pi_{k_f|h} q_{hj|k_f} U_h \right] \tag{7}$$

the function $L : D \subseteq \mathbb{R}_+^n \to \mathbb{R}_+$ *is a function that optimizes the reward through all possible transitions (i.e. trough all the possible trajectories defined by the different* a_k *'s), D is the decision set formed by the k's:* $0 \le k_l \le f$ *of all those possible transitions* (s_h, a_k, s_j), η_{jk} *is the index sequence of the list of previous places to* s_j *through action* a_k *and* s_h $(h \in \eta_{jk_l})$ *is a specific previous place of* s_j *through action* a_k.

From the above definition we have the following remark.

Remark 3.3

- *Note that the Lyapunov-like function L guarantees that the optimal course of action is followed (taking into account all the possible paths defined). In addition, the function L establishes a preference relation because by definition L is asymptotic; this condition gives to the decision maker the opportunity to select a path that optimizes the reward.*
- *The iteration over time* $n \in \mathbb{N}^*$ *for U is as follows:*
  1. *for* $n = 0$ *the trajectory function value is* U_0 *at state* s_0 *and for the rest of the states* s_i *the value is 0,*
  2. *for* $n > 0$ *the trajectory function value is* U_j *at each state* s_j, *is computed by taking into account the value of the previous states* s_i.

Property 3.1 *The function U satisfies the following properties:*

1) $\exists s^\Delta$ *such that*
   a) *if there exists an infinite sequence* $\{s_i\}_{i=1}^\infty$ *with* $s_n \underset{n\to\infty}{\to} s^\Delta$ *(*s_n *converge at* s^Δ*) such that* $0 \le \ldots < U_n < U_{n-1} \ldots < U_1$, *then* $U(s^\Delta)$ *is the infimum of the infinite sequence, i.e.* $U(s^\Delta) = 0$,
   b) *if there exists a finite sequence* $s_1, \ldots, s_n$ *with* $s_1, \ldots, s_n \to s^\Delta$ *(* $s_1, \ldots, s_n$ *converge at* s^Δ*) and there exists a constant* $C \in \mathbb{R}$ *such that* $C = U_n < U_{n-1} \ldots < U_1$, *then* $U(s^\Delta)$ *is the minimum of the finite sequence, i.e.* $U(s^\Delta) = C$, $(C \in \mathbb{R}$ *and* $s^\Delta = s_n)$.
2) *there exists a constant* $C \in \mathbb{R}$ *such that* $U(s_i) > \max\{0, C\}$, $\forall s_i$ *such that* $s_i \ne s^\Delta$.
3) $\forall s_i, s_{i-1}$ *such that* $s_{i-1} \le_U s_i$ *then* $\forall i \; \Delta U_i = U_i - U_{i-1} < 0$ *(a trajectory function* $U : S \to \mathbb{R}$ *is consistent with the preference relationship of a decision problem* $(S, \le)$ *if* $\forall w, z \in S : w \le_u z$ *if and only if* $U_w \le U_z$*).*

Property 3.2 *The trajectory function U is a Lyapunov-like function.*

Explanation. Intuitively, a Lyapunov-like function can be considered as routing function and optimal cost function. In our case, an optimal discrete problem, the cost-to-target values are calculated using a discrete Lyapunov-like function. Every time a discrete vector field of possible actions is calculated over the decision process. Each applied optimal action (selected via some `criteria') decreases the optimal value, ensuring that the optimal course of action is followed and establishing a preference relation. In this sense, the criteria 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 from Bellman's equation.

Remark 3.4 *From property 3.1 and 3.2 we have that:*

- $U(s^{\Delta})=0$ *or* $U(s^{\Delta})=C$ *(for a given C) means that a final state is reached. Without loss of generality we can say that* $U(s^{\Delta})=0$ *by means of a translation to the origin.*
- *In property 3.1 we determine that the Lyapunov-like function* $U(s)$ *approaches to an infimum/minimum when* s *is large thanks to property 4 of definition 3.3.*
- *Property 3.1, point 3 is equivalent to the following statement:* $\exists\{\varepsilon_i\}, \varepsilon_i > 0$ *such that* $|U_i - U_{i-1}| > \varepsilon_i$, $\forall s_i, s_{i-1}$ *such that* $s_{i-1} \leq_U s_i$.
- *Property 3.1, point 3 means that the Lyapunov-like function* $U(s)$ *will progress without cycling and the equilibrium point will eventually be reached.*

For instance, the trajectory function *U* being equal to the entropy is a particular Lyapunov-like function used in information theory as a measure of the informational disorder.

Remark 3.5 *It is important to note that the trajectory function value can be re-normalized after each transition of the net. That is, when* $\pi \neq 0$ *and* $q=0$ *implies a re-normalization of the* π*'s.*

Remark 3.6 *In property 3.1 point 3 we state that* $\Delta U = U_i - U_{i-1} < 0$ *for determining the asymptotic condition of the Lyapunov-like function. However, it easy to show that such property is convenient for deterministic systems. In Markov decision process systems is necessary to include probabilistic decreasing asymptotic conditions to guarantee the asymptotic condition of the Lyapunov-like function.*

Remark 3.7 *We are using* $[\ ]$ *to denote the OR-Path,* $\sum$ *to denote the AND-Path, and* $\{\ |$ *to denote the Conc-Path.*

4 Decision Petri Nets Model

We introduce the concept of Decision Process Petri nets (DPPN) by locally randomizing the possible choices, for each individual place of the Petri net ([Clempner1], [Clempner3]).

Definition 4.1 *A Decision Process Petri net is a 7-tuple* $DPPN=\{P,Q,F,W,M_0,\pi,U\}$ *where*

- $P=\{p_0,p_1,p_2,...,p_m\}$ *is a finite set of places,*
- $Q=\{q_1,q_2,...,q_n\}$ *is a finite set of transitions,*
- $F \subseteq I \cup O$ *is a set of arcs where* $I \subseteq (P \times Q)$ *and* $O \subseteq (Q \times P)$ *such that* $P \cap Q = \varnothing$ *and* $P \cup Q \neq \varnothing$,
- $W : F \rightarrow \mathrm{N}_{+}^{1}$ *is a weight function,*
- $M_0 : P \rightarrow \mathrm{N}$ *is the initial marking,*
- $\pi : I \rightarrow \mathrm{R}_{+}$ *is a routing policy representing the probability of choosing a particular transition (routing arc), such that for each* $p \in P, \sum_{q_j : (p,q_j) \in I} \pi((p,q_j)) = 1,$
- $U : P \rightarrow \mathrm{R}_{+}$ *is a trajectory-tracking function.*

The previous behavior of the *DPPN* is described as follows. When a token reach a place, it is reserved for the firing of a given transition according to the routing policy determined by *U*. A transition *q* must fire as soon as all the places $p_1 \in P$ contain enough tokens reserved for transition *q*. Once the transition fires, it consumes the corresponding tokens and immediately produces an amount of tokens in each subsequent place $p_2 \in P$. When $\pi(\iota)=0$ for $\iota \in I$ means that there are no arcs in the place-transitions Petri net.

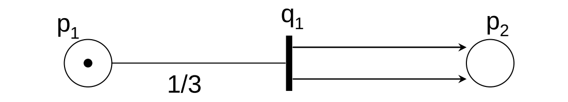

Figure 1. DPPN Routing policy case 1

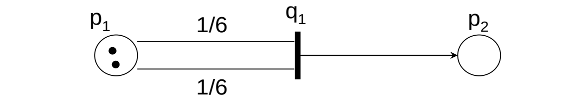

Figure 2. DPPN Routing policy case 2

In Figures 1 and 2 we have represented partial routing policies π that generates a transition from state p_1 to state p_2 where $p_1, p_2 \in P$:

- case 1. In figure 1 the probability that q_1 generates a transition from state p_1 to p_2 is 1/3. But, because q_1 transition to state p_2 has two arcs, the probability to generate a transition from state p_1 to p_2 is increased to 2/3.
- case 2. In Figure 2 we set by convention for the probability that q_1 generates a transition from state p_1 to p_2 is 1/3 (1/6 plus 1/6). However, because q_1 transition to state p_2 has only one arc, the probability to generate a transition from state p_1 to p_2 is decreased to 1/6.
- case 3. Finally, we have the trivial case when there exists only one arc from p_1 to q_1 and from q_1 to p_2.

It is important to note, that by definition the trajectory-tracking function *U* is employed only for establishing a trajectory tracking, working in a different execution level of that of the place-transitions Petri net. The trajectory-tracking function *U* in no way changes the place-transitions Petri net evolution or performance.

$U_k(.)$ denotes the trajectory-tracking value at place $p_i \in P$ at time *k* and let $U_k = [U_k(.),...,U_k(.)]^T$ denote the trajectory-tracking state of DPPN at time *k*. $FN : F \to \mathbb{R}_+$ is the number of arcs from place *p* to transition *q* (the number of arcs from transition *q* to place *p*).

Consider an arbitrary $p_i \in P$ and for each fixed transition $q_j \in Q$ that forms an output arc $(q_j, p_i) \in O$, we look at all the previous places p_h of the place p_i denoted by the list (set) $p_{\eta_{ij}} = \{p_h : h \in \eta_{ij}\}$ where $\eta_{ij} = \{h : (p_h, q_j) \in I \;\&\; (q_j, p_i) \in O\}$, that materialize all the input arcs $(p_h, q_j) \in I$ and form the sum

$$\sum_{h\in\eta_{ij}} \Psi(p_h, q_j, p_i) * U_k(p_h) \qquad (8)$$

where $\Psi(p_h, q_j, p_i) = \pi(p_h, q_j) * \frac{FN(q_j, p_i)}{FN(p_h, q_j)}$ and the index sequence *j* is the set $\{j : q_j \in (p_h, q_j) \cap (q_j, p_i) \;\&\; p_h$ running over the set $p_{\eta_{ij}}\}$.

Proceeding with all the q_j's we form the vector indexed by the sequence *j* identified by $(j_0, j_1, ..., j_f)$ as follows:

$$\left[\sum_{h\in\eta_{ij0}} \Psi(p_h, q_{j_0}, p_i) * U_k(p_h), \sum_{h\in\eta_{ij1}} \Psi(p_h, q_{j_1}, p_i) * U_k(p_h), ..., \sum_{h\in\eta_{ijf}} \Psi(p_h, q_{j_f}, p_i) * U_k(p_h)\right] \qquad (9)$$

Intuitively, the vector (9) represents all the possible trajectories through the transitions q_j's where $(j_1, j_2, ..., j_f)$ to a place p_i for a fixed *i*.

Then, formally we define the trajectory-tracking function U as follows:

Definition 4.2 *The trajectory-tracking function* U *with respect a Decision Process Petri net* $DPPN = \{P,Q,F,W,M_0,\pi,U\}$ *is represented by the equation*

$$U_k^{q_j}(p_i) = \begin{cases} U_k(p_0) & if \;\; i = 0, k = 0 \\ L(\alpha) & if \;\; i > 0, k = 0 \,\&\, i \geq 0, k > 0 \end{cases} \tag{10}$$

where

$$\alpha = \begin{bmatrix} \sum\limits_{h \in \eta_{ij_0}} \Psi(p_h, q_{j_0}, p_i) * U_k^{q_{j0}}(p_h), \sum\limits_{h \in \eta_{ij_1}} \Psi(p_h, q_{j_1}, p_i) * U_k^{q_{j1}}(p_h), ..., \\ \sum\limits_{h \in \eta_{ij_f}} \Psi(p_h, q_{j_f}, p_i) * U_k^{q_{jf}}(p_h) \end{bmatrix} \tag{11}$$

the function $L : D \subseteq \mathrm{R}_+^n \to \mathrm{R}_+$ *is a Lyapunov-like function, which optimizes the trajectory-tracking value through all possible transitions (i.e. trough all the possible trajectories defined by the different* q_j *s),* D *is the decision set formed by the j's;* $0 \leq j \leq f$ *of all those possible transitions* $(q_j, p_i) \in O$, $\Psi(p_h, q_j, p_i) = \pi(p_h, q_j) * \frac{FN(q_j, p_i)}{FN(p_h, q_j)}$, η_{ij} *is the index sequence of the list of previous places to* p_i *through transition* q_j , p_h $(h \in \eta_{ij})$ *is a specific previous place of* p_i *through transition* q_j.

From the previous definition we have the following remark.

Remark 4.1 *The iteration over* k *for* U *is as follows:*

- *for* $i = 0$ *and* $k = 0$ *the trajectory-tracking value is* $U_0(p_0)$ *at place* p_0 *and for the rest of the places* p_i *the value is 0,*
- *for* $i \geq 0$ *and* $k > 0$ *the trajectory-tracking value is* $U_k^{q_j}(p_i)$ *at each place* p_i, *is computed by taking into account the trajectory-tracking value of the previous places* p_h *for* k *and k-1 (when needed).*

4.1 DPPN Mark-Dynamic Properties

Theorem 4.1 *The Decision Process Petri net* $DPPN = \{P,Q,F,W,M_0,\pi,U\}$ *is finite and bounded by a place p.*

Theorem 4.2 *Let* $DPPN = \{P,Q,F,W,M_0,\pi,U\}$ *be a Decision Process Petri net bounded by a state s. Then, a Lyapunov-like trajectory function can be constructed iff p is reachable from* p_0.

Definition 4.3 *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* $M_l(p^*) = S < \infty$, $\forall l \geq k$ *and* p^* *is the last place of the net.*

Theorem 4.3 *The Decision Process Petri net* $DPPN = \{P,Q,F,W,M_0,\pi,U\}$ *is uniformly practically stable iff there exists a* Φ *strictly positive* m *vector such that* $\Delta v = u^T A \Phi \leq 0$ *([31]).*

4.2 DPPN Trajectory-Dynamic Properties

Definition 4.4 *A final decision point* $p_f \in P$ *with respect a Decision Process Petri net*

$DPPN = \{P, Q, F, W, M_0, \pi, U\}$ *is a place* $p \in P$ *where the infimum is asymptotically approached (or the minimum is attained), i.e.* $U(p) = 0$ *or* $U(p) = C$ *(for a specified C)* .

Definition 4.5 *An optimum point* $p^{\Delta} \in P$ *with respect a Decision Process Petri net* $DPPN = \{P, Q, F, W, M_0, \pi, U\}$ *is a final decision point* $p_f \in P$ *where the best choice is selected 'according to some criteria'.*

Property 4.1 *Every Decision Process Petri net* $DPPN = \{P, Q, F, W, M_0, \pi, U\}$ *has a final decision point.*

Remark 4.3 *In case that* $\exists p_1, ..., p_n \in P,$ *such that* $U(p_1) = ... = U(p_n) = 0,$ *then* $p_1, ..., p_n$ *are optimum points.*

Theorem 4.4 *Let* $DPPN = \left(P, Q, F, W, M_0, \pi, U\right)$ *be a finite and non-blocking Decision Process Petri net and let* $p_0 = p_{\varsigma(1)} \leq_{U_k} p_{\varsigma(2)} \leq_{U_k} ... \leq_{U_k} p_{\varsigma(n)} \leq_{U_k} ...$ *a realized trajectory which converges to* p^{Δ} *such that* $\exists \varepsilon_{\varsigma(j)} : \left| U(p_{\varsigma(j)}) - U(p_{\varsigma(i)}) \right| > \varepsilon_{\varsigma(j)}$ *(with* $\varepsilon_{\varsigma(j)} > 0)$ *.Let* $\varepsilon = \min\{\varepsilon_{\varsigma(j)}\},$ *then the optimum decision point* p^{Δ} *is reached in a time step bounded by* $O(U(p_0)/\varepsilon).$

Theorem 4.5 *Let* $DPPN = \left(P, Q, F, W, M_0, \pi, U\right)$ *be a finite and non-blocking Decision Process Petri net. Then, U converges to an optimum (final) decision point* p^{Δ} $(p_f).$

Explanation. Bellman's equation is expressed as a sum over the state of a trajectory needs to be solved backwards in time from the equilibrium point. It results in an optimal function when is governed by Bellman's principle, described by Poznyak [37], producing the shortest path needed to reach a known equilibrium point. Notice that the necessity to know the equilibrium point beforehand when applying the equation is a significant constraint, given that, in many practical situations, the state space of a DPPN is too large for an easy identification of the equilibrium point.

Moreover, algorithms using Bellman's equation usually solve the problem in two phases: pre-processing and search. In the pre-processing phase, the distance is usually calculated between each state and the equilibrium points (final states) of the problem, in a backward direction. Then, in the search phase, these results are employed to calculate the distance between each state and the equilibrium points, leading the search process in a forward search.

Lyapunov-like functions can be used as forward trajectory-tracking functions. Each applied optimal action produces a monotonic progress toward an equilibrium point. Because it is a solution to the difference equation, it will naturally lead the system from the starting state to the equilibrium point. Tracking the state space in a forward direction lets the decision maker to avoid invalid states that occur in the space generated by a backward search. In most cases, the forward search gives the impression to be more useful than the backward search. The explanation is that in the backward direction, when the case of incomplete final states arises, invalid states appear, which cause obvious problems.

Proposition 4.1 *Let* $DPPN = \{P, Q, F, W, M_0, \pi, U\}$ *be a Decision Process Petri net and let* $p^{\Delta} \in P$ *an optimum point. Then* $U(p^{\Delta}) \leq U(p),$ $\forall p \in P$ *such that* $p \leq_U p^{\Delta}.$

Theorem 4.6 *The Decision Process Petri net* $DPPN = \{P, Q, F, W, M_0, \pi, U\}$ *is uniformly practically stable iff* $U(p_{i+1}) - U(p_i) \leq 0.$

4.3 Convergence of the DPPN Mark-Dynamic and Trajectory-Dynamic Properties

Theorem 4.7 *Let $DPPN = \{P, Q, F, W, M_0, \pi, U\}$ be a Decision Process Petri net. If $p^* \in P$ is an equilibrium point then it is a final decision point.*

Theorem 4.8 *Let $DPPN = \{P, Q, F, W, M_0, \pi, U\}$ be a finite and non-blocking Decision Process Petri net (unless $p \in P$ is an equilibrium point). If $p_f \in P$ is a final decision point then it is an equilibrium point.*

Corollary 4.1 *Let $DPPN = \{P, Q, F, W, M_0, \pi, U\}$ be a finite and non-blocking Decision Process Petri net (unless $p \in P$ is an equilibrium point). Then, an optimum point $p^\Delta \in P$ is an equilibrium point.*

Remark 4.4 *The finite and non-blocking (unless $p \in P$ is an equilibrium point) condition over the DPPN can not be relaxed:*

1) Let us suppose that the DPPN is not finite, i.e. p is in a cycle then, the Lyapunov-like function converges when $k \to \infty$, to zero i.e., $L(p) = 0$ but the DPPN has no final place, and therefore, it is not an equilibrium point.

2) Let us suppose that the DPPN blocks at some place (not an equilibrium point) $p_b \in P$. Then, the Lyapunov-like function has a minimum at a place p_b, lets say $L(p_b) = C$, but p_b is not an equilibrium point, because it is not necessarily the last place of the net.

5 Game Theory Model

The interaction among players obligates each player to develop a belief about the possible strategies of the other players. Nash equilibria are supported by two premises: i) each player behaves rationally given the beliefs about the other players' strategies; and ii) these beliefs are correct. Both premises allow us to regard the Nash equilibrium point as a steady-state of the strategic interaction. In particular, the second premise makes this an equilibrium concept, because when every individual is acting in agreement with the Nash equilibrium, no one has the need to take another strategy.

Notation 5.1 *As usual let $\mathbb{R}$ be the set of real number and let $\mathbb{N}^*$ be the set of non-negative integers. Let $C \in \mathbb{R}$ be a given constant, let $\bar{\mathbf{0}}$ be the vector $(0,...,0) \in \mathbb{R}_d$ and let $\bar{\mathbf{C}}$ be the vector of constants $(C,...,C) \in \mathbb{R}_d$.*

A game is a finite set of states S_i for each player and a trajectory function U^i for each player mapping $S_1 \times ... \times S_n$ to the integers.

Definition 5.1 *A Game Markov Decision Process is a 6-tuple $GMDP = \{N, S, A, \Upsilon, Q, U\}$ where:*

- *$N = \{1, 2, ..., n\}$ denotes a finite set of players.*
- *$S = S_1 \times ... \times S_n$ is a countable set of feasible states, endowed with discrete topology.*
- *$A = A_1 \times ... \times A_n$ is the set of actions, which is a metric space. For each $s \in S$, $A(s) \subset A$ is the nonempty set of admissible actions at state $s \in S$. Without loss of generality we may take $A = \bigcup_{s \in S} A(s)$.*
- *$\Upsilon = \{(s, a) \mid s \in S, a \in A(s)\}$ is the set of admissible state-action pairs, which is a measurable subset of $S \times A$.*
- *$Q = \lfloor q^\iota_{ij|k} \rfloor$ such that $\iota \in N$, is an array of probabilities, where $q^\iota_{ij|k} \equiv P^\iota(s_i \mid s_j, a_k)$ represents the probability associated with the occurrence transition from state s_i to state s_j under an action $a_k \in A(s_i)$ for player ι. Note that for any fixed k, $Q|_k$ is a stochastic matrix.*
- *$U : S \to \mathbb{R}^n_+$ is a trajectory function, associating to each state a vector of real values. U^ι is a trajectory*

function for player ι mapping $S_1 \times ... \times S_n$ to a corresponding integer. Note that U is a function bounded from below.

Definition 5.2 *Let $GMDP = \{N, S, A, \Upsilon, Q, U\}$ be a Markov Decision Process, let $(s_0, s_1, ..., s_n)$ be a realized trajectory of the system and let $L : \mathbb{R}^n \to \mathbb{R}^n_+$ be a continuous map. Then, L is a vector Lyapunov-like function [20] if it satisfies the following properties:*

1) $L(s_1,...,s_n) = [L_1(s_1), \ \ ..., \ \ L_n(s_n)]$,

2) if $\exists s_i^*$ *such that* $\forall i \ L_i(s_i^*) = 0$ *then* $L(s_1^*,...,s_n^*) = \bar{\mathbf{0}}$,

3) if $L_i(s_i) > 0$ *for* $\forall s_i \neq s_i^*$ *then* $L(s_1,...,s_n) > \bar{\mathbf{0}}$,

4) if $L_i(s_i) \to \infty$ *when* $s_i \to \infty$ *then* $L(s_1,...,s_n) \to \infty$,

5) if $\Delta L_i = L_i(s_i) - L_i(t_i) < 0$ *for all* $(t_1,...,t_n) \leq_U (s_1,...,s_n)$ *and* $(s_1,...,s_n),(t_1,...,t_n) \neq (s_1^*,...,s_n^*)$ *then* $\Delta L = L(s_1,...,s_n) - L(t_1,...,t_n) < \bar{\mathbf{0}}$.

Then, formally we define the trajectory function *U* as follows:

Definition 5.3 *For the discrete time $n \in \mathbb{N}^*$ the trajectory function U for a player $\iota \in N$ with respect a Game Markov Decision Process $GMDP = \{N, S, A, \Upsilon, Q, U\}$ is represented by*

$$U_j^\iota = \begin{cases} U_0^\iota & \text{if } n = 0 \\ L(\alpha) & \text{if } n > 0 \end{cases} \tag{12}$$

where

$$\alpha = \left[\sum_{h \in \eta_{jk_0}} \pi^\iota_{k_0|h} q^\iota_{hj|k_0} U^\iota_h, \sum_{h \in \eta_{jk_1}} \pi^\iota_{k_1|h} q^\iota_{hj|k_1} U^\iota_h, ..., \sum_{h \in \eta_{jk_f}} \pi^\iota_{k_f|h} q^\iota_{hj|k_f} U^\iota_h \right] \tag{13}$$

the function $L : D \subseteq \mathbb{R}^n_+ \to \mathbb{R}^n_+$ is a function that optimizes the trajectory function value through all possible transitions (i.e. trough all the possible trajectories defined by the different a_k s), D is the decision set formed by the k's; $0 \leq k_l \leq f$ of all those possible transitions (s_h, a_k, s_j), η_{jk} is the index sequence of the list of previous places to s_j through action a_k, s_h $(h \in \eta_{jk_l})$ is a specific previous place of s_j through action a_k.

6 Game Petri Nets Model

The aim of this section is to associate to any game a Game Petri net -- GPN -- ([8], [9]). The GPN structure will represent all the possible strategies existing within the game.

Definition 6.1 *A Game Petri Net is a 8-tuple $GPN = (N, P, Q, F, W, M_0, \pi, U)$ where:*

- $N = \{1, 2, ..., n\}$ denotes a finite set of players
- $P = P_1 \times P_2 \times ... \times P_n$ *is the set of places that represents the Cartesian product of states (each tuple is represented by a place).*
- $Q = Q_1 \times Q_2 \times ... \times Q_n$ *is the set of transitions that represents the Cartesian product of the conditions (each tuple is represented by a transition).*
- $F \subseteq I \cup O$ *is a set of arcs where* $I \subseteq (P \times Q)$ *and* $O \subseteq (Q \times P)$ *such that* $P \cap Q = \emptyset$ *and* $P \cup Q \neq \emptyset$,
- $W : F \to \mathbb{N}^n$ *is a weight function,*

- $M_0 : P \rightarrow \mathsf{N}^n$ *is the initial marking,*
- $\pi \ : \ I \rightarrow \mathsf{R}_+^n$ *is a routing policy representing the probability of choosing a particular transition (routing arc), such that for each* $p_i \in P$, $\sum_{q_{j_\iota} \ :(p_{i_\iota}, q_{j_\iota}) \in I} \pi((p_{i_\iota}, q_{j_\iota})) = 1$, $\forall \iota \in N$,
- $U \ : \ P \rightarrow \mathsf{R}_+^n$ *is a trajectory function.*

Interpretation: the previous behavior of the *GPN* is described as follows. When a token reaches a place, it is reserved for the firing of a given transition according to the routing policy determined by *U*. A transition *q* must fire as soon as all the places $p_1 \in P$ contain enough tokens reserved for transition *q*. Once the transition fires, it consumes the corresponding tokens and immediately produces an amount of tokens in each subsequent place $p_2 \in P$. When $\pi(\delta) = 0$ for $\delta \in I$ means that there are no arcs in the place-transitions Petri net.

It is important to note that, by definition, the trajectory function *U* is employed only for establishing a trajectory tracking, working in a different execution level of that of the place-transitions Petri net. The trajectory function *U* changes in no way the place-transitions Petri net evolution or performance.

$U_k(.)$ denotes the trajectory value at place $p_i \in P$ at time k and let $[U_k] = [U_k(.),...,U_k(.)]^T$ denote the trajectory value state of GPN at time *k*. $FN \ : \ F \rightarrow \mathsf{R}_+$ is the number of arcs from place p to transition q (the number of arcs from transition q to place p). The rest of the GPN functionality is as described in the PN preliminaries.

Let us recall some basic notions in game theory. We denote $S_\iota = \{s_i\}$ the set of pure strategies for player ι (strategies are represented by the probability that a transition can be fired in the GPN). For notational convenience we write $S = \prod_{\iota \in \mathbf{N}} S_\iota$ (the pure strategies profile), and $S_{-\iota} = \prod_{j \in \mathbf{N}|\{\iota\}} S_j$ (the pure strategies profile of all the players but for player ι). For an action tuple $s = (s_1,...,s_n) \in S$ we denote $s_{-\iota} = (s_1,...,s_{\iota-1}, s_{\iota+1},...,s_n)$ and, with an abuse of notation, $s = (s_\iota, s_{-\iota})$.

Similarly, we denote $\Gamma_\iota = \{\sigma_i\}$ the set of mixed strategies for player ι, identified with the routing policy representing the probability of choosing a particular transition. Analogously, we use notations $\Gamma = \prod_{\iota \in \mathbf{N}} \Gamma_\iota$ to denote the mixed strategies profile that combine strategies one for each player and $\Gamma_{-\iota} = \prod_{j \in \mathbf{N}|\{\iota\}} \Gamma_j$ to denote the mixed strategies profile of all the players except for player ι. For a strategy tuple $\sigma = (\sigma_1,...,\sigma_n) \in \Gamma$ we denote $\sigma_{-\iota} = (\sigma_1,...,\sigma_{\iota-1}, \sigma_{\iota+1},...,\sigma_n)$ and, with an abuse of notation, $\sigma = (\sigma_\iota, \sigma_{-\iota})$. For a strategy profile $\sigma_{-\iota}$, we write $\sigma_{-\iota} = \prod_{j \in \mathbf{N}|\{\iota\}} \sigma_j$, the probability identified with the routing policy π that the opponents of player ι play strategy profile $s_{-i} \in S_{-i}$. We restrict our attention to independent strategy profiles. For our construction of the GPN a strategy profile determines an outcome representing the corresponding trajectory value of each player.
Then, formally we introduce the following definitions.

Definition 6.2 *A final decision point* $p_f \in P$ *with respect to a Game Petri net* $GPN = (N, P, Q, F, W, M_0, \pi, U)$ *is a place* $p \in P$ *where the infimum is asymptotically approached (or the minimum is attained), i.e.* $U(p) = \bar{\mathbf{0}}$ *or* $U(p) = \bar{\mathbf{C}}$.

Definition 6.3 *An optimum point* $p^\Delta \in P$ *with respect a Game Petri net* $GPN = (N, P, Q, F, W, M_0, \pi, U)$ *is a final decision point* $p_f \in P$ *where the best choice is selected 'according to some criteria'.*

Definition 6.4 *A strategy with respect to a Game Petri net* $GPN = (N, P, Q, F, W, M_0, \pi, U)$ *is identified by* σ *and* *consists of the routing policy transition sequence represented in the GPN graph model such that some point* $p \in P$ *is reached.*

Definition 6.5 *An optimum strategy with respect a Game Petri net* $GPN = \left(N, P, Q, F, W, M_0, \pi, U\right)$ *is identified by* σ^{Δ} *and consists of the routing policy transition sequence represented in the GPN graph model such that an optimum point* $p^{\Delta} \in P$ *is reached.*

Remark 6.1 *It is important to note that a strategy can be conceptualized in a different manner depending on the implementation point of view. It can be implemented as the probability that a transition can be fired, as usual, or a more general definition is as a chain of such probabilities. Both perspectives are correct, however in the latter case, we only have to give an interpretation to the strategy optimality in terms of the chain of transitions.*

Consider an arbitrary $p_i \in P$ and for each fixed transition $q_j \in Q$ that forms an output arc $(q_j, p_i) \in O,$ we look at all the previous places p_h of the place p_i denoted by the list (set) $p_{\eta_{ij}} = \{p_h \, : \, h \in \eta_{ij}\}$ where $\eta_{ij} = \left\{h \, : (p_h, q_j) \in I \,\&\, (q_j, p_i) \in O\right\},$ that materialize all the input arcs $(p_h, q_j) \in I$ and form the sum

$$\sum_{h \in \eta_{ij}} \left(\left\langle \sigma_{hj}(p_i) * U_k^{\sigma_{hj}}(p_h) \right\rangle \right)_{\iota} \tag{14}$$

where $\sigma_{hj}(p_i) = (\pi(p_{h_1}, q_{j_1}) * \frac{FN(q_j, p_i)}{FN(p_h, q_j)}, \pi(p_{h_2}, q_{j_2}) * \frac{FN(q_j, p_i)}{FN(p_h, q_j)}, ..., \pi(p_{h_n}, q_{j_n}) * \frac{FN(q_j, p_i)}{FN(p_h, q_j)})$ where $\left(\langle * \rangle\right)_{\iota}$ represent the product of the vector element by element, i.e. $\left(\left\langle (a_1, a_2, ..., a_n) * (b_1, b_2, ..., b_n) \right\rangle\right)_{\iota} = (a_1 b_1, a_2 b_2, ..., a_n b_n),$ p_{h_ι} is the $\iota \in N$ element of the tuple routing policy π, and the index sequence j_ι is the set $\{ j_\iota \, : \forall \iota \;\; q_{j_\iota} \in (p_{h_\iota}, q_{j_\iota}) \cap (q_{j_\iota}, p_{i_\iota})$ & p_{h_ι} running over the set $p_{\eta_{ij}} \}$. The quotient $\frac{FN(q_j, p_i)}{FN(p_h, q_j)}$ is used for normalizing the routing policies π, note that in the formula of $\sigma_{hj}(p_i)$ it is not necessary to specify $\forall \iota,$ $FN(q_{j_\iota}, p_{i_\iota})$ and $FN(p_{h_\iota}, q_{j_\iota})$ for calculating $\frac{FN(q_{j_\iota}, p_{i_\iota})}{FN(p_{h_\iota}, q_{j_\iota})}$ because the number of arcs ($FN(\cdot,\cdot)$) is the same for all players.

Proceeding with all the $q_j's$ for a given player $\iota \in N$ we form the vector indexed by the sequence *j* identified by $(j_0, j_1, ..., j_f)$ as follows:

$$\alpha = \begin{bmatrix} \sum\limits_{h \in \eta_{ij_0}} \left(\left\langle \sigma_{hj_0}(p_i) * U_k^{\sigma_{hj_0}}(p_h) \right\rangle \right)_{\iota}, \sum\limits_{h \in \eta_{ij_1}} \left(\left\langle \sigma_{hj_1}(p_i) * U_k^{\sigma_{hj_1}}(p_h) \right\rangle \right)_{\iota}, ..., \\ \sum\limits_{h \in \eta_{ij_f}} \left(\left\langle \sigma_{hj_f}(p_i) * U_k^{\sigma_{hj_f}}(p_h) \right\rangle \right)_{\iota} \end{bmatrix} \tag{15}$$

Intuitively, the vector (17) represents all the possible trajectories through the transitions $q_j's$ where $(j_1, j_2, ..., j_f)$ to a place p_i for a fixed *i* and a given player $\iota \in N$.

Then, formally we define the trajectory function *U* as follows:

Definition 6.6 *The trajectory function U for a given player* $\iota \in N$ *with respect a Game Petri net* $GPN = \left(N, P, Q, F, W, M_0, \pi, U\right)$ *is represented*

$$U_{k,\iota}^{\sigma_{hj}}(p_i) = \begin{cases} U_k(p_0) & if \;\; i = 0, k = 0 \\ L(\alpha) & if \;\; i > 0, k = 0 \,\&\, i \geq 0, k > 0 \end{cases} \tag{16}$$

where the vector function $L : D \subseteq \mathsf{R}_{+}^{n} \rightarrow \mathsf{R}_{+}^{n}$ *is a vector Lyapunov-like function which optimizes the trajectory value through all possible strategies (i.e. trough all the possible trajectories defined by the different* q_j*'s), D is the decision set formed by the j's (* $0 \le j \le f$ *) of all those possible transitions* $(q_j, p_i) \in O$*, and* α *is given in (17).*

Theorem 6.1 *Let* $GPN = (N, P, Q, F, W, M_0, \pi, U)$ *be a non-blocking Game Petri net (unless* $p \in P$ *is an equilibrium point) then we have that:*

$$U_k^{\Delta}(p^{\Delta}) \le U_k(p), \;\; \forall \sigma, \sigma^{\Delta}$$

Remark 6.2 *The inequality* $U_k^{\Delta}(p^{\Delta}) \le U_k(p)$ *means that the trajectory-tracking value is optimum when the optimum strategy is applied.*

Corollary 6.1 *Let* $GPN = (N, P, Q, F, W, M_0, \pi, U)$ *be a non blocking Game Petri net (unless* $p \in P$ *is an equilibrium point) and let* σ^{Δ} *an optimum strategy. Set* $L = \left[\min\limits_{i=1,\ldots,|\alpha|} \{\alpha_i\}, \;\; \ldots, \;\; \min\limits_{i=1,\ldots,|\alpha|} \{\alpha_i\} \right]$ *then,* $U_k^{\Delta}(p)$ *is equal to:*

$$\underbrace{\begin{bmatrix} \sigma_{0j_m}^{\Delta}(p_{\varsigma(0)}) & \sigma_{1j_m}^{\Delta}(p_{\varsigma(0)}) & \ldots & \sigma_{nj_m}^{\Delta}(p_{\varsigma(0)}) \\ \sigma_{0j_n}^{\Delta}(p_{\varsigma(1)}) & \sigma_{1j_n}^{\Delta}(p_{\varsigma(1)}) & \ldots & \sigma_{nj_n}^{\Delta}(p_{\varsigma(1)}) \\ \ldots & \ldots & \ldots & \ldots \\ \sigma_{0j_v}^{\Delta}(p_{\varsigma(i)}) & \sigma_{1j_v}^{\Delta}(p_{\varsigma(i)}) & \ldots & \sigma_{nj_v}^{\Delta}(p_{\varsigma(i)}) \\ \ldots & \ldots & \ldots & \ldots \end{bmatrix}}_{\sigma^{\Delta}} \underbrace{\begin{bmatrix} U_k(p_0) \\ U_k(p_1) \\ \ldots \\ U_k(p_i) \\ \ldots \end{bmatrix}}_{U} \tag{17}$$

Where p is a vector whose elements are those places which belong to the optimum trajectory ω *given by* $p_0 \le p_{\varsigma(1)} \le_{U_k} p_{\varsigma(2)} \le_{U_k} \ldots \le_{U_k} p_{\varsigma(n)} \le_{U_k} \ldots$ *which converges to* p^{Δ}.

Consider the game $GPN = (N, P, Q, F, W, M_0, \pi, U)$. Denote for each player $\iota \in N$ and each profile $\sigma_{-\iota} \in \Gamma_{-\iota}$ of strategies of his opponent the set of *best replies*, i.e. the strategies that player ι can not improve upon, and it is defined as follows:

$$B_{\iota}(\sigma_{-\iota}) := \left\{ \sigma_{\iota}^{\Delta} \in \Gamma_{\iota} \mid \forall \sigma_{\iota}^{'} \in \Gamma_{\iota} \; : \; U_{\iota}^{(\sigma_{\iota}^{\Delta}, \sigma_{-\iota})}(p^{\Delta}) \ge U_{\iota}^{(\sigma_{\iota}^{'}, \sigma_{-\iota})}(p) \right\}$$

Definition 6.7 *A strategy profile* σ_{ι}^{Δ} *is a Nash equilibrium point if, for all players* ι $U_{\iota}^{(\sigma_{\iota}^{\Delta}, \sigma_{-\iota}^{\Delta})}(p^{\Delta}) \ge U_{\iota}^{(\sigma_{\iota}^{'}, \sigma_{-\iota}^{\Delta})}(p)$ $\forall \sigma_{\iota}^{'} \in \Gamma_{\iota}$.

Remark 6.3 *It is important to note that in case the strategy is implemented as a chain of transitions* $\ge$ *does not represent a vectorial inequality, the interpretation is obtained from calculating the best reply* B_{ι}.

Definition 6.8 *A strategy* σ *has the fixed point property if it leads to the optimum point* $(U_{\iota}^{(\sigma_{\iota}^{\Delta}, \sigma_{-\iota}^{\Delta})}(p^{\Delta}))$.

Remark 6.4 *From the two previous definitions the following characterization is obtained: A strategy which has the fixed-point property is equivalent to being a Nash equilibrium point.*

Theorem 6.2 *A non-blocking (unless $p \in P$ is an equilibrium point) Game Petri net $GPN = (N, P, Q, F, W, M_0, \pi, U)$ has a strategy σ which has the fixed point property.*

Corollary 6.2 *If in addition to the hypothesis of the theorem the game GPN is finite, the strategy σ leads to an equilibrium point .*

Theorem 6.3 *The optimum point*[2] *coincides with the Nash equilibrium.*

Remark 6.5 *The potential of the previous theorem resides in the simplicity of its formal proof for the existence of an equilibrium point.*

7 Conclusion

In this work, a formal framework for decision-process and game-shortest-path-problem representation has been presented. There are still a number of questions relating classical decision-process theory with game theory, which may in future be addressed satisfactorily within this framework. The traditional notions of stability in the sense of Lyapunov, used to characterize the stability properties of the decision process and game theory, have been explored. We introduce the notion of uniform practical stability and provide sufficient and necessary conditions of stability for the decision process. In addition, we show that the system/mark-dynamic and trajectory-dynamic properties of equilibrium, stability, decision point and equilibrium point converge under some mild restrictions. The Lyapunov method introduces a new equilibrium and stability concept in decision process and game theory. Moreover, we introduce a new type of equilibrium point in the sense of Lyapunov to game theory, lending necessary and sufficient conditions of stability to the game, under certain restrictions. We prove that the equilibrium concept in a Lyapunov sense coincides with the equilibrium concept of Nash, representing an alternative way to calculate the equilibrium and stability of the game. We introduce a novel application area, to the best of our knowledge, in Markov decision process, game theory and Petri nets. An algorithm for finding the equilibrium point has been described. The expressive power and the mathematical formality of the DPPN/GPN contribute to bridging the gap between Petri nets, Markov decision processes and game theory.

References

1. **D. P. Bertsekas** and **S. E. Shreve.** Stochastic Optimal Control: The Discrete Time Case, Academic Press, N.Y., 1978.
2. **D. P. Bertsekas.** Dynamic Programming: Deterministic and Stochastic Models, Prentice--Hall, Englewood Cliffs, N.Y., 1987.
3. **P. Bertsekas** and **J. N. Tsitsiklis.** Parallel and Distributed Computation: Numerical Methods. Prentice--Hall, Englewood Cliffs, N.Y., 1989.
4. **P. Bertsekas** and **J. N. Tsitsiklis.** An analysis of stochastic shortest path problems. --- Mathematics of Operations Research, 16, 3, 580--595, 1991.
5. **Blackwell.** Positive Dynamic Programming. Proc. Fifth Berkeley Sympos. Math. Statist. and Probability (Berkeley, Calif., 1965/66), 1: Statistics, 415--418 Univ. California Press, Berkeley, Calif., 1967.
6. **J. Clempner, J. Medel** and **A. Cârsteanu.** Extending Games with Local and Robust Lyapunov Equilibrium and Stability Condition. International Journal of Pure and Applied Mathematics, 19, 4, 441-454, 2005.

[2] The definition of optimum point is equivalent to the definition of steady-state equilibrium point in the Lyapunov sense given by Kalman in [20].

7. **J. Clempner.** Colored Decision Process Petri Nets: Modeling, Analysis and Stability. International Journal of Applied Mathematics and Computer Science, 15, 3, 405-420, 2005.
8. **J. Clempner.** Modeling Shortest-Path Games with Petri Nets: A Lyapunov Based Theory. International Journal of Applied Mathematics and Computer Science, 16, 3, 387-397, 2006.
9. **J. Clempner.** Towards Modeling The Shortest Path Problem and Games with Petri Nets. Proc. of The Doctoral Consortium at the 27th International Conference on Application and Theory of Petri Nets and Other Models of Concurrency, 1-12, 2006.
10. **C. Derman.** Finite State Markovian Decision Processes, Academic Press, N.Y., 1970.
11. **E. B. Dynkin.** The Optimum Choice of the Instant for Stopping a Markov Process, Soviet Math. Doklady, 150, 238--240, 1963.
12. **J. H. Eaton** and **L. A. Zadeh.** Optimal Pursuit Strategies in Discrete State Probabilistic Systems. Trans. ASME Ser. D, J. Basic Eng., 84, 23--29, 1962.
13. **R. I. Grigelionis** and **A. N. Shiryaev.** On Stefan's Problem and Optimal Stopping Rules for Markov Processes. Theory of Probability and its Applications, 11, 541--558, 1966.
14. **Hernández-Lerma** and **J.B. Lasserre.** Discrete-Time Markov Control Process: Basic Optimality Criteria. --- Berlin, Germany : Springer, 1996.
15. **Hernández-Lerma, G. Carrasco** and **R. Pére-Hernández.** Markov Control Processes with the Expected Total Cost Criterion: Optimality, Stability and Transient Model. Acta Applicadae Matematicae, 59, 3 229-269, 1999.
16. **Hernández-Lerma** and **J.B. Lasserre.** Futher Topics on Discrete-Time Markov Control Process. Berlin, Germany: Springer-Verlag, 1999.
17. **K. Hinderer** and **K. H. Waldmann.** The Critical Discount Factor for Finite Markovian Decision Process With an Absorbing Set. Mathematical Methods of Operation Research, 57, 1-19, 2003.
18. **K. Hinderer** and **K. H. Waldmann.** Algorithms for Countable State Markov Decision Model with an Absorbing Set. SIAM Journal of Control and Optimization, 43, 2109-2131, 2005.
19. **A. Howard.** *Dynamic Programming and Markov Processes*. MIT Press, Cambridge, 1960.
20. **R. E. Kalman** and **J. E. Bertram.** Control System Analysis and Design Via the "Second Method" of Lyapunov. Journal of Basic Engineering, 82(D), 371-393, 1960.
21. **H. J. Kushner** and **S. G. Chamberlain.** Finite State Stochastic Games: Existence Theorems and Computational Procedures, IEEE Transactions on Automatic Control, 14, 3, 1969.
22. **P. R. Kumar** and **T. H. Shiau.** Zero Sum Dynamic Games, in Control and Dynamic Games, (C. T. Leondes, ed.)- --- Academic Press, 1345-1378, 1981.
23. **H. Kushner.** Introduction to Stochastic Control, Holt, Rinehart, and Winston, N.Y., 1971.
24. **Lakshmikantham, S. Leela** and **A.A. Martynyuk,** *Practical Stability of Nonlinear Systems*, World Scientific, Singapore, 1990.
25. **V. Lakshmikantham, V.M. Matrosov** and **S. Sivasundaram,** *Vector Lyapunov Functions and Stability Analysis of Nonlinear Systems,* Kluwer Academic Publ., Dordrecht, 1991.
26. **P. Mandl** and **E. Seneta.** The theory of non-negative matrices in a dynamic programming problem. The Australian Journal of Statistics, 11, 85-96, 1969.
27. **Murata T.** (1989). Petri Nets: Properties, Analysis and Applications. Proceedings of the IEEE, **77**, 4, 541-580.
28. **J. Nash:** Non-cooperative games, *Ann. Math.* 54 (1951), 287-295.
29. **J. Nash:** *Essays on Game Theory*, Elgar, Cheltenham 1996.
30. **J. Nash:** *The essential John Nash*, Eds: H.W. Kuhn and S. Nasar, Princeton UP 2002.
31. **K. M. Passino, K. L. Burguess** and **A. N. Michel.** *Lagrange Stability and Boundedness of Discrete Event Systems,* Journal of Discrete Event Systems: Theory and Applications, 5, 383-403, 1995.
32. **S. D. Patek.** Stochastic Shortest Path Games: Theory and Algorithms. PhD thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, September 1997.
33. **S. D. Patek** and **D. P. Bertsekas.** Stochastic Shortest Path Games. SIAM Journal on Control and Optimization, 37, 3, 804-824, 1999.
34. **S. D. Patek.** On Terminating Markov Decision Processes with a Risk Averse Objective Function. Automatica, 37, 9, 1379-1386, 2001.

35. **R. Pallu de la Barriere.** Optimal Control Theory, Saunders, Phila., 1967.
36. **S. R. Pliska.** On the transient case for Markov decision chains with general state space. ---In: Dynamic Programming and Its Applications, (M.L. Puterman, Ed.). New York: Springer, 335--349, 1978.
37. **S. Poznyak, K. Najim** and **E. Gomez-Ramirez.** Self-learning control of finite Markov chains. Marcel Dekker, Inc., New York, 2000.
38. **U. Rieder.** Bayesian Dynamic Programming. Advances in Applied Probability, 7, 330-348, 1975.
39. **L. S. Shapley.** Stochastic Games, Proceedings of the National Academy of Sciences, Mathematics, 39, 1095-1100, 1953.
40. **N. Shiryaev.** Optimal Stopping Problems. Springer-Verlag, N.Y., 1978.
41. **R. Strauch.** Negative Dynamic Programming. Ann. Math. Statistics, 37, 871--890, 1966.
42. **J. van der Wal,** Stochastic Dynamic Programming, Mathematical Centre Tracts 139, Mathematisch Centrum, Amsterdam 1981.
43. **F., Jr. Veinott.** Discrete Dynamic Programming with Sensitive Discount Optimality Criteria. Ann. Math. Statistics, 40, 5, 1635--1660, 1969.
44. **P. Whittle.** Optimization over Time. Wiley, N.Y., 2, 1983.

Julio Clempner *holds a Ph.D. from the Center for Computing Research at the National Polytechnic Institute. His research interests are focused on justifying and introducing the Lyapunov equilibrium point in shortest-path decision processes and shortest-path game theory. This interest has lead to several streams of research. One stream is on the use of Markov decision processes for formalizing the previous ideas, changing the Bellman's equation by a Lyapunov-like function which is a trajectory-tracking function, and also it is an optimal cost-to-target function for tracking the net. A second stream is on the use of Petri nets as a language for modeling decision process and game theory introducing colors, hierarchy, etc. Petri nets are used for process representation taking advantage of the formal semantic and the graphical display. The final stream examines the possibility to meet modal logic, decision processes and game theory.*

Jesus Medel *holds a PhD from the Center for Research and Advanced Studies (Cinvestav), National Polytechnic Institute. He is a member of the Mexican National System of Researchers (SNI) and a member of the Mexican Academy of Sciences. Since 1999 he is a Professor at the National Polytechnic Institute. At present, he has graduated seven master's and two doctoral students. He has published more than sixty papers and one book, and a second book is in press. In 2004, he was awarded the degree of Magister and PhD in postgraduate education management. His research areas of interest include Filter Theory, Real-time Filter Theory, Non-stationary Sample Time, Real-time Fuzzy Logic Filters, and others.*

***Alin Cârsteanu** is a PhD in Engineering and in Mathematics from the University of Minnesota, USA (1997). His postdoctoral studies bore on Stochastic Hydrology at the National Institute for Scientific Research (INRS-Eau) in Quebec, Canada. He is a member of the Mexican National System of Researchers (SNI), and of several North American and European professional organizations. Since 2000 he has been an Associate, and currently a Professor and Senior Researcher at the Center for Research and Advanced Studies (Cinvestav), National Polytechnic Institute, in Mexico City. Refereed publications and research interests deal with the areas of Time Series Analysis, Wavelet Theory, Stochastic Hydro-meteorology, Atmospheric Precipitation Modeling, and others.*