Multiagent Systems with Hybrid Interacting Dynamics

by

Jorge Luis Piovesan Pedraza

"Licenciado", Electronic Systems Engineering,
Military School of Engineering, La Paz Bolivia, 2002

M.S., Electrical Engineering, University of New Mexico, 2005

DISSERTATION

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

Engineering

The University of New Mexico

Albuquerque, New Mexico

May, 2009

©2009, Jorge Luis Piovesan Pedraza

Dedication

To my wife Milagros and my daughter Natalia, my motivation, inspiration, and emotional support.

To my parents, who raised me with love and gave me the greatest example of hardwork, and family values.

To my brother José and my little sister Pia, my unconditional friends.

To my grandpa Jorge and the memory of my grandma Yolanda, who spoiled me as much as they could.

A mi esposa Milagros y mi hija Natalia, mi motivación, inspiración, y apoyo emocional.

A mis papás, que me criaron con amor y me dieron un gran ejemplo de trabajo duro y valores familiares.

A José y Pia, mis hermanos y amigos incondicionales.

A mi abuelo Jorge, y a la memoria de mi abuela Yolanda, que me consintieron tanto como pudieron.

Acknowledgments

I would like to thank my advisor and mentor Prof. Chaouki Abdallah for all his guidance, friendship, and help. He allowed me to pursue my own ideas, helping me to focus my research efforts on interesting directions without constraining my creativity. He also encouraged and supported my interest on pursuing professional activities beyond the research work, and gave me professional and personal advises whenever I needed one.

I would also like to thank Prof. Herbert Tanner for his countless contributions to this work, and for his personal and professional guidance. He was also a mentor for me. His attention to details helped me improve the quality of my work. He participated in the entire research that resulted in this dissertation and gave me valuable advises that contributed to my personal and professional growth.

I would like to thank Professors Yasamin Mostofi and Rafael Fierro, for their useful comments during the final phases of this work.

To my friends and colleagues at K&A Wireless, LLC, for a great learning experience, and the opportunity of applying my skills on challenges their company faces everyday.

To the faculty and staff at ECE and other departments at UNM for sharing their knowledge and time with students and helping them go through school successfully.

To Milie and Naty for the smile they show me everyday I get home after work, and the encouraging words that made me pick myself up after numerous falls.

To my families in Bolivia, the Piovesan Pedraza and the Fiorilo Nogales, for their constant support, orientation, love, and example of life.

And to my friends in Albuquerque, for the nice moments we all shared.

Multiagent Systems with Hybrid Interacting Dynamics

by

Jorge Luis Piovesan Pedraza

ABSTRACT OF DISSERTATION

Submitted in Partial Fulfillment of the

Requirements for the Degree of

Doctor of Philosophy

Engineering

The University of New Mexico

Albuquerque, New Mexico

May, 2009

Multiagent Systems with Hybrid Interacting Dynamics

by

Jorge Luis Piovesan Pedraza

"Licenciado", Electronic Systems Engineering,

Military School of Engineering, La Paz Bolivia, 2002
M.S., Electrical Engineering, University of New Mexico, 2005

Ph.D., Engineering, University of New Mexico, 2009

Abstract

Multiagent systems with hybrid interacting dynamics are groups of systems whose individual components have hybrid dynamics that affect each other's behavior at both the continuous and discrete levels. These systems are found in a wide range of applications including multivehicle systems, networks of sensors, actuators and embedded control systems, and communication networks. In this dissertation we study the modeling and control of these systems from various different perspectives. Motivated by a control problem in the design of future communication networks, we study a multiagent system whose agents move across a network of discrete locations competing for resources they obtain from such locations. The resources they compete for are continuous while the movements of agents in the network are discrete due to the discrete nature of their environment. The ultimate objective is to be able to control agents and nodes such that their interacting dynamics converge to a point that optimizes the usage of the network's resources. This motivates the formulation of a hybrid dynamical model for agents and nodes to be able to capture the continuous dynamics of the resource allocation, and the discrete dynamics of moving agents and changing network conditions. Additionally we apply resource allocation theory on the design of the continuous dynamics of agents and nodes. We then explore dynamical properties of Interconnected Hybrid Systems (IHS), which is a framework we propose to model general multiagent systems with hybrid interacting dynamics. In particular we obtain conditions for the existence and uniqueness of the executions of IHS and apply these conditions in the design of the general structure of the discrete dynamics of the agents in the resource allocation problem. Then we explore the application of randomized optimization algorithms to the optimal control of individual and multiagent hybrid systems. In the case of individual hybrid systems we compare the randomized approach to existing gradient based techniques obtaining comparable performance to this model-based technique. In the multiagent case we apply the same algorithm to the resource allocation problem, achieving the initial design objective of controlling the movements of agents towards a configuration that optimizes the usage of resources in the network. Finally, we explore the use of abstractions for the model simplification of individual hybrid systems, and discuss its applicability to the resource allocation problem. The abstraction we propose substitutes the continuous dynamics of a hybrid system for clocks, creating a timed automaton that is simpler than the original system but retains the reachability and stability properties of the original system, and keeps track of the time that takes the original system to converge to an equilibrium set.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation and Current Work

Advances in computation and communication technologies provide interesting possibilities for substituting complex and expensive devices, with more cost-effective distributed systems of multiple simple devices. The advantage of these distributed systems is that they can perform complicated tasks by generating a group behavior through the coordination of the agent's actions, usually using only local policies and information that is available through limited communication and sensing. This results in greater efficiency and robustness of the system, compared to that of a single device. However, the control of multiple entities with common goals raises new challenges that include, but are not limited to, the design of local policies that enable a stable (and maybe optimal) group behavior, the reliable information sharing under communication constraints, and consensus among agents with potentially different measurements.

There exists extensive literature on multi-agent systems. One major thrust of the research on cooperative control is that of safe group navigation. Different approaches have been proposed to address this problem. They include formation control [38,42,46,70,80], flocking and swarming algorithms [20, 27, 69] and platooning [31, 64]. Another frequently discussed problem is that of positioning agents in a given environment. Results in this direction include facility location via distributed optimization [17], perimeter tracking [14], formations using implicit functions [13], and coordination using Internet-like protocols [58]. Researchers have also studied general consensus problems [41,44,45,57], sensor network applications [21,43], and distributed task assignment using load balancing schemes [19].

Most of the surveyed literature on multi-agent systems deals with agents that have continuous dynamical descriptions. Some of these results even consider switching dynamics in the communication topology [41, 44, 45, 57, 69]. However very few researchers have addressed systems that involve agent models with discrete dynamics that are related to events or switching modes of operation [19, 32, 39]. This type of dynamics may be found in multi-vehicle applications where the individual vehicles are capable of changing operating conditions (switching from pursuing an opponent to evading it), or in multi-agent systems where agents change their location in discrete form (moving from one room to another).

In particular, the motivation for this work comes from a novel Internet architecture designed to deal with retransmissions, delays, and communication failures that occur when a mobile device moves across multiple networks or has an intermittent network connection. These problems arise because the current implementation of the Internet delivers packets to static locations, assumption that is no longer valid when the communication task involves mobile devices because such devices may connect/disconnect without prior notice, and may even move across different networks during the communication task [28].

This architecture postulates an abstract network that treats the nodes and the traffic as digital entities, separating the functional components of the network from the hardware that enables them. In this fashion, a logical (intelligent) network is created on top of the physical network, enabling functions like persistent identification of different objects and smarter routing to avoid retransmissions. The functional components of the network are conceptualized as software agents (e.g. routing agents, storage agents, DNS implementation agents, etc.), and the hardware is seen as a resource to be used by these agents, such that their tasks can be efficiently executed (Figure 1.1).

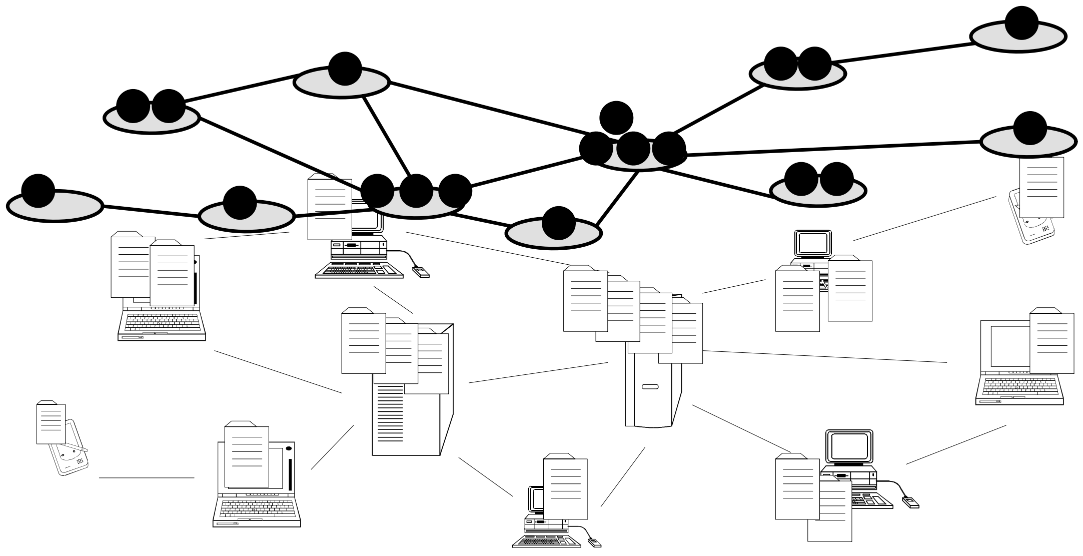

Figure 1.1: Network example: Each platform in the network can run several processes concurrently. The network is abstracted as a graph and the processes as agents (black circles) that can move among the nodes.

In this setting, each particular node in the physical network is capable of simultaneously hosting more than one agent. Moreover, agents can move around searching for nodes with more resources in order to complete their tasks in the best possible way. Agents are viewed as greedy entities competing with each other for the completion of their tasks (benefit), by consuming hardware resources required for the task execution. The objective is to use the resources of the network so that the aggregate benefit of all the agents in the network is maximized.

This challenge sets the stage for an optimization problem that, due to the nature of the network, must be solved in a decentralized form by a multi-agent system. Each node needs to distribute its resources among the agents that it hosts. Agents need to decide whether they must migrate to a different node if this action increases their benefit. These actions must be executed without a centralized authority controlling the whole network, meaning that the desired solution of this problem should be in the form of local and semi-local policies (based on information about the node currently occupied and its immediate neighbors).

So the multi-agent system that motivates this dissertation is a set of heterogeneous agents whose goal is to optimize a utility function via the utilization of resources available in the environment. The environment is composed of discrete locations connected by paths used by the agents to locate resources at such locations. Different locations may have different types and amounts of available resources, and each location (a node) allocates its resources according to requests from its resident agents. The agents request resources according to their particular tasks, which are encoded in their utility functions. The resources are allowed to vary in discrete form and according to environment related events. The agents are therefore capable of two types of decisions: requesting more (or less) of a resource from a location they already occupy, and moving from one location to another in order to obtain more resources. The ultimate goal of the cooperative system is to optimize the aggregate of the agent's utility functions using only local policies i.e., to control decisions at the agent level, such that the usage of the environment resources is globally optimized by the multi-agent system.

1.2 Contributions

In this dissertation we approach the continuous resource allocation problem (RAP) for agents moving on a discrete environment from several different directions. We formulate an optimization problem that expresses the design objective of the RAP, and propose a hybrid framework to design the dynamics of agents and nodes in the system. We also propose a model for general multiple interacting hybrid systems, and study their basic dynamical properties (existence and uniqueness of executions). Motivated by the complexity of these models, we explore the application of randomized optimization techniques to individual hybrid systems, and study the application of this result on the RAP using simulations. Finally, with complexity as motivation again, we formulate an asymptotic abstraction technique for individual hybrid systems, and discuss how it can be applied on the simplification of the dynamic model of the RAP.

In order to understand the RAP, we start this dissertation by formulating optimization problem that expresses the design objective of the problem. This optimization turns out to be a mixed-integer nonlinear programming problem that grows exponentially with the number of nodes in the network. This problem is however decomposed into two hierarchical optimization problems, a global integer optimization related to the position of the agent in the network, and a local nonlinear optimization related to the resource allocation of each node's resources among the agents that occupy such node. The key assumption that results in this decomposition restrict the agents to access resources in the nodes they occupy only.

The hybrid framework we propose for the RAP describes each agent and each location as a hybrid system [12]. This allows us to capture the continuous dynamics of the allocation of resources among different agents, and the discrete dynamics of the movement of agents among different locations as well as changes in the environment. With this framework we intend to design all the components of the agents/location hybrid models. In particular, we apply Internet congestion control like algorithms [1, 29, 62, 77], to design stable continuous dynamics for all agents and nodes in the network.

The model for general multiple interacting hybrid systems consists of a set of hybrid systems where interactions occur at both the continuous and discrete levels. The continuous interactions are achieved by making each agent's continuous input a function of the continuous states of its neighbors in similar form as it is done in current multi-agent system literature surveyed above. The discrete interactions are achieved through the transition guards of each agents: a discrete transition on a single agent depends on the continuous states of this agent and the hybrid states of its neighbors. Using this model we recast various hybrid systems concepts to the multi-agent case including reachability and hybrid execution, and study existence and uniqueness properties of the multi-agent executions extending the work in [37]. The main result in this part is a set of conditions for the existence and uniqueness of the execution of a multi-agent system that are expressed in terms of the local models of each agent composing the system. In this form, the global behavior of the multi-agent system can be predicted by studying the dynamic description of the individual composing agents. This part is concluded with the application of the theoretical results to the RAP.

Motivated by the complexity of both models above, we explore the application of randomized algorithms [72, 76] to the optimal control of individual hybrid systems and the optimal design of the discrete dynamics of agents and nodes in the RAP. Randomized algorithms are attractive for complicated problems because they are model-free techniques, and therefore do not depend on the complexity of the system description. In this arena we obtained and evaluated a general randomized algorithm for optimal control of individual hybrid systems. The algorithm is theoretically simple and independent of the model's complexity. The performance of this algorithm is comparable to that obtained using gradient based algorithms [4, 10, 18]. However the algorithms can be used for more general cases than those considered in [4, 10, 18]. We applied the same technique to the design of the discrete dynamics of agents and nodes in the RAP. The discrete dynamics were designed based on the conditions for existence and uniqueness of Interconnected Hybrid Executions, and then optimized using the randomized approach. The numerical results we obtained suggest that the optimization algorithm makes the agent move among nodes in the network seeking better resources, and ultimately optimizing the performance of the whole network.

Finally, we propose an asymptotic abstraction for hybrid systems that simplify their reachability and stability analysis tasks. Abstraction techniques [2, 47, 74] can be thought of as selective retention of information pertinent to a specific task or objective. In this dissertation we first explore the obtention of a discrete asymptotic abstraction for a individual hybrid systems. In this abstraction, continuous dynamics are discarded completely and substituted for information about the time the system takes to reach an equilibrium point. This simplifies the reachability and stability analysis for complicated hybrid systems. We then discuss the application of this technique to the simplification of the dynamic models of agents and nodes relying on the stable design obtained using Internet congestion control like algorithms [1, 29, 62, 77].

The contributions of this dissertation can be summarized as follows:

- An optimization problem formulation for the design objective of the continuous resource allocation among agents moving on a discrete environment.

- A hybrid framework for the design of the dynamic descriptions for agents and nodes in the RAP.

- A general model for multiple interacting hybrid systems that allows us to study their dynamical properties.

- The application of randomized optimization techniques to the optimal control of (individual) hybrid systems and its subsequent application to the RAP.

- The formulation of a discrete asymptotic abstraction for (individual) hybrid systems and a discussion of its applicability to the RAP.

1.3 Document Organization

The remainder of this document is organized as follows: Chapter 2 presents relevant hybrid concepts used along this dissertation. Chapter 3 presents the optimization formulation and the hybrid modeling framework for the multi-agent resource allocation problem described above. Chapter 4 present the general model for multiple interacting hybrid systems and the conditions for existence and uniqueness of their executions. Chapter 5 presents the results on the application of randomized algorithm for optimal control of individual hybrid systems and the optimal design of the agents and nodes' dynamics in the RAP while Chapter 6 describes our work on abstractions for hybrid systems. Finally Chapter 7 outlines our conclusions.

Chapter 2

Hybrid Systems' Concepts

We now introduce concepts from hybrid systems theory that will be used along the rest of this document. We provide formal definitions for a controlled hybrid dynamical system extracted from [12], and for concepts such as hybrid time trajectory, hybrid input, hybrid state trajectory, and hybrid executions. We consider a general description for a hybrid system that includes autonomous and controlled transitions, as well as continuous and discrete inputs.

Definition 2.1 (Controled hybrid dynamical system) *Let a* Controlled Hybrid Dynamical System (CHDS) *[12] be a tuple* $\mathbf{H} = [Q, \mathbf{\Sigma}, \mathbf{G}, \mathbf{Z}, \mathbf{S}]$ *where:*

- *Q is the set of discrete states.*
- *$\mathbf{\Sigma} = \{\Sigma_q\}_{q \in Q}$ where $\Sigma_q = (X_q, f_q, U_q, \mathbb{R}^+)$ is a dynamical system that corresponds to $q \in Q$ with X_q being the continuous state space, f_q the continuous dynamics, U_q the set of continuous controls, and $\mathbb{R}^+ = [0, \infty)$ the time set.*
- *$\mathbf{S} = \{S_q\}_{q \in Q}$ is the set of discrete transition labels of $\mathbf{H}$. Symbol $s_q \in S_q$ determines the discrete state after a transition from $q \in Q$. There exist two types of transitions:*

Transitions triggered by external events or control inputs (called controlled transitions) and transitions that are triggered by functions of the state of the system (called autonomous transitions).

- $\mathbf{G} = \{G_q\}_{q \in Q}$ *is the set of guard conditions for* $\mathbf{H}$. G_q *is a map that determines when a transition from* $q \in Q$ *is possible. A guard condition for a controlled transition is denoted as* G_q^{C}. *This guard must satisfy a condition on the state of the system and on the existence of an event or control input i.e,* $G_q^{\mathrm{C}} : S_q \rightarrow E \times X_q$ *where* E *is the set of possible events or discrete control inputs of* $\mathbf{H}$. *A guard condition for an autonomous transition, denoted* $G_{q_i}^{\mathrm{A}}$ *needs to satisfy a condition on the state of the system only i.e,* $G_q^{\mathrm{A}} : S_q \rightarrow X_q$.

- $\mathbf{Z} = \{Z_q\}_{q \in Q}$ *is the set of transition maps of* $\mathbf{H}$, *where* $Z_q : G_q \times S_q \rightarrow \bigcup_{p \in Q}\{X_p\}$ *determines the continuous state of* $\mathbf{H}$ *after a transition* $s_q \in S_q$.

Finally, $H = \left(\bigcup_{q \in Q} X_q\right) \times Q$ *is the hybrid state space of* $\mathbf{H}$. *Note that* $\mathbf{S}$ *may include the no transition element* $\{id\}$.

The concepts we define below allow us to describe the evolution of a hybrid dynamical system. The time trajectory provides the time information needed to detail the continuous and discrete components of the hybrid evolution. The state trajectory contains information about the evolution of the hybrid state of the system, while the hybrid input stores the continuous and discrete inputs history. Finally the hybrid execution gathers all this information in a single sequence that satisfies minimum conditions for the system to be correctly specified.

A *Hybrid Time Trajectory* [36,60] is a sequence $\tau := \{\tau^0, \tau^1, \dots, \tau^N\}$ such that $\tau^n \leq \tau^{n+1}$ for all $n \in \{0, 1, \dots, N-1\}$, where $1)$ $\tau^0 \in \tau$ is the time when $\mathbf{H}$ starts its evolution, $2)$ $\tau^n \in \tau$ is the time at which $\mathbf{H}$ makes a discrete transition from q^n to q^{n+1} for $n = \{0, 1, \dots, N-1\}$, and $3)$ $\tau^N \in \tau$ is the time when $\mathbf{H}$ ends its evolution. τ is infinite if

$N = \infty$ and is finite otherwise. Let the set $\langle\tau\rangle \triangleq \{1, 2, ..., N\}$ if N is finite and $\{1, 2, ...\}$ if $N = \infty$. We also define $|\tau| = \sum_{n\in\langle\tau\rangle\setminus N}(\tau^{n+1} - \tau^n)$.

Definition 2.2 (Hybrid State Trajectory [60]) *A hybrid state trajectory $\mathcal{H} = (\tau, \mathrm{q}, \mathrm{x})$ consists of hybrid time trajectory τ, a sequence of discrete states $\mathrm{q} = \{q^0, q^1, \ldots, q^N\}$ and a sequence of differentiable maps $\mathrm{x} = \{x^0, x^1, \ldots, x^N\}$, where q^0 is the initial discrete state, $x^0 = x^0(0)$ is the initial continuous state, $q^n : [\tau^{n-1}, \tau^n) \to Q$ is a constant map for all $n \in \{1, \ldots, N\}$, and $x^n = x^n(t) : [\tau^{n-1}, \tau^n) \to X_{q^n}$ is a differentiable function for all $n \in \{1, \ldots, N\}$ and all $t \in [\tau^{n-1}, \tau^n)$.*

A *Hybrid Switching Sequence* [36, 60] is a sequence $(\tau, \mathrm{s}) = \{(\tau^0, s^0), (\tau^1, s^1), \ldots, (\tau^N, s^N)\}$ of pairs of switching times τ^n and discrete (autonomous or controlled) transition labels $s^n \in \mathbf{S}$, where s^n is the transition from q^n to q^{n+1} that takes place at the time τ^n for all $n \in \{0, 1, \ldots, N-1\}$. Note that τ is a hybrid time trajectory, and s is called a location schedule.

Definition 2.3 (Hybrid Input [60]) *A hybrid input $\mathcal{I} := (\tau, \mathrm{s}, \mathrm{u})$ consists of a hybrid switching sequence (τ, s) and a piecewise continuous input $\mathrm{u} = \{u^1, u^2, \ldots, u^N\}$, where $u^n(t) \in U_{q^n}$ for all $n \in \{1, \ldots, N\}$ and all $t \in [\tau^{n-1}, \tau^n)$.*

Definition 2.4 (Hybrid Execution [36, 60]) *A hybrid execution of a CHDS is a collection $\chi(h_0) = (\tau, \mathrm{q}, \mathrm{s}, \mathrm{x}, \mathrm{u})$, composed by a hybrid input $\mathcal{I}$ and a hybrid state trajectory $\mathcal{H}$ such that $(\tau, \mathrm{s}, \mathrm{u}, \mathrm{q}, \mathrm{x})$ satisfies:*

- $h_0 = (q^0, x^0(0))$ *is an initial condition of* $\mathbf{H}$
- *Continuous Dynamics: For all $t \in [\tau^{n-1}, \tau^n)$, $\dot{x}^n(t) = f_{q^n}\big(x^n, u^n, t\big)$ and $x^n(t) \in X_{q^n}$ for all $n \in \{1, 2, \ldots, N\}$;*
- *Discrete Dynamics (Autonomous or Controlled): For all $n \in \{0, 1, \ldots, N-1\}$, $q^{n+1} = s^n \in S_{q^n}$, $x^n(\tau^n) \in G^*_{q^n}$, and $\big(q^{n+1}, x^{n+1}(\tau^n)\big) \in Z_{q^n}$.*

Where $(\cdot)^*$ denotes $(\cdot)^A$ or $(\cdot)^C$ depending on the type of discrete transition (autonomous or controlled respectively).

$\chi^S(h_0)$ denotes the set of all executions with initial condition h_0, and similarly $\chi^F(h_0)$ denotes the set of all finite executions and $\chi^\infty(h_0)$ denotes the set of all infinite executions with initial condition h_0. Init denotes the set of all initial conditions. We say that $\chi(h_0) = (\tau, \mathrm{q}, \mathrm{s}, \mathrm{x}, \mathrm{u}) \in \chi^F(h_0)$ maps h_0 to $\vec{h}$ if $\tau = \{\tau^0, \tau^1, \ldots, \tau^N\}$ and $h = (q^N, x^N(\tau^N))$.

Definition 2.5 (Reachable Set [36]) *A hybrid state $h \in Reach(h_0)$ if there exists a finite execution $\chi(h_0) \in \chi^F(h_0)$ that maps h_0 to h. The set of states h that can be reached from any initial condition $Reach_{\mathbf{H}} = \bigcup_{h_0 \in \mathrm{Init}} Reach(h_0)$ is called* Reachable set of $\mathbf{H}$.

We finalize this section noting that a hybrid system can be represented by a directed graph [36], such that each discrete mode in Q is mapped to a vertex, which will contain the label of the mode, its state space and its continuous flow equation. Similarly, each edge, that represents a discrete transition label will have a guard and a reset function attached to it (For an example see Figure 2.1). The directed graph related to the hybrid system $\mathbf{H}$ will be denoted as $\mathcal{G}_{\mathbf{H}}$.

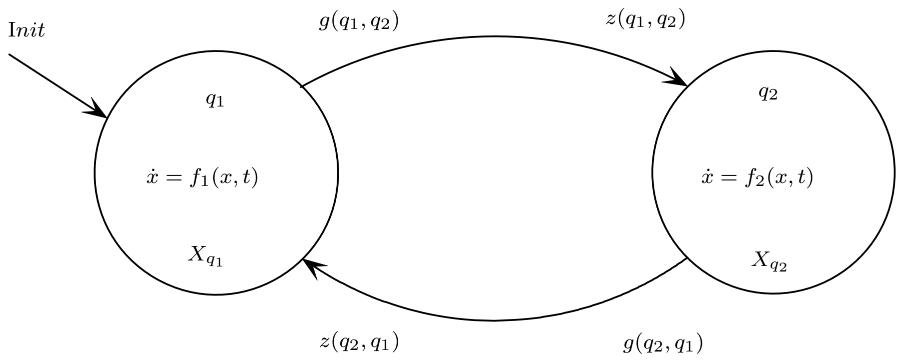

Figure 2.1: Example of a hybrid system represented using a graph.

Chapter 3

Continuous Resource Allocation among Multiple Agents Moving on Discrete Environments

3.1 Introduction

Motivated by the network design problem described in Chapter 1, we consider in this chapter the problem of controlling a multi-agent system whose agents move across discrete locations. The agents attempt to extract resources from the environment while the environment, which may vary as the system evolves, distributes its resources according to the agents requests. The environment is modeled as a network of discrete nodes. Our ultimate goal, as required by the network design problem in Chapter 1, is to design the dynamical policies that determine the behavior of agents and nodes, such that the usage of resources in the network is optimized. We propose a hybrid model to describe both agents and nodes. Several components of this model are design variables that may be obtained analytically. We then formulate an optimization problem that may be decomposed into two hierarchical optimization problems: An integer optimization problem that considers the distribution of agents among the nodes of the network, and a convex optimization problem within each node that corresponds to the distribution of resources of each node among its resident agents. We show that the optimization problem within each node is a special case of the formulation that models congestion control algorithms in the Internet. We then use available results to solve a portion of the proposed hybrid description for agents and nodes. Moreover, we show that the resulting continuous dynamics are globally asymptotically stable, with their equilibrium point coinciding with the solution of the optimization problem. As a consequence, the proposed continuous dynamics yield an interconnected system that is stable on each possible configuration of agents and nodes.

Our problem is concerned with a set of heterogeneous agents whose goal is to optimize a utility function via the utilization of resources available in the environment. The environment is composed of discrete locations connected by paths used by the agents to locate resources at such locations. Different locations may have different types and amounts of available resources, and each location (a node) allocates its resources according to requests from its resident agents. The agents request resources according to their particular tasks, which are encoded in their utility functions. The resources are allowed to vary in discrete form and according to environment related events. The agents are therefore capable of two types of decisions: requesting more (or less) of a resource from a location they already occupy, and moving from one location to another in order to obtain more resources. The ultimate goal of the cooperative system is to optimize the aggregate of the agent's utility functions using only local policies i.e., to control decisions at the agent's level, such that the usage of the environment resources is globally optimized by the multi-agent system.

In Chapter 1 we surveyed the state of the art on multi-agents systems. Most of the available results on this area consider agents with almost exclusive continuous dynamics [13,14,17,20,27,31,38,41,42,44–46,57,58,64,69,70,80], limiting the discrete dynamics to switching in the communication topologies. Very few researchers have considered discrete event dynamics in the study of multi-agent systems [19].

The problem we address in this chapter has an important difference compared to the cited results: we consider agents that participate in a continuous resource allocation problem moving among discrete locations. Therefore, the agents' description have to incorporate both continuous time and discrete event dynamics. The environment is modeled as a graph where the nodes represent the discrete locations and the edges represent the paths that the agents use to obtain information about other nodes and to move between locations. The agents and nodes are modeled as hybrid systems. The general hybrid model we use allows us to capture both the continuous evolution of the resource allocation tasks (node and agent dynamics), and the discrete events related to the changes on resource availability in the locations (node dynamics) and the movement of agents among the locations in the environment (agent dynamics).

Since the final goal is to optimize the usage of the network resources, we formulate an optimization problem, which turns out to be a mixed integer nonlinear optimization problem. We then obtain an equivalent hierarchical optimization problem composed of a (global) network integer optimization problem at the higher level of the hierarchy, and several decentralized (one for each node in the network) convex optimization problems at the lower level. Since the convex optimization problem within each node is a special case of the optimization problem used to model various Internet congestion control algorithms [29, 35, 62] we perform several simplifications to the design of the agents and nodes dynamics. First, the continuous dynamics are designed separately from the discrete dynamics. Moreover, the continuous dynamics for both the agents and nodes are designed based on the dynamic model for the Internet congestion control algorithms [1, 35, 62, 77]. This implies that there is a globally asymptotically stable (continuous) equilibrium point for each possible (discrete) agent distribution in the network, and that these equilibria coincide with the solution of the optimization problem for that particular distribution. It is important to note that the results in this chapter provide a complete design for the continuous dynamics of agents and nodes, but leave open the design of the discrete transition rules. In Section 3.7 we discuss the possible solution approach to this important component of the system.

The optimization problem formulated in this paper may be solved using model-based techniques, such as combinatorial optimization [16], or mixed integer programming [25] approaches. These techniques, however, present a major problem in terms of computational complexity, since they are NP-Complete in the number of integer variables [59] which in our case may become very large. Instead, we propose a hierarchical approach exploiting the structure of the problem, which allows us to decentralize the continuous decision variables, leaving only the discrete ones for centralized optimization. While this hierarchical solution is similar in spirit to the dual decomposition approach proposed in [78], the discrete component of our problem makes it different from that handled by dual decomposition [78], which only considers continuous decision variables.

3.2 Hybrid Model and Design Objective

3.2.1 Problem description

A set of agents is moving on an environment composed of discrete locations. Each location (node) has different types and amounts of resources that may be allocated to the agents, while the agents use such resources for the completion of different tasks. The agents are greedy entities competing for the resources in the network, which means that each agent attempts to maximize its usage of resources considering only its own benefit. Agents are capable of requesting resources from the node that hosts them, as well as migrating to different nodes in the network seeking resources to complete their task. Task satisfaction is measured by a utility function that provides a real value as a function of the resources that the agent uses.

Each node distributes the resources among the agents it hosts, according to the requests of these agents. The nodes may, however, experience changes in their resource amounts. The paths (edges) that connect different locations are used by the agents to move between nodes and/or to obtain information about resources in nearby locations, may change over time. The final objective is to design the nodes' and agents' dynamics such that the usage of resources in the environment (network) is optimized with respect to the requirements of the agents. A pictorial representation of the problem is shown in Figure 3.1. We impose the assumptions below:

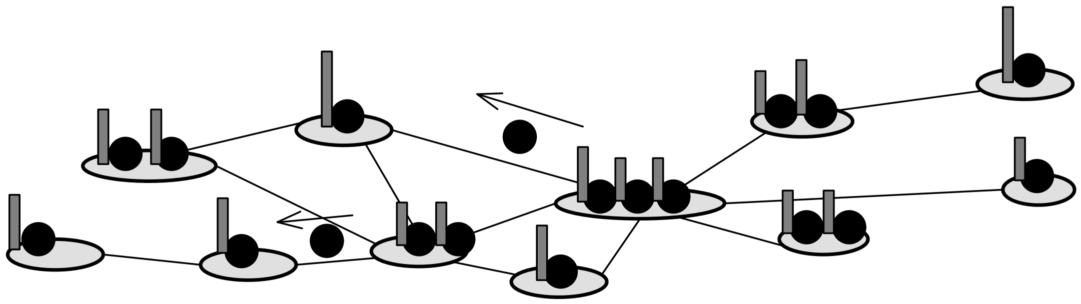

Figure 3.1: Multi-agent system example: Each location in the network distributes its local resources among its residing agents. The locations are abstracted as nodes in a graph (gray ovals), the paths available for movement of agents and communication of states between different nodes are represented by edges in the graph. The agents are represented with black circles and the resources they use by gray bars. Agents move between nodes (identified with arrows on top) expecting better resources at a destination node

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be a *graph* with *nodes* indexed by $\mathcal{V} = \{1, 2, ..., N_v\}$ and *edges* $\mathcal{E} = \{(v, w) : v, w \in \mathcal{V}, v \neq w, \text{ and } v \text{ connected to } w\}$. We call the graph *undirected* if $(v, w) \in \mathcal{E}$ whenever $(w, v) \in \mathcal{E}$. A graph is *connected* if there is a path between any pair of nodes in the graph, where a *path* from v to w is a sequence of different nodes starting at v and ending at w such that consecutive nodes are connected. We call neighborhood of v to the set $\mathcal{N}_v = \{w \in \mathcal{V} : (v, w) \in \mathcal{E}\}$.

Assumption 3.1 (Network) *The network is an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ where $N_v = |\mathcal{V}|$ is constant.*

Assumption 3.2 (Resources) *There exist N_r types of resources in the network indexed by the set $\mathcal{R} = \{1, 2, ..., N_r\}$, where N_r is fixed. For each node $i \in \mathcal{V}$ the vector of available amount of resources to be allocated among the agents located on node $i \in \mathcal{V}$ is denoted by $R_i = (r_{i,1}, r_{i,2}, ...r_{i,N_r})$, where $r_{i,j} \in \mathbb{R}$ is the amount of resource of type $j \in \mathcal{R}$ available at node $i \in \mathcal{V}$. We assume that $r_{i,j}$ may vary iver time for all $i \in \mathcal{V}$ and for all $j \in \mathcal{R}$, taking on values from a finite set $\Xi \subset \mathbb{R}$ according to the dynamics specified in Definition 3.2.*

Assumption 3.3 (Agents) *There is a fixed number of agents N_a, indexed by the set $\mathcal{A} = \{1, 2, ..., N_a\}$. The state of each agent $k \in \mathcal{A}$ is an ordered tuple $(x_{k,1}, x_{k,2}, ..., x_{k,N_r}, q_k)$, where $x_{k,j} \in \mathbb{R}$ represents the amount of resource of type $j \in \mathcal{R}$ allocated to agent $k \in \mathcal{A}$, and $q_k \in \mathcal{V}$ denotes the location of agent $k \in \mathcal{A}$ in the network. Note that $0 \leq x_{k,j} < \infty$ for all $k \in \mathcal{A}$ and for all $j \in \mathcal{R}$.*

If Assumption 3.1 is relaxed, the number of nodes may change over time and it may be possible to have asymmetric communication and agent movement capabilities between nodes. If Assumption 3.3 is relaxed, we may allow variations in the number of agents over time. Note however that Assumption 3.2 is strongly related to Assumption 3.1 because if $0 \in \Xi$, then $R_i = \vec{0}$ emulates the disappearance of node i from the network. The assumption that R_i varies over time may then be used to represent the appearance or disappearance of nodes in the network.

Practical problems modeled under this framework could potentially generate a large set of resources' types to be allocated, but this set will still be finite. Agent satisfaction depends upon their location in the network and the resources allocated to them, so the relevant information for the agents is contained in the state description introduced in Assumption 3.3.

Also note that if a dwell time exists between changes in the sets of agents and nodes, Assumptions 3.1-3.3 would still provide a valid description of the system in between these changes. We now consider node's neighborhoods in the network as follows:

Definition 3.1 (Neighborhoods of a node) *Let the Node-Neighborhood of node $i \in \mathcal{V}$ be $\mathcal{N}_{i\in\mathcal{V}} = \{w \in \mathcal{V} : (i, w) \in \mathcal{E}\}$ i.e., the Node-Neighborhood of a node is composed by the nodes that are connected to it by an edge in the network. Let V_i be the set of agents located at node $i \in \mathcal{V}$, i.e. $V_i = \{k \in \mathcal{A} : q_k = i\}$, which we also call Agent-Neighborhood of node i.*

3.2.2 System's dynamics

In order to capture the complete behavior of agents and nodes in the network, we model each of them as a hybrid dynamical system. This framework seems to be appropriate for the problem because both agents and nodes present continuous and discrete possible behaviors: Agents competing for resources within a given node can be modelled using continuous dynamics, while agents jumping within different locations in the network must be modelled by discrete dynamics. Similarly when nodes are assigning different amounts of resources to the agents they may be modelled using continuous dynamics, while nodes switching between modes of distribution that depend on the number of agents residing in the node have to be modelled using discrete dynamics. We use the controlled hybrid dynamical system from Definition 2.1.

Let $\mathbb{N}^{N_a} = \{\alpha \in \mathbb{N} : \alpha \leq |\mathcal{A}| = N_a\}$, i.e. the set natural numbers smaller than the cardinality of $\mathcal{A}$. In what follows we use subindexes to denote dependence of variable on sets of nodes, agents and/or resources, e.g. α_i with $i \in \mathcal{V}$ denotes dependence of α on the set of nodes, while $\alpha_{k,j}$ with $k \in \mathcal{A}$ and $j \in \mathcal{R}$ denotes dependence of α on the set of agents and resources. We use the notation $(\alpha_n)_{n\in S}$ with $S = \{1, 2, ..., |S|\}$ to denote the vector $(\alpha_1, \alpha_2, ..., \alpha_{|S|})$.

Definition 3.2 (Node dynamics) *Each node $i \in \mathcal{V}$ is described as a Controlled Hybrid Dynamical System $\mathbf{H}_i = [Q_i, \mathbf{\Sigma}_i, \mathbf{G}_i, \mathbf{Z}_i, \mathbf{S}_i]$ that satisfies the following conditions:*

- *There exists one $q_i \in Q_i$ for each $(R_i, c_i) \in \Xi^{N_r} \times \mathbb{N}^{N_a}$ for all $i \in \mathcal{V}$, where c_i represents the number of agents that occupy node $i \in \mathcal{V}$. Note that Q_i is guaranteed to be finite.*

- *The set of continuous dynamics is defined as $\mathbf{\Sigma}_i = \{\Sigma_{q_i}\}_{q_i \in Q_i}$. The continuous dynamics Σ_{q_i} are to be designed subject to the following conditions: $X_{q_i} = X_i$ for all $q_i \in Q_i$ (the continuous state space is the same for all discrete modes), and $u_{q_i} : \prod_{k \in V_i} H_k \rightarrow U_{q_i}$ for all $u_{q_i} \in U_{q_i}$ for all $q_i \in Q_i$ (the continuous controls of a node are functions of the states of the agents located at that node).*

- *The sets $\mathbf{G}_i,\ \mathbf{Z}_i,\ \mathbf{S}_i$ are defined as: $\mathbf{G}_i = \{G_{q_i}\}_{q_i \in Q_i}$, $\mathbf{S}_i = \{S_{q_i}\}_{q_i \in Q_i}$, and $\mathbf{Z}_i = \{Z_{q_i}\}_{q_i \in Q_i}$.*

- *The discrete transitions are purely controlled, and triggered by the occurrence of an external event. Therefore the discrete transition guard is a condition on the occurrence of an event and on the state of the node dynamical description $G_{q_i} : S_{q_i} \rightarrow E_i \times X_{q_i}$ for all S_{q_i} for all $q_i \in Q_i$, where E_i is the set of possible event for H_i.*

- *E_i contains two types of events: 1) Changes in the resources R_i for node i, and 2) Changes in the sets $\mathcal{N}_i$ or V_i (changes in other nodes connected to the node or in the agents hosted by the nodes).*

Definition 3.3 (Agent dynamics) *Each agent $k \in \mathcal{A}$ is described as a Controlled Hybrid Dynamical System $\mathbf{H}_k = [Q_k, \mathbf{\Sigma}_k, \mathbf{G}_k, \mathbf{Z}_k, \mathbf{S}_k]$ that satisfies the following conditions:*

- *There exists one $q_k \in Q_k$ for each $i \in \mathcal{V}$. Note that Q_k is guaranteed to be finite.*

- *The set of continuous dynamics is defined as $\mathbf{\Sigma}_k = \{\Sigma_{q_k}\}_{q_k \in Q_k}$. The continuous dynamics Σ_{q_k} are to be designed subject to the following restrictions: $X_{q_k} = X_k$ for all $q_k \in Q_k$ (the continuous state space is the same for all discrete modes), and $u_{q_k} : H_{q_i} \rightarrow U_{q_k}$ s.t. $i = q_k$ for all $u_{q_k} \in U_{q_k}$ for all $q_k \in Q_k$ (the continuous controls* *are functions of the state of the node that agent k occupies). Note in Assumption 3.3 that the continuous part of the state of the agent $(x_{k,1}, x_{k,2}, ..., x_{k,N_r}) \in X_k$.*

- *The sets $\mathbf{G}_k$, $\mathbf{Z}_k$, $\mathbf{S}_k$ are defined as: $\mathbf{G}_k = \{G_{q_k}\}_{q_k \in Q_k}$, $\mathbf{S}_k = \{S_{q_k}\}_{q_k \in Q_k}$, and $\mathbf{Z}_k = \{Z_{q_k}\}_{q_k \in Q_k}$.*

- *The discrete transitions are purely controlled and triggered by the occurrence of an external event. Therefore the discrete transition guard is a condition on the occurrence of an event and on the state of the node dynamical description $G_{q_k} : S_{q_k} \to E_k \times X_{q_k}$ for all S_{q_k} for all $q_k \in Q_k$, where E_k is the set of possible events for H_k.*

- *E_k is a set of logic valued functions for all $k \in \mathcal{A}$. The functions are left unspecified at this moment because of being part of the design parameters. If the output of $e_k \in E_k$ is true, then an event is generated, otherwise no event is generated.*

Definitions 3.2 and 3.3 describe the dynamic behavior of the nodes and the agents. The *dynamics of a node* evolve as follows: Given an initial hybrid condition – a discrete and a continuous state: (q_i, x_{q_i}) for $i \in \mathcal{V}$ – the continuous state evolves according to the active continuous dynamics (Σ_{q_i}) until the occurrence of a discrete event, caused by a change in the resources of the node R_i or by the arrival (departure) of an agent to (from) the node. This event causes the system to change to a new discrete state q_i', where the evolution of the system continues according to the new continuous dynamics $\Sigma_{q_i'}$.

The *dynamics of an agent* evolve in a similar fashion to those of a node, with the following caveats: The discrete states in the hybrid model of the agent represent the nodes in the network that may host the agent. Similarly, the discrete transitions represent the migration of an agent to a different node. Thus the events E_k of an agent $k \in \mathcal{A}$, which are discrete valued functions, must be designed to allow the agent to choose the best node in the network as a function of its requirements. Finally, note that the *interactions between nodes and agents* happen at both the continuous and discrete levels: 1) The continuous input of the nodes dynamics are functions of the continuous states of the agents, and *vice-versa*. 2) The discrete dynamics of the nodes are influenced by the movements of agents between nodes, while the discrete dynamics of the agents are influenced by the availability of resources in the nodes.

3.2.3 Design objective

The objective is to design the nodes and agents dynamical equations such that the usage of the resources in the environment (network) is optimized with respect to the requirements of the agents. In order to express these requirements in a more formal way,

Definition 3.4 (Utility functions) *Each agent* $k \in \mathcal{A}$ *has a utility function* $W_k(x_k) : \mathbb{R}^{N_r} \rightarrow \mathbb{R}$ *where* $x_k = (x_{k,1}, x_{k,1}, ..., x_{k,N_r})^T$ *for all* $k \in \mathcal{A}$*. The utility function is of the form:*

$$W_k(x_k) = \sum_{j \in \mathcal{R}} w_{k,j}(x_{k,j}), \quad \forall k \in \mathcal{A} \tag{3.1}$$

where $w_{k,j}(x_{k,j}) : \mathbb{R} \rightarrow \mathbb{R}$ *is strictly concave, non-decreasing, and differentiable function of* $x_{k,j}$ *for all* $k \in \mathcal{A}$ *and all* $j \in \mathcal{R}$*. Moreover, we assume that* $w_{k,j}(x_{k,j}) \rightarrow -\infty$ *as* $x_{k,j} \rightarrow 0$.

This assumption is not overly restrictive; the least information that each agent should have is its own utility function. Moreover, it is reasonable to assume that the more resources an agent obtains, the more benefit it achieves (strictly increasing utility function). The concavity and differentiability assumptions allow us to apply convex optimization techniques [11, 56] without restricting the problem solution, and the requirement that $w_{k,j}(x_{k,j}) \rightarrow -\infty$ as $x_{k,j} \rightarrow 0$ allows us to avoid the possibility of any agent getting zero resources. Therefore, the design objective can be stated as:

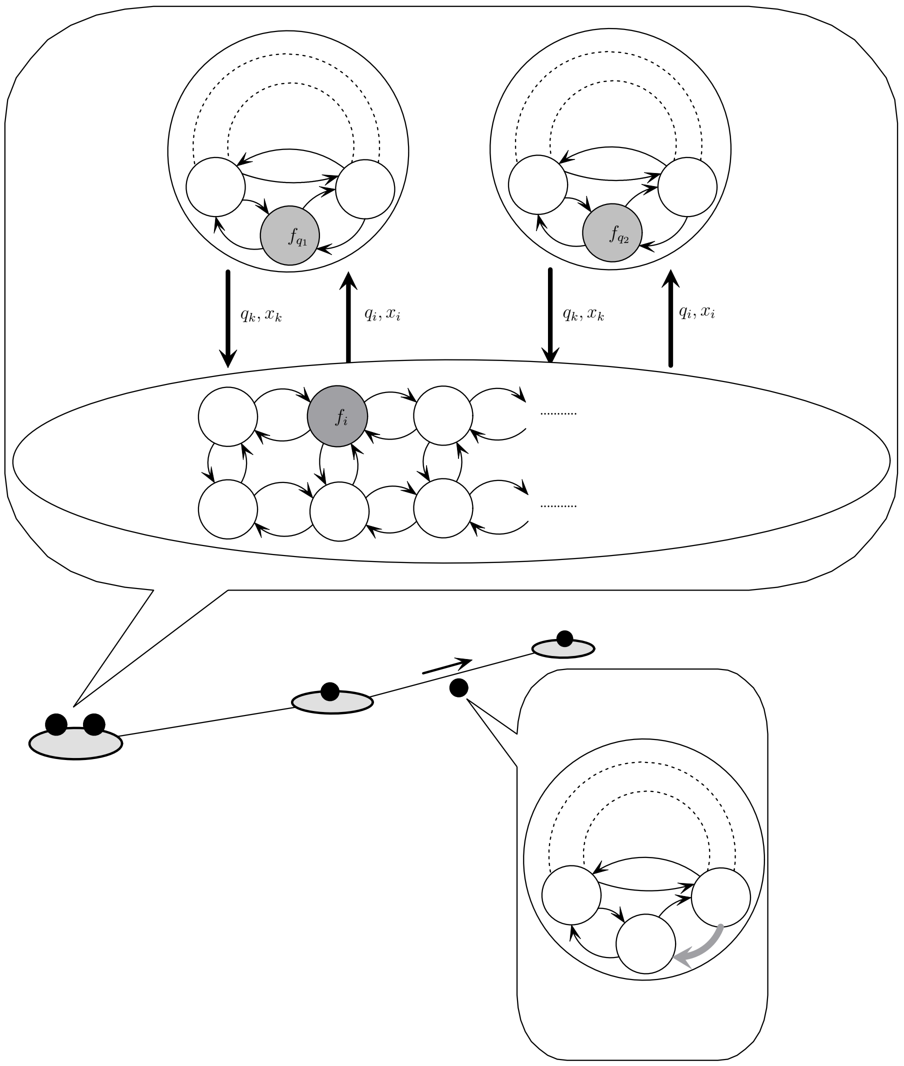

Figure 3.2: Example of the dynamical behavior of agents and nodes. Agents are modeled as hybrid automata. Each discrete state in an automaton corresponds to a possible location of an agent in the network (Agents on top). Each transition between modes represents a change of location made by an agent (agent at the bottom). The dynamics of the nodes are also modeled as hybrid systems. Each mode represents a number of agents residing at a node paired with the availability of resources that varies in discrete manner. The agents on top are located on a node, and therefore have a fixed discrete state, while the continuous dynamics of agents and the nodes that hosts them are interacting. The agent at the bottom is moving between nodes, so a discrete transition is occurring.

Problem 3.1 *Design the node and agent dynamics for all the components of the hybrid multiagent system described in Assumptions 3.1-3.3 and Definitions 3.2-3.4, such that it is asymptotically stable with equilibrium state (q_k, x_k) for all $k \in \mathcal{A}$, and the equilibrium maximizes the aggregate utility of all the agents in the network as given by $\sum_{k\in\mathcal{A}} W_k(x_k)$.*

3.3 Equivalent Hierarchical Optimization Problem

In order to gain insight into the design problem, we first analyze an optimization problem that is based on the network objective (maximization of $\sum_{k\in\mathcal{A}} W_k(x_k)$) and the constraints imposed by the system dynamics (Assumptions 3.1-3.3 and Definitions 3.2-3.4). Note that to formulate an optimization problem, we must consider a fixed configuration of the network as in the following result:

Lemma 3.1 *Given a fixed configuration of the network $\mathcal{G}$ (Fixed number of nodes, number of agents, amount of resources), the state $(q_k, x_k)_{k\in\mathcal{A}}$ that maximizes the aggregate utility function $\sum_{k\in\mathcal{A}} W_k(x_k)$ in Problem 3.1, is the solution, under the same configuration, to the following optimization problem:*

$$\max_{\left\{\substack{(q_k)_{k\in\mathcal{A}}\in\mathcal{V}^{N_a},\\ (x_k)_{k\in\mathcal{A}}\in\prod_{k\in\mathcal{A}} X_k}\right\}} \sum_{k\in\mathcal{A}} W_k(x_k) \tag{3.2}$$

subject to

$$x_{k,j} \geq 0, \quad \text{for all } (k,j) \in \mathcal{A}\times\mathcal{R} \tag{3.3a}$$

$$q_k \in \mathcal{V}, \quad \text{for all } k \in \mathcal{A} \tag{3.3b}$$

$$\sum_{\{k\in\mathcal{A}:q_k=i\}} x_{k,j} \leq r_{i,j}, \quad \text{for all } (i,j) \in \mathcal{V}\times\mathcal{R} \tag{3.3c}$$

Proof: (3.2) follows from **Problem 1**, (3.3a) and (3.3b) from Assumption 3.3, and (3.3c) from Assumption 3.2. ■

The optimization problem in Lemma 3.1 is a mixed integer-nonlinear programming problem, and as a consequence NP-Complete in the number of the discrete states [59], which in our case is given by the expression $N_d = N_v^{N_a}$ that grows exponentially with the number of nodes in the network. Therefore, the numerical solution of this problem becomes computationally intractable as the number of nodes in the network increases. We are not, however, interested in solving this problem directly. Instead, we would like to use the formulation in Lemma 3.1 to help us identify the desired characteristics of the dynamics of the nodes and the agents.

First, note that the resources of a node are allocated among the agents located in that node (Assumption 3.2), which means that the agents only have access to the resources of the nodes that hosts them, as implied by (3.3c). We show that this observation allows us to convert the mixed integer-nonlinear optimization problem into a hierarchical problem, with two subproblems: A convex optimization problem within each node in the network, and an integer optimization problem on the global behavior of the network.

Let $\bar{D} = (\bar{q}_k)_{k\in\mathcal{A}}$ be a fixed possible distribution of agents, i.e a fixed choice of q_k for all $k \in \mathcal{A}$. Let $\mathcal{V}^{N_a}$, be the set of all possible distributions of agents in the network.

Lemma 3.2 *Given a fixed possible distribution of agents $\bar{D} = (\bar{q}_k)_{k\in\mathcal{A}}$, the solution of* (3.2)-(3.3) *is given by the solution for each $(i, j) \in \mathcal{V} \times \mathcal{R}$ of the concave optimization problem:*

$$\max_{\{(x_{k,j})_{k\in V_i}\in\prod_{k\in V_i} X_k\}} \sum_{\{k\in V_i\}} w_{k,j}(x_{k,j}) \tag{3.4}$$

subject to

$$x_{k,j} \geq 0, \quad \text{for all } k \in V_i \tag{3.5a}$$

$$\sum_{\{k\in V_i\}} x_{k,j} \leq r_{i,j} \tag{3.5b}$$

Proof: Assigning a fixed value $i \in \mathcal{V}$ to each q_k allows us to discard equation (3.3b), rewrite equation (3.2) as

$$\sum_{\{i \in \mathcal{V}\}} \left[\max_{\{(x_k)_{k \in \mathcal{A}} \in \prod_{k \in \mathcal{A}} X_k : q_k = i\}} \sum_{\{k \in \mathcal{A} : q_k = i\}} W_k(x_k) \right]$$

because agents at node $i \in \mathcal{V}$ only have access to resources of node $i \in \mathcal{V}$. Equations (3.3a) and (3.3c) are also rewritten as a set of equations indexed by $\mathcal{V}$ that are independent of the choice of q_k for all $k \in \mathcal{A}$, obtaining for each $i \in \mathcal{V}$:

$$\max_{\{(x_k)_{k \in V_i} \in \prod_{k \in V_i} X_k\}} \sum_{\{k \in V_i\}} W_k(x_k), \text{ subject to}$$

$$x_{k,j} \geq 0, \text{ for all } (k,j) \in V_i \times \mathcal{R}$$

$$\sum_{\{k \in V_i\}} x_{k,j} \leq r_{i,j}, \text{ for all } j \in \mathcal{R}$$

but the objective equation can be rewritten using (3.1) as:

$$\max_{\{(x_k)_{k \in V_i} \in \prod_{k \in V_i} X_k\}} \sum_{j \in \mathcal{R}} \left(\sum_{\{k \in V_i\}} w_{k,j}(x_{k,j}) \right) \tag{3.7}$$

Since $w_{k,j}(x_{k,j})$ is a strictly concave function of $x_{k,j}$ for each $(k,j) \in \mathcal{A} \times \mathcal{R}$, the terms inside the parentheses of equation (3.7), and the complete utility function are all concave functions of their arguments. Similarly (3.3a)-(3.3c) are concave in their arguments. Therefore we can consider N_r independent concave optimization problems within each node indexed by $\mathcal{R}$. ■

Theorem 3.1 *The optimization problem* (3.2)-(3.3)*, is equivalent to the following hierarchical optimization problem:*

$$\max_{\bar{D} \in \mathcal{V}^{N_a}} \sum_{(i,j) \in \mathcal{V} \times \mathcal{R}} \mathbf{W}_{i,j}(\bar{D}) \tag{3.8}$$

where for each $(i,j) \in \mathcal{V} \times \mathcal{R}$:

$$\mathbf{W}_{i,j}(\bar{D}) = \max_{\{(x_{k,j})_{k \in V_i} \in \prod_{k \in V_i} X_k\}} \sum_{\{k \in V_i\}} w_{k,j}(x_{k,j}) \tag{3.9}$$

subject to

$$x_{k,j} \geq 0, \quad \text{for all } k \in V_i : (i,j) \in \mathcal{V} \times \mathcal{R} \tag{3.10a}$$

$$\sum_{\{k \in V_i\}} x_{k,j} \leq r_{i,j}, \quad \text{s.t. } (i,j) \in \mathcal{V} \times \mathcal{R} \tag{3.10b}$$

Proof: Equations (3.9)-(3.10) are identical to (3.4)-(3.5). Then by Lemma 3.2, (3.9)-(3.10) are solution for the optimization problem (3.2)-(3.3) if the distribution of agents is considered fixed. Therefore to obtain an equivalent description to problem (3.2)-(3.3), the agent location has to be added as a decision variable to the problem in Lemma 3.2. Note that because the agents are constrained to use resources from the node they occupy, the total benefit in the network $\sum_{k \in \mathcal{A}} W_k(x_k)$ is identical to the sum of the benefit that each node in the network generates through the agents it hosts i.e.

$$\sum_{k \in \mathcal{A}} W_k(x_k) = \sum_{(i,j) \in \mathcal{V} \times \mathcal{R}} \sum_{\{k \in V_i\}} w_{k,j}(x_{k,j})$$

If both sides of this equations are maximized with respect to x and then with respect to q we obtain the equivalence between (3.2) and (3.8)-(3.9). ■

The solution of the problem in Theorem 3.1 takes on the following conceptual form (as depicted in Figure 3.3): A centralized algorithm generates a set of possible distributions of agents in the network, and communicates this information to the nodes in the network, who solve the convex optimization problem (3.9)-(3.10) for each one of these possible configurations. As a result, they obtain a set of benefit values, one for each possible configuration, that are communicated back to the centralized algorithm which selects the configuration that yields the optimum performance for the complete network. While, this type of solution provides insight to the potential behavior of the final design of the system as shown in section 3.5, it is however undesirable and may even be unfeasible because of its centralized nature (a feasible solution using a centralized randomized algorithm is discussed in [55]).

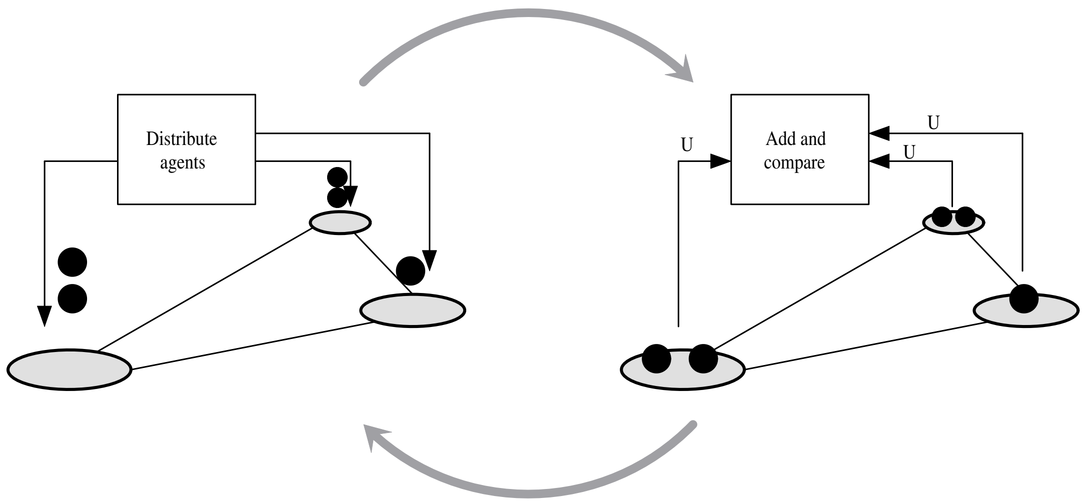

Figure 3.3: Conceptual view of the hierarchical solution: A centralized algorithm distributes the agents (Left). The nodes then allocate the resources to the agents they host, compute the benefit, and send it back to the centralized algorithm that obtains the aggregate benefit in order to compare the possible distributions (Right).

3.4 Concave Optimization Problem within each Node

Consider the optimization problem in Lemma 3.2 (or equivalently the problem (3.9)-(3.10) in Theorem 3.1). For simplicity, we drop the notation that indicates the node and type of resource $(i, j) \in \mathcal{V} \times \mathcal{R}$. Thus we consider the problem of maximizing:

$$\mathcal{W} = \sum_{k \in \hat{\mathcal{A}}} w_k(x_k), \tag{3.11}$$

subject to

$$x_k \geq 0, \quad \text{for all } k \in \hat{\mathcal{A}} \tag{3.12a}$$

$$\sum_{k \in \hat{\mathcal{A}}} x_k \leq r \tag{3.12b}$$

where $\hat{\mathcal{A}} \subseteq \mathcal{A}$ is the set of agents participating in this particular optimization task. Let $n_a = |\hat{\mathcal{A}}|$.

Lemma 3.3 *Let* $\lambda \in \mathbb{R}^{n_a+1}$ *such that* $(\lambda_p)_{p\in\{1,2,\ldots,n_a\}} = (\lambda_p)_{p\in\hat{\mathcal{A}}}$. *The necessary and sufficient conditions for* $(x_k^*)_{k\in\hat{\mathcal{A}}}$ *to be the maximal solution of the problem* (3.11)-(3.12) *are:*

$$\frac{dw_p}{dx_p} + \lambda_{n_a+1} - \lambda_p = 0, \ \ \forall p \in \hat{\mathcal{A}} \tag{3.13a}$$

$$\lambda_p x_p = 0, \ \ \forall p \in \hat{\mathcal{A}} \tag{3.13b}$$

$$\lambda_{n_a+1}\left(\sum_{k\in\hat{\mathcal{A}}}(x_k) - r\right) = 0, \tag{3.13c}$$

$$-x_p \leq 0, \ \ \forall p \in \hat{\mathcal{A}} \tag{3.13d}$$

$$\sum_{k\in\hat{\mathcal{A}}}(x_k) - r \leq 0, \tag{3.13e}$$

$$\lambda_p \leq 0, \ \ \forall p \in \hat{\mathcal{A}}\bigcup\{n_a+1\} \tag{3.13f}$$

Proof: Follows from applying Lagrange multipliers and Karush-Khun-Tucker conditions [56] to (3.11)-(3.12). ■

Lemma 3.4 *The solution* $(x_k^*)_{k\in\hat{\mathcal{A}}}, (\lambda_p^*)_{p\in\hat{\mathcal{A}}\bigcup\{n_a+1\}}$ *of equation* (3.13) *is given by* $\lambda_p^* = 0$ *for* $p \in \hat{\mathcal{A}}$ *and by*

$$\frac{dw_p}{dx_p} + \lambda' = 0, \ \ \forall p \in \hat{\mathcal{A}} \tag{3.14a}$$

$$\sum_{k\in\hat{\mathcal{A}}}(x_k) - r, = 0 \tag{3.14b}$$

$$x_p > 0, \ \ \forall p \in \hat{\mathcal{A}} \tag{3.14c}$$

$$\lambda' \leq 0 \tag{3.14d}$$

for $(x_k^*)_{k\in\hat{\mathcal{A}}}$, *and* $\lambda_{n_a+1}^* = \lambda'$.

Proof: Consider equation (3.13e), and note that since $x_p \geq 0$ for all $p \in \hat{\mathcal{A}}$ and that $w_p(x_p)$ is a strictly increasing function of x_p for all $p \in \hat{\mathcal{A}}$, any choice of $(x_k)_{k\in\hat{\mathcal{A}}}$ for (3.13e) such that $\sum_{k\in\hat{\mathcal{A}}}(x_k) - r < 0$ will be suboptimal. Thus equation (3.13e) must be modified as $\sum_{k\in\hat{\mathcal{A}}}(x_k) - r = 0$, therefore discarding equation (3.13c). Equation (3.13b) provides two choices for each p: $\lambda_p = 0$ or $x_p = 0$. The second choice however yields $u_p(x_p) = -\infty$, violating the condition for maximization of $\mathcal{W}$. Thus $\lambda_p = 0$ for all $p \in \hat{\mathcal{A}}$. The same argument leads to the modification of equation (3.13d) to $-x_p < 0$ for all $p \in \hat{\mathcal{A}}$. As a consequence (3.13a) and (3.13f) are simplified. Conditions (3.13) are then modified to obtain (3.14). ■

Applying Lemmas 3.3 and 3.4 to equations (3.4) and (3.5), the main result of this section is stated as:

Theorem 3.2 *Given the available resource $r_{i,j}$ of type $j \in \mathcal{R}$ in the node $i \in \mathcal{V}$, the utility function* (3.4) *is maximized by $(x^*_{k,j})_{k\in V_i}$, subject to* (3.5) *if and only if for each $(i,j) \in \mathcal{V}\times\mathcal{R}$, $(x^*_{k,j})_{k\in V_i}$ satisfies:*

$$\left.\frac{dw_{k,j}}{dx_{k,j}}\right|_{x_{k,j}=x^*_{k,j}} + \lambda^*_{i,j} = 0, \quad \forall k \in V_i \tag{3.15a}$$

$$\sum_{k\in V_i}(x^*_{k,j}) - r_{i,j} = 0, \tag{3.15b}$$

$$x^*_{k,j} > 0, \quad \forall k \in V_i \tag{3.15c}$$

$$\lambda^*_{i,j} \leq 0 \tag{3.15d}$$

3.5 Continuous Dynamics of Agents and Nodes

Based on the results of Section 3.3 we now provide a precise description for the continuous dynamics for both agents and nodes. We start by recognizing that the hierarchical structure proposed in Theorem 3.1 allows us to design the continuous dynamics independently from the discrete dynamics. We say that an optimization problem $\mathcal{P}$ is *solved exactly* by a system $\mathbf{H}$, if $\mathbf{H}$ has a unique asymptotically stable equilibrium point h_e that solves $\mathcal{P}$.

Proposition 3.1 *The solution to the optimization problem stated in Lemma 3.2 using the* *hybrid model for nodes and agents given in Definitions 3.2 and 3.3, only requires the consideration of the continuous dynamics in such models.*

Proof: In general, $r_{i,j}$ is allowed to vary taking on discrete values from the set Ξ for all $(i, j) \in \mathcal{V} \times \mathcal{R}$ (Assumption 3.2). However, the optimization problem in Lemma 3.2 considers a fixed amount of resources available on each node i.e., making Ξ a singleton in this case. Then from Definition 3.2 note that there is a discrete state q_i in the node's hybrid model for each possible number of agents residing in node $i \in \mathcal{V}$. So for any given choice of discrete states $(q_k)_{k\in\mathcal{A}} \in \prod_{k\in\mathcal{A}} Q_k$ in the agent's model, there is a discrete state $(q_i)_{i\in\mathcal{V}} \in \prod_{i\in\mathcal{V}} Q_i$ in each node's model that remains invariant as long as the discrete states of the agents remain fixed. Then the hybrid states of both nodes $h_i = (q_i, x_{q,i}),\ \forall i \in \mathcal{V}$ and agents $h_k = (q_k, x_{q,k}),\ \forall k \in \mathcal{V}$ have fixed discrete dynamics if the distribution of agents $\bar{D}$ remains fixed, which is assumed in Lemma 3.2. This implies that the continuous dynamics of the nodes $\mathbf{\Sigma_i}$ and the agents $\mathbf{\Sigma_k}$ interact without switching between discrete states as long as the distribution remains fixed. Therefore the optimization within each node must be solved by the corresponding continuous dynamics. ■

The optimization problem in Lemma 3.2 is a special case of a resource allocation problem considered in the literature, namely the dynamic modeling of congestion control algorithms on the Internet [1, 29, 35, 62, 77]. As a consequence, we use such results in the design of the continuous dynamics of agents and nodes in our problem, enabling a coordination algorithm that solves the optimization problem of interest. Specifically, following the treatment in [35], the optimization problem in Lemma 3.2 is a special case of equation (1 in [35]) when L has only one link. Therefore it is possible to apply the results in [1,29,35,62,77] to solve our problem.

According to [35] there are three types of dynamical systems capable of solving the optimization problem in Lemma 3.2: A primal algorithm, a dual algorithm, and a primal-dual algorithm. We choose the primal-dual approach for our problem because it is better suited for the hybrid models in Definitions 3.2 and 3.3. It is important to note that the primal and the dual approaches may also be used. A discussion of these alternatives can be found in Section 3.7.

We now provide a description for a dynamical system that solves exactly the optimization problem in Lemma 3.2. This description is based on the primal-dual algorithm developed in [1, 35, 62, 77]. Note that the state of the agents $x_{k,j}$ is similar to the state of the routes x_r in [35], and the resources $r_{i,j}$ are the analog of the link capacity c_l in [35]. We now let $p_{i,j}$ be the continuous state of the node $i \in \mathcal{V}$ and resource $j \in \mathcal{R}$, which is the analog to the price p_l in [35]. We note that the primal-dual description in [35] differs slightly from that in [62]. The following result uses the description in [62].

Lemma 3.5 *Given a fixed distribution of agents $\bar{D} \in \mathcal{V}^{N_a}$, the optimization problem in Lemma 3.2 is solved exactly for each $(i, j) \in \mathcal{V} \times \mathcal{R}$, by the following dynamical system:*

$$\dot{x}_{k,j} = K_{k,j}(x_{k,j})\big(w'_{k,j}(x_{k,j}) - p_{i,j}\big), \quad \forall k \in V_i \tag{3.16a}$$

$$\dot{p}_{i,j} = \big[L_{i,j}(p_{i,j})(y_{i,j} - r_{i,j})\big]^+_{p_{i,j}} \tag{3.16b}$$

where $x_k = (x_{k,j})_{j\in\mathcal{R}}$ is the continuous state of agent $k \in V_i \subseteq \mathcal{A}$, $p_i = (p_{i,j})_{j\in\mathcal{R}}$ is the continuous state of node $i \in \mathcal{V}$, $y_{i,j} = \sum_{k\in V_i} x_{k,j}$ for all $(i, j) \in \mathcal{V} \times \mathcal{R}$, $K_{k,j}(x_{k,j})$ is any nondecreasing, continuous function with $K_{k,j}(x_{k,j}) > 0$ for $x_{k,j} > 0$ for all $k \in V_i$ and for all $j \in \mathcal{R}$, $L_{i,j}(p_{i,j})$ is a positive, nondecreasing continuous function of $p_{i,j}$ for all $(i, j) \in \mathcal{V} \times \mathcal{R}$, $w'_{k,j}(x_{k,j}) = \frac{dw_{k,j}}{dx_{k,j}}(x_{k,j})$ is the derivative of the utility function of agent $k \in V_i$ and resource $j \in \mathcal{R}$ with respect to its argument, and

$$\big[g(t)\big]^+_t = \begin{cases} g(t), & t > 0, \\ \max(g(t), 0), & t = 0. \end{cases}$$

Moreover the dynamical system (3.16) *is asymptotically stable at its equilibrium point (the solution of the optimization problem in Lemma 3.2)*

Proof: From Proposition 3.1 it follows that the problem in Lemma 3.2 is solved using only continuous dynamics in agents and nodes. Then, given a pair $(i, j) \in \mathcal{V} \times \mathcal{R}$

we note that the optimization problem (3.4)-(3.5) is a special case of the problem (2.1)-(2.2) in [62] (or equivalently (1) in [35]). So following the discussion on the Primal-Dual Algorithm in [35, 62] the optimization problem in Lemma 3.2 is exactly solved by (3.16), which converges asymptotically to this solution. ■

Note that the continuous state of the node $p_i \in P$ with $i \in \mathcal{V}$ is related to the demand of resources within each node, and it is commonly called price. Therefore, the state of each node is directly related to the demand of resources within it, i.e the more demanded are the resources, the larger is the value of p.

Based on the previous result we can establish a detailed model for the continuous dynamics of the nodes and the agents. This dynamic description is guaranteed to solve the optimization problem in Lemma 3.2, or equivalently the problem (3.9)-(3.10) in Theorem 3.1. Therefore the continuous dynamics of this interconnected system will solve the optimization of the network resources *locally*, leaving the global optimization to the discrete dynamics of the hybrid models.

Proposition 3.2 *The following dynamical description is satisfied for all $q_i \in Q_i$ for all $i \in \mathcal{V}$:*

1. *The continuous state space $X_{q_i} = P$ where $P = \mathbb{R}^{N_r}$.*

2. *The continuous dynamics are given in a diagonal matrix:*

$$f_{q_i} = \begin{bmatrix} f_{q_{i,1}} & 0 & \dots & 0 \\ 0 & f_{q_{i,2}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & f_{q_{i,N_r}} \end{bmatrix}$$

where the state equation $\dot{p}_{q_{i,j}} = f_{q_{i,j}} = \left[L_{q_{i,j}}(p_{q_{i,j}})\big((\sum_{\mu \in U_{q_{i,j}}} \mu) - r_{q_{i,j}}\big)\right]^+_{p_{q_{i,j}}}, \forall j \in \mathcal{R}$, where $L_{q_{i,j}}(p_{q_{i,j}})$ is a positive, nondecreasing continuous function of $p_{q_{i,j}}$.

3. *The set of continuous inputs $U_{q_i} = \bigcup_{j \in \mathcal{R}} U_{q_{i,j}}$, where $U_{q_{i,j}} = \{x_{q_{k,j}} : k \in V_i\}$.*

Proof: Assumption 3.2 states that the number of types of resources in the network N_r is constant for all nodes $i \in \mathcal{V}$, and Lemma 3.5 implies there exists one state dimension for each type of resource in the network, then $X_{q,i} \subseteq \mathbb{R}^{N_r}$. Since $X_{q,i}$ has no restrictions, this implies $X_{q,i} = \mathbb{R}^{N_r}$

For item 2) note that (3.16b) describes the dynamics of one resource being allocated inside each node. Therefore in order to completely describe the N_r resources available in each node one must consider N_r decoupled resource dynamics proving the claim.

Finally for item 3), the continuous control inputs for each node $i \in \mathcal{V}$ described by $y_{i,j}$ in (3.16b) are the states of all the agents located in that node i.e., $\{x_k : k \in V_i\}$, which implies the third item for all $q_i \in Q_i$ and all $i \in \mathcal{V}$. ■

Proposition 3.3 *The following dynamical description is satisfied for all $q_k \in Q_k$ for all $k \in \mathcal{A}$:*

1. *The continuous state space $X_{q_k} = X$ where $X = \{x \in \mathbb{R}^{N_r} : x = (x_1, x_2, ..., x_{N_r}),$ $x_i \geq 0,\ \forall i \in \mathcal{R}\}$.*

2. *The continuous dynamics are given in a diagonal matrix:*

$$f_{q_k} = \begin{bmatrix} f_{q_{k,1}} & 0 & \dots & 0 \\ 0 & f_{q_{k,2}} & \ddots & \vdots \\ \vdots & \ddots & \ddots & 0 \\ 0 & \dots & 0 & f_{q_{k,N_r}} \end{bmatrix}$$

where $\dot{x}_{q_{k,j}} = f_{q_{k,j}} = K_{q_{k,j}}(x_{q_{k,j}})\big(w'_{q_{k,j}}(x_{q_{k,j}}) - u_{q_{k,j}}\big), \forall j \in \mathcal{R}$, where $K_{q_{k,j}}(x_{q_{k,j}})$ is any nondecreasing, continuous function with $K_{q_{k,j}}(x_{k,j}) > 0$ for $x_{q_{k,j}} > 0$ for all $q_k = i$ and $w'_{q_{k,j}}(x_{q_{k,j}}) = \frac{dw_{q_{k,j}}}{dx_{q_{k,j}}}(x_{q_{k,j}})$.

3. *The set of continuous inputs $U_{q_k} = \bigcup_{j \in \mathcal{R}} U_{q_{k,j}}$ where $U_{q_{k,j}}$ is the singleton set $\{p_{q_{i,j}};\ k \in V_i\}$ i.e., $u_{q_{k,j}} = p_{q_{i,j}}$ for $k \in V_i$.*

Proof: Similar to proof of Proposition 3.2. ■

We now prove that each possible distribution of agents forms a continuous dynamical system that is globally asymptotically stable at an equilibrium point that solves exactly the optimization problem in Lemma 3.2. This result guarantees that whatever the location of the agents in the nodes is, the system will asymptotically converge to a local optimum solution for that particular choice of location. Therefore, each particular combination of discrete states of nodes and agents $(q_i)_{i\in\mathcal{V}} \in \prod_{i\in\mathcal{V}} Q_i$, and $(q_k)_{k\in\mathcal{A}} \in \prod_{k\in\mathcal{A}} Q_k$, (which we call interconnection) will have a globally asymptotically stable point. The state will only be perturbed from that equilibrium when a discrete event occurs on the agents or nodes, but will converge to the equilibrium point of the new interconnection, and locally optimize the resource distribution for this new interconnection.

Theorem 3.3 *A selection of discrete states of the agents $(q_k)_{k\in\mathcal{A}} \in \prod_{k\in\mathcal{A}} Q_k$, and of discrete states of the nodes $(q_i)_{i\in\mathcal{V}} \in \prod_{i\in\mathcal{V}} Q_i$, generates $N_v \times N_r$ interconnected systems indexed by $(i, j) \in \mathcal{V} \times \mathcal{R}$, where each of them is governed by the following continuous dynamics:*

$$\dot{x}_{q_{k,j}} = K_{q_{k,j}}(x_{q_{k,j}})\big(w'_{q_{k,j}}(x_{q_{k,j}}) - p_{q_{i,j}}\big),\ \forall k \in V_i \tag{3.17a}$$

$$\dot{p}_{q_{i,j}} = \Big[L_{q_{i,j}}(p_{q_{i,j}})\big((\sum_{\kappa\in V_i} x_{q_{\kappa,j}}) - r_{q_{i,j}}\big)\Big]^{+}_{p_{q_{i,j}}} \tag{3.17b}$$

where $K_{q_{k,j}}$, $L_{q_{i,j}}$, and $w'_{q_{k,j}}$ satisfy the same conditions from Propositions 3.2 and 3.3.

Moreover, each interconnected system (indexed by $(i, j) \in \mathcal{V} \times \mathcal{R}$) is globally asymptotically stable with an equilibrium point that satisfies the conditions in Theorem 3.2 i.e., solves exactly the optimization problem in Lemma 3.2.

Proof: Given a selection of discrete states $(q_k)_{k\in\mathcal{A}} \in \prod_{k\in\mathcal{A}} Q_k$ made by the agents, the nodes, which have information about the resource availability, automatically jump to a discrete set of modes $(q_i)_{i\in\mathcal{V}} \in \prod_{i\in\mathcal{V}} Q_i$. The agents selection $(q_k)_{k\in\mathcal{A}}$ imply that each one of these agents $k \in \mathcal{A}$ has located itself in a node identified by q_k. This implies that each node $i \in \mathcal{V}$ indexes an interconnected system composed of itself and the set of agents located on it $\{k \in \mathcal{A} : k \in V_i\}$ obtaining N_v interconnected systems. However, since the the dynamics for both agents and nodes are decoupled in the resources (second item in Propositions 3.2 and 3.3) we can consider the system as formed by $N_v \times N_r$ interconnected systems, indexed by $(i, j) \in \mathcal{V} \times \mathcal{R}$. Then from Propositions 3.2 and 3.3 the interconnected system (i, j) is governed by the dynamics composed by $f_{q,i,j}$ and $\{f_{q,k,j}\}_{k \in V_i}$, which is written as (3.17). Since this equation is identical to (3.16), except for the notation stressing the dependence on the discrete mode, Lemma 3.5 implies that (3.17) are globally asymptotically stable, and that they exactly solve the optimization problem in Lemma 3.2. ■

3.6 A Numerical Example

3.6.1 Simulation set-up

In this section we provide a simulation example to clarify the concepts developed in the paper. We are interested in testing the validity of Theorem 3.3 and its relationship to the solution of the optimization problem (3.9)-(3.10) as given in Theorem 3.2.

We consider a set of ten agents ($N_a = 10$) and a graph composed of three nodes ($N_v = 3$). We assume for simplicity that the graph is completely connected and that there is only one type of resource available in the network ($N_r = 1$). The utility functions of the agents, in reference to Definition 3.4, have the form:

$$W_k(\mathbf{x}_k) = \nu_k \ln(x_k) \tag{3.18}$$

for all $k \in \mathcal{A} = \{1, .., 10\}$, where $\nu_k \in \mathbb{R}$ for all $k = 1, \cdots, 10$, and where we have dropped the dependence on the resource index for simplicity. The utility function (3.18) satisfies Definition 3.4 as long as $\nu_k > 0$. Note that ν_k are weighting factors for each agent, and are used to quantify the importance that the resource has for each agent (the greater the value of ν_k the more important is the resource for agent k). In our example, we choose ν_k as: $(\nu_1, \nu_2, ..., \nu_{10}) = (0.5, 0.6, 0.1, 0.3, 0.4, 0.9, 0.4, 0.3, 0.2, 0.1)$. Note that the particular choice of utility function for this test is commonly referred to as proportional fairness [1,29,35,62,77].

The dynamics of nodes and agents following Propositions 3.2 and 3.3, are described by:

$$f_{q_i} = \left[L_{q_i}(p_{q_i})\Big(\sum_{\{k\in\mathcal{A}:q_k=i\}} x_{q_k} - r_{q_i}\Big)\right]^+_{p_{q_i}}, \ \forall i \in \mathcal{V}, \tag{3.19a}$$

$$f_{q_k} = K_{q_k}(x_{q_k})\Big(\frac{\nu_k}{x_{q_k}} - p_{q_{i=q_k}}\Big), \ \forall k \in \mathcal{A} \tag{3.19b}$$

where $\mathcal{V} = \{1,2,3\}$, $L_{q_i}(p_{q_i}) = \tanh(p_{q_i}) + 1$, $\forall q_i \in Q_i \ \forall i \in \mathcal{V}$, and $K_{q_k}(x_{q_k}) = 50x_{q_k} \ \forall q_k \in Q_k \ \forall k \in \mathcal{A}$, satisfying the conditions in Propositions 3.2 and 3.3.

The interconnected system is tested over the time interval $T = [0, 9]$ sec. The agents start at $t = 0$ located as: $(q_1, q_2, ..., q_{10}) = (1, 3, 2, 3, 2, 1, 1, 1, 3, 2)$ with the continuous initial condition $(x_1(0), x_2(0), ..., x_{10}(0)) = (2, 1, 3, 2.2, 2, 1, 4.5, 8, 2, 2)$. The nodes start with the resource amounts $(r_1, r_2, r_3) = (2, 4, 3)$ and the continuous initial conditions $(p_1(0), p_2(0), p_3(0)) = (2, 3, 2)$. During the simulation, two events are generated to test different conditions on the interconnected system: At $t = 3$ agent 7 changes its location from $q_7 = 1$ to $q_7 = 2$, creating the new configuration $(q_1, q_2, ..., q_{10}) = (1, 3, 2, 3, 2, 1, 2, 1, 3, 2)$, and at $t = 6$ the resource at node 3 is changed from $r_3 = 3$ to $r_3 = 2$, so the new resource vector becomes $(r_1, r_2, r_3) = (2, 4, 2)$. Note from this simulation conditions that agents and nodes only visit a subset of the discrete modes in their model: Agents $1, 2, ..., 6, 8, 9, 10$ only visit the mode that corresponds to their location in the graph, which does not change during the test, while agent 7 starts at $MODE : q_7 = 1$ and at $t = 3$ changes to $MODE : q_7 = 2$. Node 1 which initially hosts agent 7 starts the simulation at $MODE : q_1 = (4 \ agents, r = 2)$ and at $t = 3$ switches to $MODE : q_1 = (3 \ agents, r = 2)$. Node 2, which is the final destination of agent 7, starts the simulation at $MODE : q_2 = (3\ agents, r = 4)$ and at $t = 3$ switches to $MODE : q_2 = (4\ agents, r = 4)$. Finally node 3 starts the simulation at $MODE : q_3 = (3\ agents, r = 3)$ and at $t = 6$ switches to $MODE : q_3 = (3\ agents, r = 2)$. These changes of modes have a direct effect on the continuous dynamics. A mode switch on an agent causes the term $p_{q,i=q_k}$ to change in (3.19b) (because the rest of the term are identical for all modes int his model). A switch on a node causes $r_{q,i}$ to change in (3.19a) if it is because of a change in the resource while causing $(\sum_{\{k\in\mathcal{A}:q_k=i\}} x_{q,k})$ to change in (3.19a) if the switch is caused by a change in the number of agents residing in the node.

3.6.2 Results

Given the set-up described in the previous subsection, the state of the system is expected to converge to an asymptotically stable equilibrium point that coincides with the solution of the optimization problem (3.9)-(3.10) with our particular choice of utility function (3.18). In order to obtain such equilibrium point we apply Theorem 3.2 to (3.18), obtaining, for each $i \in \mathcal{V}$:

$$x_k^* = \frac{r_i \nu_k}{\sum_{\{\kappa\in V_i\}} \nu_\kappa},\ \forall k \in V_i \tag{3.20a}$$

$$\lambda_k^* = -\frac{1}{r_i}\sum_{\{\kappa\in V_i\}} \nu_\kappa,\ \forall k \in V_i \tag{3.20b}$$

Substituting the values for ν_k and r_i given in previous subsection, and considering the three possible discrete configurations, one for each interval between the beginning of the simulation, the events, and the end of the simulation, we obtain the results are summarized in Figures 3.4-3.6. We only show results for some of the agents. The rest of the agents behave in similar form. The plots in Figures 3.4 and 3.5, show the time evolution of the continuous states of the agents 5, 6, 7, and 9. The vertical segmented lines indicate the time of occurrence of the events that were mentioned in the previous subsection. The horizontal dotted lines with $*$-marks at the end points indicate the expected equilibria during for each interval between events given by the solution of the optimization problem (3.9)-(3.10) with utility function (3.18). As seen from Figures 3.4 and 3.5, the states of all the agents converge to a stable equilibrium point on each interval between events. This equilibrium coincides with the solution of the optimization problem.

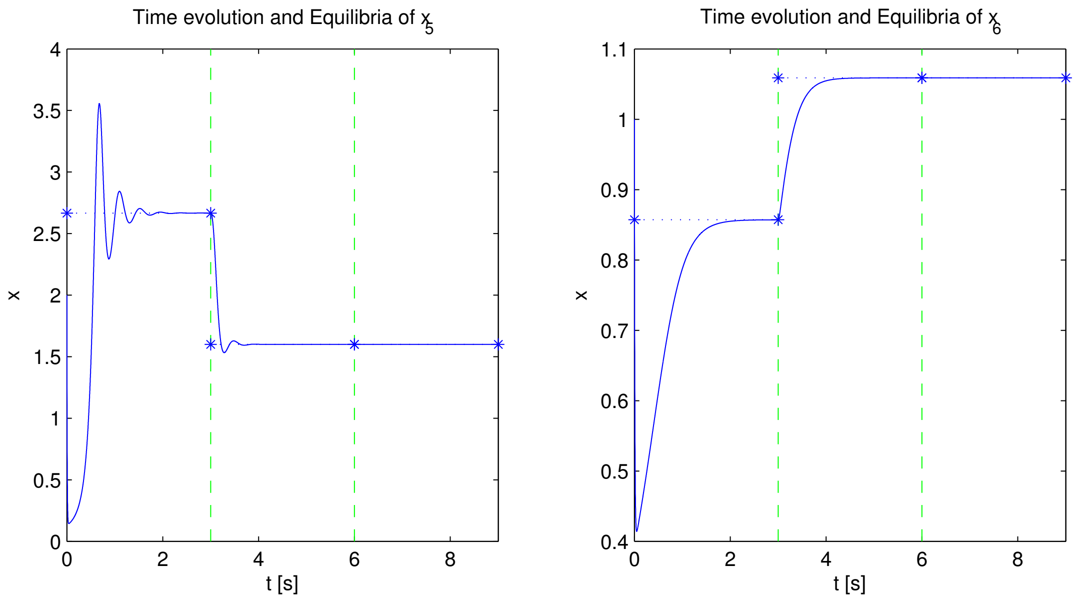

Figure 3.4: Dynamic behavior and optimal stable equilibria of agents 5 (left) and 6 (right). The solid (blue) curves show the dynamic behavior of the states. The segmented (green) vertical lines indicate the occurrence of events that change the operating conditions of the system.The dotted (blue) horizontal lines with $*$-marks at the end points indicate the optimal solution to the corresponding optimization problem for the system configuration during that time intervalthat is expected to coincide with the equilibrium point where the dynamics approach during such interval.

To see the effects of the first event on agents, observe the behavior shown in Figures 3.4 and 3.5. Agent 7 (Fig. 3.5 left) converges from an initial condition to the equilibrium point in the interval $t \in [0, 3)$. Then, after the event at $t = 3$ where it is moved to a different node, agent 7 converges to a different equilibrium point in $t = [3, 9]$. Note that the second event does not affect the evolution of agent 7 because this event happens at node 3, while agent 7 is located at node 2 when this event happens. Observe also the behavior of agents

5 and 6 in Figure 3.4 left and right respectively. Agent 6 which is located at node 1, the node where agent 7 starts the simulation, suffers a change in its equilibrium point when agent 7 moves out from node 1. Moreover note that the new equilibrium value is greater than the previous one. This is because agent 7 released resources from node 1. Similarly, agent 5 that is located at node 2 changes its equilibrium point after agent 7 arrives this node. Its new equilibrium value is smaller than its previous one because agent 7 obtains some resources at this node after arriving to it. Finally note that the second event does not affect the behavior of agents 5 or 6. Similar behavior is observed on other agents located at nodes 1 and 2.

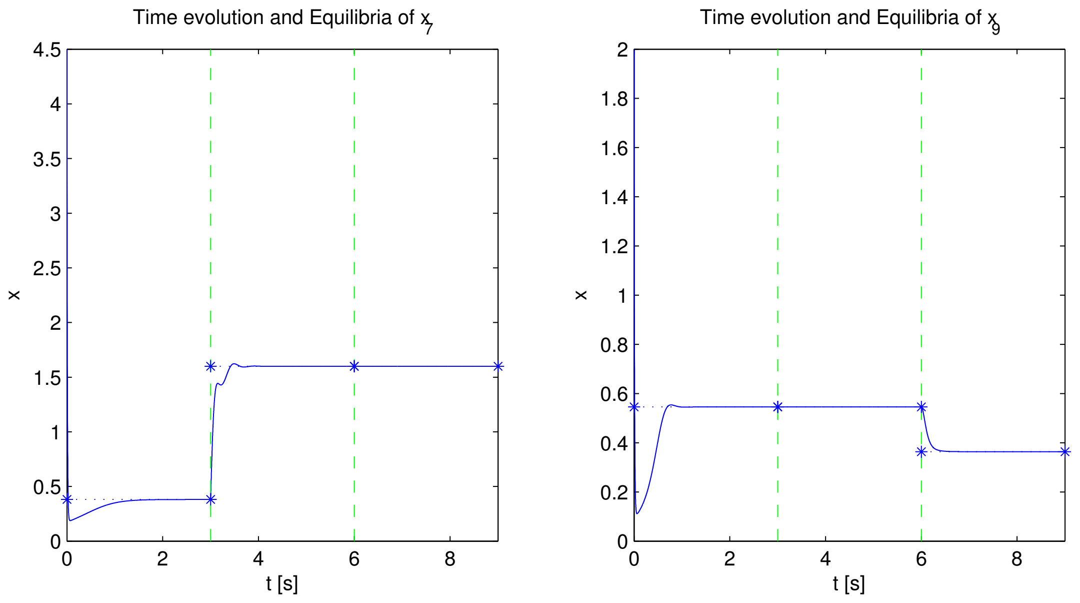

Figure 3.5: Dynamic behavior and optimal stable equilibria of agents 7 (left) and 9 (right). The solid (blue) curves show the dynamic behavior of the states. The segmented (green) vertical lines indicate the occurrence of events that change the operating conditions of the system.The dotted (blue) horizontal lines with $*$-marks at the end points indicate the optimal solution to the corresponding optimization problem for the system configuration during that time interval that is expected to coincide with the equilibrium point where the dynamics approach during such interval.

The effects of the second event on the agents is showcased by the evolution of agent 9 in Figure 3.5 (right). This agent, which is located at node 3, changes its equilibrium point after the resources in this node are reduced. Note that the new equilibrium value is smaller than the older one reflecting the reduction of resources in the node. Also note that the first event does not affect the behavior of this agent. Similar behavior is observed on all agents located at node 3.

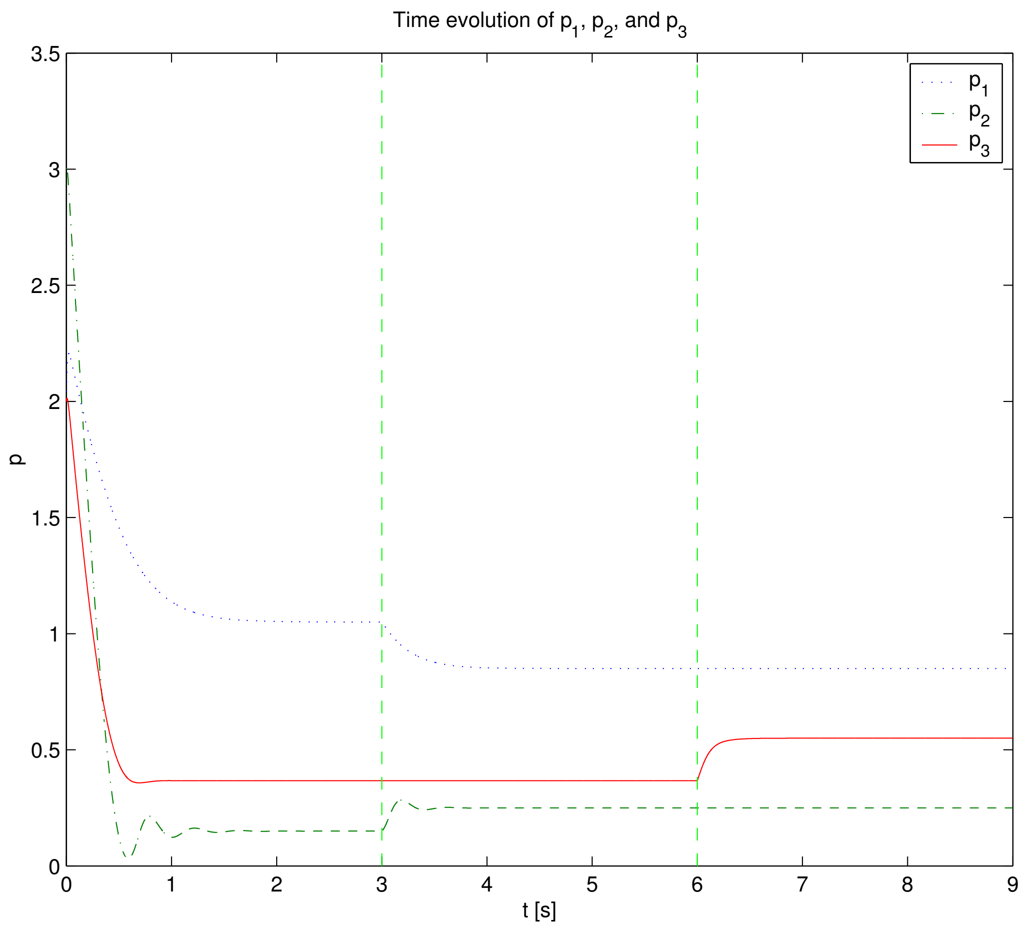

Figure 3.6: Dynamic behavior of nodes 1 (dotted blue curve), 2 (segmented green curve), and 3 (solid red curve). The segmented (green) vertical lines indicate the occurrence of events that change the operating conditions of the system.

The effect of the events is also observed in the nodes as seen in Figure 3.6. The first event affects nodes 1 and 2, because agent 7 moves from node 1 to node 2. After this event the state of the node 1 switches its equilibrium point to a smaller value than its original one. This is expected because the state on the nodes are directly related to the demand of resources caused by the agents. Therefore as agent 7 leaves node 1, the demand of resources in this node is reduced, causing a reduction in the value of its equilibrium point. On the other hand, node 2 increases its equilibrium value because agent 7 increases the demand of resources in this node. Finally node 3 increases its equilibrium value after the second event because this event reduces the resources available in this node, causing an increase of resources' demand.

To summarize, we observe as expected from Theorem 3.3, that each different configuration of agents locations and amounts of resources, with dynamics given in Propositions 3.2 and 3.3 have a stable equilibrium point which coincides with the solution to its corresponding optimization problem. We have also observed that agent-related events affect the dynamic behavior and optimal solution of the systems at both the origin and destination nodes, while node-related events only affect the condition at the local node. This happens in part because we have not included discrete transition rules in the agent's and node's hybrid models. We expect this to change when the design is complete.

3.7 Conclusions

The problem studied in this chapter considers agents moving on a network of discrete locations. The agents need resources in order to perform some tasks, and such resources are provided by the environment. The agents' objective is to obtain the best possible resources from the network in order to maximize their satisfaction measured using a utility function. The objective of the multi-agent system, however, is to achieve a group behavior such that the utilization of the network resources is globally optimized.

The overall behavior of the system includes resource allocation, movement of agents between discrete locations, and a change of network conditions. Therefore, both agents and nodes need to be described using continuous (resource allocation) and discrete dynamics (agents movement and varying network conditions) that can be captured by a hybrid model [12, 36]. The hybrid model is not complete, and this chapter outlines how to obtain the continuous dynamics only, leaving the discrete dynamics unspecified.

The continuous dynamics are designed using results borrowed from Internet congestion control algorithms [1, 29, 35, 62, 77]. This is done by posing an optimization problem that is equivalent to the multi-agent system overall objective, and then using the results in [35, 62] to obtain a precise dynamical description of the continuous dynamics of nodes and agents. This model forms an interconnected system for each possible configuration of agents and nodes that is globally asymptotically stable by design, and that optimizes the usage of resources locally within each node.

The discrete dynamics are a key factor to achieving global optimization of resource utilization in the network. The design of this part of the model is expected to benefit from several techniques that are discussed in the following chapters, which include the the study of dynamical properties for multi-agent systems with hybrid interacting dynamics, abstraction procedures of the continuous dynamics of the system, in order to obtain a simplified, but still meaningful description of the dynamic behavior of the interconnected system, and the design of the discrete dynamical components of the hybrid model for agents and nodes using optimal control techniques.

Chapter 4

Dynamical Properties of Multiple Interacting Hybrid Systems

4.1 Introduction

Motivated by the discussion in the previous chapter, we present a general framework for describing multi-agent systems with hybrid interacting dynamics, where the interaction between agents occurs at both the continuous and discrete levels, and study dynamical properties of these systems. We define multi-agent systems as Interconnected Hybrid Systems, recast fundamental hybrid concepts such as hybrid execution and reachability in this new interconnected hybrid systems framework, and prove a necessary and sufficient condition for the existence and uniqueness of the interconnected hybrid executions, extending previous work on hybrid systems. We provide conditions on each agent's hybrid model that guarantee the multi-agent system's existence and uniqueness property. Finally, we continue to study the problem described in Chapter 3 by showing how to apply the existence and uniqueness conditions in the design of the agents and nodes' dynamics.

The vision of an Internet in which functions are not fixed to physical nodes, but are instead implemented by software agents that are free to migrate from node to node, depending on resources that they may have to compete for gives rise to a new type of multi-agent system where agent dynamics are composed by discrete states that represent the location of the agent in the network and its operating mode, and by continuous states that represent the amount of resources that the agent is receiving from the network. The node dynamics are also composed by discrete and continuous states. The discrete states represent changes in the agents hosted by the node, while continuous states represent the evolution of the resource availability due to the competition of agents for such resources. Agents start at initial locations in the network and with a given set of resources. Nodes start at discrete states that reflect the initial distribution of agents and at continuous states corresponding the initial availability of resources. The continuous states of the agents may then evolve according the agents requirements affecting the availability of resources in the nodes. Agents may also jump to different locations depending on the conditions at the nodes. These jumps will affect the continuous evolution of other agents and nodes, and will also cause discrete jumps at the nodes reflecting the new agent distribution. A pictorial example of this situation is depicted in Figure 4.1.

It is not clear how to capture the operation of such a system with existing hybrid frameworks. The interactions between the hybrid systems that model agents and nodes happen at both the continuous and discrete levels. The continuous and discrete dynamics of the agents depend on both the continuous and discrete states of the nodes and viceversa. If a single hybrid model was formulated to study this type of multi-agent system the result could be a that of a centralized model where it would be difficult to decouple individual agent's descriptions. Instead, we attempt to capture this interaction with a new class of systems: the interconnected hybrid systems. Such systems are not mere parallel compositions [61], or products of the component hybrid subsystems. The existence and evolution of an individual subsystem can be meaningless if isolated. Moreover, interactions are not limited to common or uncommon events. In our case, the hybrid state in one of the systems modifies the execution in another one. Therefore we formally define the inter-

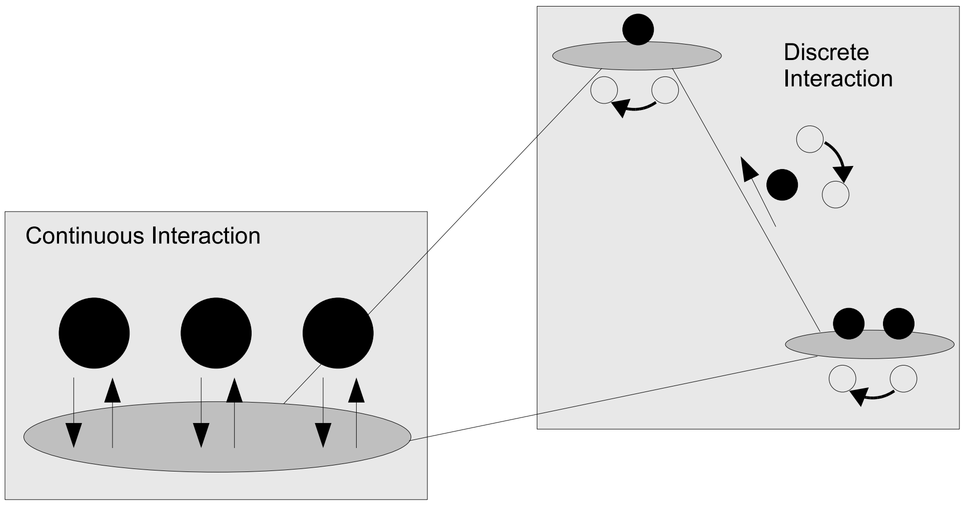

Figure 4.1: Example of dynamical behavior of agents and nodes. Agents are modeled as hybrid automata. Each mode in an automaton corresponds to a possible location of an agent in the network. Each transition between modes represents a change of location made by an agent. The dynamics of the nodes are also modeled as hybrid systems. Each mode represents a set of agents residing at a node paired with the availability of resources that varies in discrete manner. The agents on the left are located on a node, therefore have a discrete state fixed and the continuous dynamics of agents and the nodes that hosts them are interacting. The agent on the right is moving between nodes, so a discrete transition is occurring.

connected hybrid system such that the continuous evolution in one agent depends on the continuous states of agents that are connected to it, and similarly the discrete dynamics depend on continuous and discrete dynamics of neighboring agents. This definition also includes a description of the connectivity of the multi-agent system in each agent's hybrid state. We then recast the reachability and the hybrid execution concepts from hybrid systems theory into the new framework, and provide a necessary and sufficient condition for the existence and uniqueness of the interconnected hybrid execution (the hybrid analog to the state's evolution in continuous dynamical systems), in terms of each agent's hybrid model, extending some of the concepts in [37]. We use an application example to illustrate how to use the existence and uniqueness condition for designing the general dynamics of the agents locally, while guaranteing the existence and uniqueness of the execution of the multi-agent system globally.

4.2 Interconnected Hybrid Systems

We consider a multi-agent system with individual hybrid dynamics. The agents in the system interact at both the continuous and discrete level through the continuous control input and the discrete transition guards respectively. We index the agents in the system by the set I. A hybrid system is denoted by $\mathbf{H}_i$, for all $i \in I$.

We use the following notation: ν_k denotes dependence of ν on k. ν_{q_k} denotes dependence of ν on q_k which depends on k. ν^n, denotes the n^{th} element of a sequence in ν, $\nu(t)$ denotes the value of ν at time t, and with some abuse of notation, ν_0 marks an initial condition. $\{\nu_k\}_{k\in K}$ denotes a collection of ν_k indexed by the set K. Similarly $(\nu_k)_{k\in K}$ denotes the vector $(\nu_1, \nu_2, \dots, \nu_{|K|})$ indexed by the set K.

Let O_i is the set of operating states and D_i is the set of connectivity states of $\mathbf{H}_i$. Each $o_i \in O_i$ represents a different operating condition of $\mathbf{H}_i$. Each $d_i \in D_i$, represents different connectivity conditions. Let Q_i be the set of discrete states of $\mathbf{H}_i$, such that $Q_i = O_i \times D_i$, where $(o_i, d_i) \in Q_i$ is denoted as q_i. Each q_i has an associated set $V(q_i) \subseteq I \; \forall q_i \in Q_i$, which stores the indexes of the systems that are connected to $\mathbf{H}_i$, i.e., if $j \in V(q_i)$ then $\mathbf{H}_j$ is connected to $\mathbf{H}_i$. Note that $V(q) = V(q')$ for all $q = (o, d), q' = (o', d') \in Q_i$ that satisfy $d = d'$.

Let $\Sigma_i = \{\Sigma_{q_i}\}_{q_i \in Q_i}$ be a collection of continuous dynamical systems Σ_{q_i} indexed by the set Q_i. Each continuous system is a tuple $\Sigma_{q_i} = (X_{q_i}, f_{q_i}, U_{q_i}, \mathbb{R}^+)$ where X_{q_i} is the continuous state space, f_{q_i} the continuous dynamics, U_{q_i} the set of continuous controls, and $\mathbb{R}^+ = [0, \infty)$ the time set. Also let $X_i = \bigcup_{q_i \in Q_i} X_{q_i}$ be the continuous state space over all the discrete states of agent i.

Let $\mathbf{S}_i = \{S_{q_i}\}_{q_i \in Q_i}$ be the set of discrete transition labels of $\mathbf{H}_i$. Symbol $s_{q_i} \in S_{q_i}$ determines the discrete state after a transition from $q_i \in Q_i$ in system $\mathbf{H}_i$. We consider only state based (autonomous) transition in this paper.

Let $\mathbf{G}_i = \{G_{q_i}\}_{q_i \in Q_i}$ be the set of guard conditions for $\mathbf{H}_i$. G_{q_i} is a map that determines when a transition is possible from $q_i \in Q_i$. Let $\mathbf{Z}_i = \{Z_{q_i}\}_{q_i \in Q_i}$ be the set of transition maps of $\mathbf{H}_i$, where $Z_{q_i} : G_{q_i} \times S_{q_i} \to X_i$ determines the continuous state of $\mathbf{H}_i$ after a transition label in S_{q_i} from a hybrid state in G_{q_i}.

The hybrid state space of agent i is $H_i = Q_i \times X_i$, the continuous state space of the agents that are connected to i is $X_{V(q_i)} = \prod_{j \in V(q_i)} X_j$, and the hybrid state space of the agents that are connected to i is $H_{V(q_i)} = \prod_{j \in V(q_i)} H_j$.

Definition 4.1 (Interconnected Hybrid System (IHS)) *An* Interconnected Hybrid System *is a set* $\mathbf{H}^* = \{\mathbf{H}_i\}_{i \in I}$ *of Controlled Hybrid Dynamical Systems [12]* $\mathbf{H}_i$ *indexed by the set I. For each* $i \in I$, $\mathbf{H}_i = [Q_i, \Sigma_i, \mathbf{G}_i, \mathbf{Z}_i, \mathbf{S}_i]$, *such that*

- *The continuous control inputs in* U_{q_i} *are the continuous states of the systems that are connected to* $\mathbf{H}_i$. *Therefore* $U_{q_i} = X_{q_i} \times X_{V(q_i)}$.
- *A guard condition for a discrete transition of agent* i *is a function* $G_{q_i} : S_{q_i} \to X_{q_i} \times H_{V(q_i)}$. G_{q_i} *specifies when a transition is possible as a function of the continuous state of agent* i *and the hybrid states of agents connected to* i. $G_{q_i}(s) = G^L_{q_i}(s) \times G^R_{q_i}(s)$ *where* $G^L_{q_i}(s) \subseteq X_{q_i}$ *denotes the local condition of* $G_{q_i}(s)$ *i.e. the condition on the continuous state of agent i, and* $G^R_{q_i}(s) \subseteq H_{V(q_i)}$ *denotes the remote condition of* $G_{q_i}(s)$ *i.e. the condition on the hybrid states of agent connected to* i.

The discrete state space of the IHS $\mathbf{H}^*$ *is* $Q_I = \prod_{i \in I} Q_i$, *its continuous state space is* $X_I = \prod_{i \in I} X_i$, *and its hybrid state space is* $H_I = \prod_{i \in I} H_i$. *The state of the IHS is denoted as* $\vec{h} = (\vec{q}, \vec{x}_{\vec{q}})$ *where* $\vec{q} = (q_i)_{i \in I} \in Q_I$, *and* $\vec{x}_{\vec{q}} = (x_{q_i})_{i \in I} \in X_I$.

Note in Definition 4.1 that the discrete states of the systems are divided into operating states, which are used to describe modes of operation of each individual agent in the system, and connectivity states, which describe the possible configurations for information exchange between agents in the system. If one thinks in the usual graph theoretic concept that describes connectivity between agents in multi-agent systems literature [17, 20, 27, 41, 45, 57] different connectivity states correspond to its different possible neighborhoods. Also note that no assumptions are made about the direction of information flow, so this definition includes the possibility of agent $i \in I$ being connected to agent $j \in I : j \neq i$ without j being connected to i (a directed graph).

Interactions between the continuous dynamics of the agents occur through their continuous control inputs. The continuous control inputs of agent $i \in I$ in the IHS are functions of the continuous state of agent $i \in I$ and the continuous states of the agents that are directly connected to agent $i \in I$.

Interactions between the discrete dynamics of the agents occur at their discrete transition guards. The transition guards of agent $i \in I$ set conditions on the continuous states of agent $i \in I$ through $G^L_{q_i}(s)$ and on the hybrid states of the agents that are connected to agent $i \in I$ through $G^R_{q_i}(s)$. So a discrete transition may occur when both the continuous state of agent i and the hybrid states the the agents connected to $i \in I$ reach a guard condition. Therefore the discrete dynamics of agent $i \in I$ are influenced by the hybrid states of the agents that are connected to agent $i \in I$. A graphical example of an IHS is shown in Figure 4.2.

Definition 4.1 presents a hybrid analog to the standard multi-agent setting [17, 20, 27, 41, 45, 57] where each agent uses the states of its neighbors to update its own evolution. The following is an standing assumption for the rest of this paper.

Assumption 4.1 *The sets of discrete states Q_i are finite for all $i \in I$. The continuous state space $X_i \subseteq \mathbb{R}^d$ for all $i \in I$, where d is an integer. The vector fields $f_{q_i}(x_{q_i}, u_{q_i}, t)$ are*

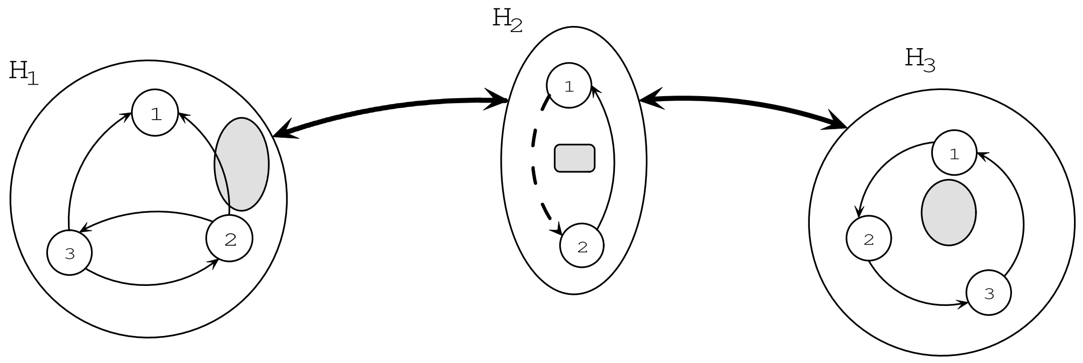

Figure 4.2: Graphical representation of an IHS. Three hybrid systems interconnected in line topology with bidirectional connectivity. The area inside ovals and circles represent the continuous state space of the agents. The graphs inside the continuous spaces represent the automata for each system. The gray area within the state space represents the guard for the discrete transition indicated with segmented thicker line in $\mathbf{H}_2$. Note that this transition depends on the continuous states of the agents connected to $\mathbf{H}_2$

globally Lipschitz continuous on both x_{q_i} and u_{q_i} with Lipschitz constants $L^x_{q_i}$ and $L^u_{q_i}$ for all $q_i \in Q_i$ for all $i \in I$.

4.3 Interconnected Hybrid Execution

In this section we introduce the Interconnected Hybrid Execution (IHE) based on the concept of hybrid execution in [37]. The IHE is the analog of the state evolution of a continuous multi-agent dynamical system, and captures the system's hybrid behavior over time, with respect to both discrete and continuous interactions of the agents among themselves and with and with its environment.

An *Interconnected Hybrid Time Trajectory (IHTT)* is a sequence $\tau = \{\tau^0, \tau^1, \tau^2 \ldots, \tau^n, \ldots, \tau^N\}$, where 1) τ^0 is the time when $\mathbf{H}^*$ starts its evolution, 2) τ^n for $n = 1, \ldots, N-1$ is the time at which there is at least one system $\mathbf{H}_i \in \mathbf{H}^*$ that makes a discrete transition from q_i^n to q_i^{n+1}, such that the Interconnected Hybrid System $\mathbf{H}^*$ makes a discrete transition from $\vec{q}^n$ to $\vec{q}^{n+1}$, and 3) τ^N is the time when $\mathbf{H}^*$ ends its evolution. Two consecutive elements in the IHTT satisfy $\tau^n \leq \tau^{n+1}$ for all $n = 0, 1, \ldots, N-1$. Finally the IHTT τ is infinite if $N = \infty$ and is finite otherwise.

The IHTT is used to encode timing information for the continuous and discrete dynamics of the IHS $\mathbf{H}^*$. The IHTT stores the times when a discrete transition takes place at least on one of the agents in the system. As a consequence the IHTT also specifies time intervals between two consecutive elements in the sequence where uninterrupted continuous evolution takes place.

Note that between two consecutive interconnected discrete states $\vec{q}^n = (q_i^n)_{i\in I}$ and $\vec{q}^{n+1} = (q_i^{n+1})_{i\in I}$, the individual discrete states q_i^n and q_i^{n+1} are different for all $i \in I$ that executed a transition at time τ^n, while $q_i^n = q_i^{n+1}$ for all $i \in I$ that did not execute a transition at time τ^n. Therefore the IHTT as defined above allows more than one hybrid agent in the IHS to a discrete transition at the same time.

Let a (candidate) execution be the collection $(\tau, \mathbf{q}, \mathbf{s}, \mathbf{x}, \mathbf{u})$ where:

- τ is an interconnected hybrid time trajectory.
- $\mathbf{q} = \{\vec{q}^0, \vec{q}^1, \ldots, \vec{q}^n, \ldots, \vec{q}^N\}$ is a sequence of vectors of discrete locations $\vec{q}^n = (q_i^n)_{i\in I}$ where q_i^n is the discrete mode of system $\mathbf{H}_i$ at the n^{th} step in the execution.
- $\mathbf{s} = \{\vec{s}^0, \vec{s}^1, \ldots, \vec{s}^n, \ldots, \vec{s}^N\}$ is a sequence of vectors of switching labels $\vec{s}^n = (s_{q_i}^n)_{i\in I}$ where $s_{q_i}^n$ is the discrete transition that occurs on system $\mathbf{H}_i$ at n^{th} step in the execution.
- $\mathbf{x} = \{\vec{x}^0, \vec{x}^1, \ldots, \vec{x}^n, \ldots, \vec{x}^N\}$ is a sequence of continuous evolution $\vec{x}^n = (x_{q_i^n})_{i\in I}$ where $x_{q_i^n}$ is a differentiable map $x_{q_i^n} : [\tau^{n-1}, \tau^n[\to X_{q_i^n}$ of system $\mathbf{H}_i$ at the n^{th} step in the execution.
- $\mathbf{u} = \{\vec{u}^0, \vec{u}^1, \ldots, \vec{u}^n, \ldots, \vec{u}^N\}$ is a sequence of continuous control inputs $\vec{u}^n =$

$\left(u_{q_i^n}\right)_{i\in I}$ where $u_{q_i^n}$ is a continuous map $u_{q_i^n} : [\tau^{n-1}, \tau^n[\rightarrow U_{q_i^n}$ of system $\mathbf{H}_i$ at the n^{th} step in the execution.

In order to simplify notation, we use the following convention: Unless otherwise noted, if we use an interconnected hybrid state $\vec{h}$, an interconnected discrete state $\vec{q}$, and/or a discrete state q_i in the same sentence/paragraph it implies that the discrete state q_i is a component of an interconnected discrete state $\vec{q}$, which is a component of an interconnected hybrid state $\vec{h}$. We will follow that convention for continuous states, transition labels and continuous inputs as well.

We say that $\vec{h}(t)$ satisfies the discrete transition guard $G_{q_i}(s)$, if the local component of $\vec{h}(t)$ satisfies the local part of $G_{q_i}(s)$ - $x_{q_i}(t) \in G^L_{q_i}(s)$) and the remote component of $\vec{h}(t)$ satisfies the remote part of $G_{q_i}(s)$ - $(h_j)_{j\in V(q_i)}(t) \in G^R_{q_i}(s)$. We say that $\vec{h}(t)$ satisfies interconnected discrete transition guard $G_{\vec{q}}(\vec{s})$, if $\vec{h}(t)$ satisfies $G_{q_i}(s_{q_i})$ for all $i \in I$, where $\vec{q} = (q_i)_{i\in I}$ and $\vec{s} = (s_{q_i})_{i\in I}$.

We say that $\vec{x}_{\vec{q}'}$ is in the interconnected transition map $Z_{\vec{q}}(\vec{h}, \vec{s})$, if each component of $\vec{x}_{\vec{q}'}$ satisfies the transition map of each component of $\vec{s}$, i.e. $x_{q_i'} \in Z_{q_i}(\vec{h}, s_{q_i})$ for all $i \in I$, such that $\vec{q}' = (q_i')_{i\in I}$, $\vec{q} = (q_i)_{i\in I}$, $\vec{x}_{\vec{q}'} = (x_{q_i'})_{i\in I}$, and $\vec{s} = (s_{q_i})_{i\in I}$.

Definition 4.2 (Interconnected Hybrid Execution (IHE)) *An* Interconnected Hybrid Execution *with initial condition $\vec{h}_0$ is a collection $\chi(\vec{h}_0) = (\tau, \mathbf{q}, \mathbf{s}, \mathbf{x}, \mathbf{u})$ where:*

- *Initial Condition: $\vec{h}_0 = (\vec{q}^0, \vec{x}^0(0))$ is an initial condition of $\mathbf{H}^*$.*
- *Continuous Dynamics: $\dot{\vec{x}}^n = \vec{f}_{\vec{q}^n}(\vec{x}^n, \vec{u}^n, t)$ for all $t \in [\tau^{n-1}, \tau^n)$, for all $n \in \{1, 2, \ldots, N\}$, where $\vec{f}_{\vec{q}^n}(\cdot) = \left(f_{q_i^n}(\cdot)\right)_{i\in I}$ is the vector field of all agents in the IHS $\mathbf{H}^*$.*
- *Discrete Dynamics: The following conditions hold for all $n \in \{0, 1, 2, \ldots, N-1\}$:*

- *The discrete state after a transition $\vec{q}^{n+1}$ is equal to the corresponding discrete transition label $\vec{s}^n$, i.e. $\vec{q}^{n+1} = \vec{s}^n$*
- *The hybrid state before a transition $\vec{h}^n(\tau^n)$ satisfies the corresponding transition guard $G_{\vec{q}^n}(\vec{s}^n)$.*
- *The continuous state after a transition $\vec{x}_{\vec{q}^{n+1}}(\tau^{n+1})$ is in the corresponding transition map, i.e. $\vec{x}_{\vec{q}^{n+1}}(\tau^{n+1}) \in Z_{\vec{q}^n}(\vec{h}^n, \vec{s}^n)$.*

The IHE provides the information about the continuous and discrete states and inputs of the system at each instant of its evolution. It is the analog of the state-input trajectory in continuous time systems. The conditions imposed in Definition 4.2 are required for the execution to to satisfy the dynamics of $\mathbf{H}^*$. Therefore an IHE starts at a valid initial condition. The continuous evolution between two times in the interconnected hybrid time trajectory satisfies the continuous dynamics of each agent, and the discrete transitions have valid transition guards and transition maps.

4.4 Existence and Uniqueness of the IHE

We provide conditions for the existence and uniqueness of an infinite IHE. These conditions are stated as a function of each agent in the system. Therefore the desired global behavior of the system (existence and uniqueness of its execution), can be guaranteed by the specification of local design variables inside each agents dynamics.

The following concepts are very similar to those for individual hybrid systems. Let $\chi^S(\vec{h}_0)$ denote the set of all IHEs with initial condition $\vec{h}_0$, and similarly $\chi^F(\vec{h}_0)$ denotes the set of all finite IHEs, $\chi^\infty(\vec{h}_0)$ denotes the set of all infinite IHEs, and $\chi^M(\vec{h}_0)$ denotes the set of all maximal IHEs. Init denotes the set of all initial conditions.

We say that a finite IHE $\chi(\vec{h}_0) \in \chi^F(\vec{h}_0)$ maps $\vec{h}_0$ to $\vec{h}$ if its IHTT $\tau = \{\tau^0, \tau^1, \ldots, \tau^N\}$ and $\vec{h} = (\vec{q}^N, \vec{x}^N(\tau^N))$. The interconnected hybrid state $\vec{h}$ is reachable from initial condition $\vec{h}_0$ - denoted $\vec{h} \in Reach(\vec{h}_0)$ if there exists a finite IHE $\chi(\vec{h}_0) \in \chi^F(\vec{h}_0)$ that maps $\vec{h}_0$ to $\vec{h}$. The set of states $\vec{h}$ that can be reached from any initial condition is called *Interconnected Reachable Set*:

$$Reach_{\mathbf{H}^*} = \bigcup_{\vec{h}_0 \in \mathrm{Init}} Reach(\vec{h}_0)$$

Let $\psi(q_i, t)$ denote the continuous flow of $f_{q_i}(x_{q_i}, u_{q_i}, t)$. We define the *Set of Blocked Continuous Evolution* as the set that specifies what states in the system require a discrete transition for the system to continue its evolution:

$$\begin{aligned} Out_{\mathbf{H}^*} &= \{\vec{h} \in H_I; \forall \epsilon > 0, \exists t \in [0, \epsilon) \text{ and } \exists i \in I, \\ &\quad \text{s.t. } \psi(q_i, t) \notin X_{q_i}\} \end{aligned}$$

We say that $\mathbf{H}^*$ is deterministic if given $\vec{h}_0$, $\chi^M(\vec{h}_0)$ contains at most one element. The following result provides the necessary and sufficient conditions for existence of an infinite execution given that the system is deterministic. These conditions combined with those for an IHS to be deterministic yield existence and uniqueness of an infinite IHE. The proof of Lemma 4.1 is provided in Section 4.7.

Lemma 4.1 (Deterministic existence) *Suppose* $\mathbf{H}^*$ *is deterministic. Then given an initial condition* $\vec{h}_0$*,* $\chi^\infty(\vec{h}_0)$ *is nonempty (an infinite execution exists) if and only if for all* $\vec{h} \in Reach_{\mathbf{H}^*} \bigcap Out_{\mathbf{H}^*}$ *there exist a* $\vec{s} \in S_{\vec{q}}$ *such that* $\vec{h}$ *satisfies* $G_{\vec{q}}(\vec{s})$*, where* $\vec{q}$ *is the discrete state of* $\vec{h}$*, and* $S_{\vec{q}} = \prod_{i \in I} S_{q_i}$ *such that* $\vec{q} = (q_i)_{i \in I}$*.*

Proof: (Sketch) ($\Rightarrow$) Suppose all conditions in Lemma 4.1 hold except that there is a $\vec{h}$ whose continuous evolution is blocked but does not satisfy any discrete transition guard. Since $\vec{h}$ is reachable there is a finite execution $\chi(\vec{h}_0)$ that maps the initial condition to $\vec{h}$. If this finite execution is extended through continuous evolution it contradicts assumption that continuous evolution is blocked for $\vec{h}$. If this finite execution is extended through a discrete transition it contradicts assumption that $\vec{h}$ does not satisfy any discrete transition guard. These two statements imply the the finite execution is maximal. However an infinite execution starting at $\vec{h}_0$ is also maximal and different from $\chi(\vec{h}_0)$, which implies that there are two maximal executions starting at $\vec{h}_0$, contradiction assumption that the system is deterministic.

($\Leftarrow$) Suppose all conditions in Lemma 4.1 except that there is a $\vec{h}_0$ for which no infinite execution exists. This implies it is possible to find a finite, maximal execution $\chi(\vec{h}_0)$ that maps $\vec{h}_0$ to $\vec{h}$. If the continuous evolution of $\vec{h}$ is not blocked, it is possible to extend $\chi(\vec{h}_0)$ by continuous evolution contradicting assumption that $\chi(\vec{h}_0)$ is maximal. If on the other hand the continuous evolution of $\vec{h}$ is blocked, it is possible to extend $\chi(\vec{h}_0)$ via a discrete transition contradicting again the assumption that $\chi(\vec{h}_0)$ is maximal. ■

Note that the conditions in Lemma 4.1 require that whenever the system gets into an state where continuous evolution is blocked, it is guaranteed that a discrete transition from that state exists. In the following we state the necessary and sufficient conditions for an IHS to be deterministic. The proof of Lemma 4.2 is not included in this paper for space constraints. However, a proof sketch is provided instead.

Definition 4.3 (Forced Transition Condition) *$\vec{h} \in Reach_{\mathbf{H}^*}$ satisfies the Forced Transition (FT) condition if the following condition holds: If there exists a transition label $\vec{s} \in S_{\vec{q}}$ such that $\vec{h}$ satisfies the corresponding transition guard $G_{\vec{q}}(\vec{s})$, then $\vec{h} \in Out_{\mathbf{H}^*}$.*

Definition 4.4 (Disjoint Transition Guard Condition) *$\vec{h} \in Reach_{\mathbf{H}^*}$ satisfies the Disjoint Transition Guard (DTG) condition the following condition holds: If there exist two discrete transition labels $\vec{s}, \vec{s}\,' \in S_{\vec{q}}$ such that $\vec{s} \neq \vec{s}\,'$, then $\vec{h}$ satisfies at most one of the discrete transition guards $G_{\vec{q}}(\vec{s})$ or $G_{\vec{q}}(\vec{s}\,')$.*

Definition 4.5 (Singleton Transition Map Condition) *$\vec{h} \in Reach_{\mathbf{H}^*}$ satisfies the Singleton Transition Map (STM) condition if the following condition holds: If there exists a discrete transition $\vec{s} \in S_{\vec{q}}$ such that $\vec{h}$ satisfies the transition guard $G_{\vec{q}}(\vec{s})$, then the transition map $Z_{\vec{q}}(\vec{h}, \vec{s})$ contains at most one element.*

Lemma 4.2 (Determinism) *Given an initial condition $\vec{h}_0$, $\chi^M(\vec{h}_0)$ contains at most one element if and only if for all $\vec{h} \in Reach_{\mathbf{H}^*}$ the Forced Transition, the Disjoint Transition Guard, and the Singleton Transition Map conditions are satisfied.*

Proof: (Sketch) ($\Leftarrow$) Suppose there are two maximal executions starting at $\vec{h}_0$ but all FT, DTG, and SMT conditions hold. These two executions $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$ are different and maximal. Since they start at the same initial condition, there is a finite execution $\chi(\vec{h}_0)$ that is the maximal prefix of both $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$. If $\vec{h}$ be the state obtained from $\chi(\vec{h}_0)$ the following cases are possible:

1. Both $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$ evolve continuously from $\vec{h}$. Since the continuous dynamics are Lipschitz (Assumption 4.1), an standard existence and uniqueness argument for continuous dynamical systems implies that $\chi(\vec{h}_0)$ can be extended on continuous evolution and still be a prefix of both $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$ contradicting assumption that $\chi(\vec{h}_0)$ is the maximal prefix of $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$.

2. $\tilde{\chi}(\vec{h}_0)$ evolves continuously from $\vec{h}$, while $\check{\chi}(\vec{h}_0)$ executes a discrete transition. Since $\check{\chi}(\vec{h}_0)$ executes a discrete transition from $\vec{h}$ the FT condition implies that $\vec{h}$ has its continuous evolution blocked. On the other hand, since $\tilde{\chi}(\vec{h}_0)$ evolves continuously from $\vec{h}$ the Lipschitz dynamics imply that the continuous evolution of $\vec{h}$ is not blocked leading to a contradiction.

3. Symmetric to case 2.

4. Both $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$ execute a discrete transition from $\vec{h}$. Then $\vec{h}$ satisfies the guard conditions for both transitions. The DTG condition implies that these transitions are the same, and the STM condition implies that the state of the system after these transitions is identical for both $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$. This implies that $\chi(h_0)$ can be extended by a discrete transition and still be be a prefix of both $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$ contradicting assumption that $\chi(\vec{h}_0)$ is the maximal prefix of $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$.

($\Rightarrow$) Suppose that there is at most one maximal execution but at least one of the FT, DTG, or STM conditions is not satisfied for $\vec{h}$. Since $\vec{h}$ is reachable there exists a finite execution $\chi(\vec{h}_0)$ that maps $\vec{h}_0$ to $\vec{h}$. If the FT condition is violated, the $\vec{h}$ satisfies a transition guard but its continuous evolution is not blocked. Then $\chi(\vec{h}_0)$ can be extended on both continuous evolution and discrete transition, creating two different maximal executions starting at $\vec{h}_0$. Contradiction.

If the DTG condition does not hold, $\vec{h}$ satisfies the transition guards of two different transitions, then $\chi(\vec{h}_0)$ can be extended on two different discrete transitions creating two different maximal executions starting at $\vec{h}_0$. Contradiction.

If the STM condition does not hold, a single discrete transition may lead to two different continuous states, then $\chi(\vec{h}_0)$ can be extended, after a discrete transition, into two different continuous evolutions, creating two different maximal executions starting at $\vec{h}_0$. Contradiction. ■

The conditions in Lemma 4.2 rule out any possibility of the system taking more than one path at the same time: If a discrete transition is possible then continuous evolution is blocked and *vice versa* (FT condition). If there exist two possible transitions then only one of the corresponding transition guards may be completely satisfied (DTG condition). And for every state that may originate a discrete transition there is only one possible destination point after such transition takes place (STM condition)

Combining Lemmas 4.1 and 4.2 we obtain the following result. This holds because an infinite IHE is also maximal.

Theorem 4.1 (Existence and Uniqueness) *Given an initial condition $\vec{h}_0$, $\chi^\infty(\vec{h}_0)$ contains exactly one element if and only if the conditions of Lemmas 4.1 and 4.2 hold.*

Note that Theorem 4.1 states the necessary and sufficient conditions for the existence and uniqueness of an infinite IHE in terms of the global model of the IHS. In order to use these conditions to design the individual agent dynamics such that existence and uniqueness of the multi-agent system's execution is guaranteed globally, we translate the conditions in Theorem 4.1 into a local setup. Let the set of blocked continuous evolution of system $i \in I$ be:

$$Out_i = \{\vec{h} \in H_I; \forall \epsilon > 0, \exists t \in [0, \epsilon) \text{ s.t. } \psi(q_i, t) \notin X_{q_i}\}$$

The following result, which is proved in Section 4.7, states that for the system to have a unique IHE, every agent must be designed such that its discrete transitions are forced, two different discrete transitions are impossible, and a discrete transition can only map the continuous state of the agent to a single location:

Corollary 4.1 (Locally Specified Existence and Uniqueness) *Given an initial condition $\vec{h}_0$, $\chi^\infty(\vec{h}_0)$ contains exactly one element if and only if all the following conditions hold for all agents $i \in I$, and for all $\vec{h} \in Reach_{\mathbf{H}^*}$:*

1. *The interconnected hybrid state $\vec{h} \in Out_i$ if and only if there is a $s \in S_{q_i}$ such that $\vec{h}$ satisfies $G_{q_i}(s)$.*

2. *If there exists two discrete transitions $s, s' \in S_{q_i}$ such that $s \neq s'$ then their corresponding transition guards are disjoint $G_{q_i}(s) \bigcap G_{q_i}(s') = \emptyset$.*

3. *If there is a $s \in S_{q_i}$ such that $\vec{h}$ satisfies $G_{q_i}(s)$ then $Z_{q_i}(\vec{h}, s)$ is a singleton.*

Proof: Condition 1) and definition of Out_i imply that for all $i \in I$, $\forall \epsilon > 0, \exists t \in [0, \epsilon);\ \psi(q_i, t) \notin X_{q_i} \Leftrightarrow \exists s \in S_{q_i}; \vec{h}$ satisfies $G_{q_i}(s)$, this implies that $\forall \epsilon > 0\ \exists i \in I, \exists t \in [0, \epsilon) :\ \psi(q_i, t) \notin X_{q_i} \Leftrightarrow \exists \vec{s} \in S_{\vec{q}} : \vec{h}$ satisfies $G_{\vec{q}}(\vec{s})$, where $\vec{q}$ and $\vec{s}$ are constructed such q_i and s_i are components of $\vec{q}$ and $\vec{s}$ respectively for all $i \in I$ that satisfy Out_i. By definition of $Out_{\mathbf{H}^*}$ it follows

$$\vec{h} \in Out_{\mathbf{H}^*} \Leftrightarrow \exists \vec{s} \in S_{\vec{q}} : \vec{h} \text{ satisfies } G_{\vec{q}}(\vec{s}) \tag{4.1}$$

which shows that Condition 1) implies the conditions in Lemma 4.1 and Definition 4.3 (FT).

To show the reverse direction we note that Lemma 4.1 and Definition 4.3 imply (4.1) in last paragraph. Then the above proof can be reversed to show that Lemma 4.1 and Definition 4.3 imply Condition 1).

Condition 2) implies that any $\vec{h} \in Reach_{\mathbf{H}^*}$ can only satisfy one of the transition guards $G_{q_i}(s)$ or $G_{q_i}(s')$. If we construct $\vec{s}$ and $\vec{s}'$ such that s is component of $\vec{s}$ and s' is component of $\vec{s}'$ then $\vec{h}$ may only satisfy one of the transition guards $G_{\vec{q}}(\vec{s})$ or $G_{\vec{q}}(\vec{s}')$, where q_i is component of $\vec{q}$. This implies Definition 4.4. The reverse direction is proved similarly showing that Condition 2) and Definition 4.4 are equivalent.

Condition 3) says that $Z_{q_i}(\vec{h}, s)$ is a singleton for all $i \in I$. This is equivalent to saying that $Z_{\vec{q}}(\vec{h}, s)$ is a singleton. Therefore Condition 3) and Definition 4.5 are equivalent.

Since Conditions 1), 2), and 3) are together equivalent to conditions in Lemma 4.1 and Definitions 4.3, 4.4, and 4.5, this implies that Conditions 1), 2), and 3) are necessary and sufficient for the existence of a uniqueness of an infinite IHE proving the claim. ■

4.5 Application Example

In this section we use the results of Section 4.4 to design the general structure of the dynamics of the agents in the system discussed in Chapter 3. However, in this part we consider a special case of the system studied in [53], where the topology of the network is fixed, i.e., there are no changes in the nodes or links that form the network. Changes in the network generate event dynamics that are beyond the scope of this chapter.

The dynamics of each software agent and hardware node are described by a hybrid system. I_n denotes the set of hardware nodes, while I_a denotes the set of software agents, and $I = I_n \bigcup I_a$. H_i with $i \in I_n$ denotes the hybrid system that describes node i, while H_k with $k \in I_a$ denotes the hybrid system that describes agent k. The dynamical description of the nodes according to Chapter 3 is the following:

The set of discrete modes $Q_i = O_i \times D_i$, where O_i represents the operating condition of the node that in our discussion is a singleton due to our assumption of using a fixed network, and D_i represents the sets of agents connected to node i. Therefore there is one element $d \in D_i$ for each possible combination of agents that may occupy node i.

The continuous dynamics Σ_i are designed using resource allocation theory [62]. The continuous state space is $X_{q_i} = X_i = \mathbb{R}^N$ for all $q_i \in Q_i$. U_{q_i} is the state space X_k of the agents $k \in I_a$ that occupy node i. Function f_{q_i} is defined in in the previous chapter. We do not include it here because of space constraints. However note that the continuous state of each node $x_i \in X_i$ is inversely related to the availability of resources in such node.

The transition labels in each $S_{q_i} \in \mathbf{S}_i$. reflect the changes in the agents occupying node i. Therefore, if an agent k is located at node i, and migrates to node i', a discrete transition in the node's dynamics $s \in S_{q_i}$ must occur to update the connectivity state $d \in D_i$ so the set of agents occupying node i is updated. Likewise, if an agent arrives to node i a discrete transition $s \in S_{q_i}$ must occur to update $d \in D_i$.

A discrete transition $s \in S_{q_i}$ occurs when node i detects that an agent has arrived or left. Therefore transition s may be enabled according to the discrete states of the agents as follows: A node's transition guard $G_{q_i}(s)$ is satisfied if the elements of the neighborhood of the destination state q_i' are equal to the set of agents that are connected to node i. In this form, as soon as an agent arrives/leaves node i, this nodes changes its discrete state to reflect the new set of agents it hosts. An graphical example of this behavior is shown in the discrete interaction of Figure 4.1.

The discrete transition map $Z_{q_i}(\vec{h}, s)$ leaves the continuous state unchanged for all $\vec{h} \in H_I$ and all $s \in S_{q_i}$.

The dynamical description of the agents is the following: The set of discrete modes $Q_k = O_k \times D_k$, where O_k is a singleton because agents are assumed to have a single operating condition, and D_k represents the set network nodes that may be connected to agent k. Therefore there is one element $d \in D_k$ for each possible location of agent k in the network.

The state space is $X_{q_k} = X_k = \mathbb{R}^N$ for all $q_k \in Q_k$. The continuous controls U_{q_k} are the state space X_i of the node $i \in I_n$ where agent is located when q_k is in force. Function f_{q_k} is defined in Chapter 3.

The transition labels in each $S_{q_k} \in \mathbf{S}_k$. must reflect the ability of agent k to change its location. Therefore, if agent k is located at node i, and migrates to node i', a discrete transition $s \in S_{q_k}$ must occur to update $d \in D_k$.

The transition guards $G_{q_k} \in \mathbf{G}_k$ must be designed such that the transition improves the resources available for the agent, and the agent always migrates to the node with best resources among those in its neighborhood. An example of a transition guard that follows these guidelines is defined as: Assume agent k has a current discrete state q_k, which corresponds to agent k located at node i. Given $s \in S_{q_k}$ let the destination state of transition s be q'_k, which corresponds to agent k located at node i', then $\vec{h}$ satisfies $G_{q_k}(s)$ if

$$\mathrm{mig}(x_{i'}, i') < \mathrm{nmig}(x_i, i) \text{ and} \tag{4.2a}$$

$$\mathrm{mig}(x_{i'}, i') < \mathrm{mig}(x_j, j), \;\; \forall j \neq i'; j \in V(q_k) \tag{4.2b}$$

where $\mathrm{mig}(\alpha, \beta)$ and $\mathrm{nmig}(\alpha, \beta)$ are monotonically increasing functions of α for all $\beta \in I$. The function $\mathrm{mig}(\alpha, \beta)$ measures the benefit of migrating to node β, whose current continuous state is α. Similarly $\mathrm{nmig}(\alpha, \beta)$ measures the benefit of staying at node β, whose current continuous state is α. Therefore, since the states x_i, $x_{i'}$, and x_j are inversely related to the availability of resources in nodes i, i', and j respectively as discussed in the previous chapter, the transition guard that allows agent k to migrate from node i to node i' is satisfied when 1) the benefit of migrating to node i' is better than staying at node i, and 2) the benefit of migrating to node i' is better than migrating to any other node j that is reachable by agent k in one transition. This is illustrated in Figure 4.3.

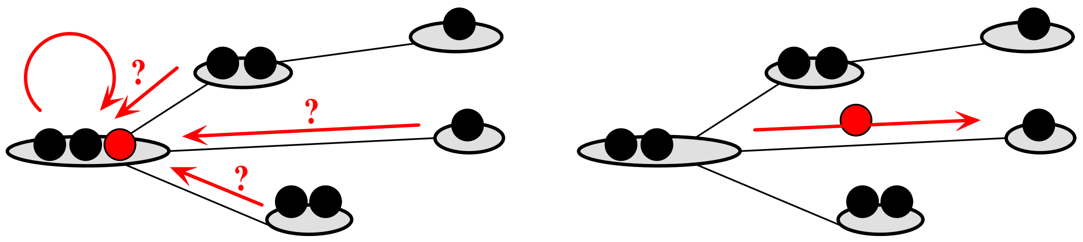

Figure 4.3: Agent decision process. The agent compares the benefit of staying in its current location with that of migrating to different nodes in its neighborhood (Arrows with ?-marks; left). Then the agent migrates to (or stays at) the node that offers the best benefit (arrow with agent; right).

Finally the discrete transition map $Z_{q_k}(\vec{h}, s)$ sets the continuous state after the transition $x_{q'_k} = 0$ for all $\vec{h} \in H_I$ and all $s \in S_{q_i}$. This is done to reflect that whenever agent k arrives to new node, it starts with no resources.

We now use Corollary 4.1 to determine wether any agent can contribute to creating multiple IHE starting from the same initial condition. Condition 1) is not satisfied in our discussions above. However if Out_k is defined as the union of of all the transition guards on agent k, i.e.

$$Out_k \triangleq \{\vec{h} \in H_I; \vec{h} \,\text{satisfies}\, G_{q_k}(s), \forall s \in S_{q_k} \text{ where } q_k \text{ is a component of } \vec{h}\}$$

then Condition 1) in Corollary 4.1 is satisfied, because as soon as a transition guard is enabled the continuous evolution is blocked. Therefore, as soon as a transition guard is satisfied in the agents dynamics, the transition occurs.

Condition 2) is satisfied by the definition of the transition guard G_{q_k}. To see this assume there is a $\vec{h} \in Reach_{\mathbf{H}^*}$ that satisfies the guards $G_{q_k}(s)$, and $G_{q_k}(s')$ of two different transitions s and s' in S_{q_k}. We denote as c the current node of agent k, and with some abuse of notation we denote as s the node that agent k reaches if it takes transition s

and s' if it takes transition s'. Then note that since the two transition guards are satisfied by $\vec{h}$, then according to (4.2b) $\mathrm{mig}(x_s, s) < \mathrm{mig}(x_j, j)$ for all $j \in V(q_k)$ including s', and also $\mathrm{mig}(x_{s'}, s') < \mathrm{mig}(x_j, j)$ for all $j \in V(q_k)$ including s, which implies that $\mathrm{mig}(x_s, s) < \mathrm{mig}(x_{s'}, s')$ and $\mathrm{mig}(x_{s'}, s') < \mathrm{mig}(x_s, s)$, which is a contradiction.

Finally, condition 3) is satisfied by the definition Z_{q_k} for all $k \in I_a$. A similar analysis can be performed on the hybrid model of the nodes to reach the conclusion that these satisfy all conditions in Corollary 4.1. Note that we have used Corollary 4.1 to analyze the model of the each agent in the system individually, but we can reach the conclusion the complete system always has a unique interconnected hybrid execution given an initial condition.

4.6 Conclusion

We present an interconnected hybrid systems framework: a set of hybrid systems with interweaved continuous and discrete dynamics that form a multi-agent system with hybrid interacting dynamics. We extend the work in [37] defining reachable sets and executions for interconnected hybrid systems. We prove necessary and sufficient conditions for the existence and uniqueness of interconnected hybrid executions that are written in terms of the local model of each hybrid agent.

We apply these conditions to the the design of the discrete dynamics of agents and nodes in the resource allocation problem discussed Chapter 3. These conditions allow us verify that the dynamics of each agent and node yield an interconnected hybrid system that is "well behaved" in terms of existence and uniqueness of executions.

4.7 Proofs

4.7.1 Preliminaries

An IHTT τ is linearly ordered by the relation $\prec$ defined as follows: Let $t_1 \in [\tau_i, \tau_{i+1}]$ and $t_2 \in [\tau_j, \tau_{j+1}]$ then $t_1 \prec t_2$ if $t_1 < t_2$ or $i < j$. We say $\tau = \{\tau^0, \tau^1, \ldots, \tau^N\}$ is a prefix of $\tilde{\tau} = \{\tilde{\tau}^0, \tilde{\tau}^1, \ldots, \tilde{\tau}^{\tilde{N}}\}$ (written $\tau \sqsubseteq \tilde{\tau}$) if either they are identical, or τ is finite, $N \leq \tilde{N}$, $\tau_n = \tilde{\tau}_n$ for all $n \in \{0, 1, \ldots, N-1\}$, and $[\tau_{N-1}, \tau_N) \subseteq [\tilde{\tau}_{N-1}, \tilde{\tau}_N)$.

We say that an IHE $\chi(\vec{h}_0) = (\tau, \mathbf{q}, \mathbf{s}, \mathbf{x}, \mathbf{u})$ with $N+1$ elements is a prefix of another IHE $\tilde{\chi}(\vec{h}_0) = (\tilde{\tau}, \tilde{\mathbf{q}}, \tilde{\mathbf{s}}, \tilde{\mathbf{x}}, \tilde{\mathbf{u}})$ with $\tilde{N}+1$ elements (written $\chi(\vec{h}_0) \sqsubseteq \tilde{\chi}(\vec{h}_0)$) if $\tau \sqsubseteq \tilde{\tau}$, and for all $n \in \{0, 1, \ldots, N\}$ and for all $t \in [\tau^{n-1}, \tau^n[$ $(\vec{q}^n, \vec{s}^n, \vec{x}^n(t), \vec{u}^n(t)) = (\vec{\tilde{q}}^n, \vec{\tilde{s}}^n, \vec{\tilde{x}}^n(t), \vec{\tilde{u}}^n(t))$. We say that $\chi(\vec{h}_0)$ is a strict prefix of $\tilde{\chi}(\vec{h}_0)$ (written $\chi(\vec{h}_0) \sqsubset \tilde{\chi}(\vec{h}_0)$) if $\chi(\vec{h}_0) \sqsubseteq \tilde{\chi}(\vec{h}_0)$, and $\chi(\vec{h}_0) \neq \tilde{\chi}(\vec{h}_0)$.

An IHE $\chi(\vec{h}_0)$ is called maximal if it is not a strict prefix of any other execution. An IHE $\chi(\vec{h}_0)$ is finite if τ is a finite sequence and the last elements of $\mathbf{u}$ and $\mathbf{x}$ are defined over compact intervals of time, i.e. $\vec{u}^N : [\tau^{N-1}, \tau^N] \to \prod_{i \in I} U_{q_i^n}$, and $\vec{x}^N : [\tau^{N-1}, \tau^N] \to \prod_{i \in I} X_{q_i^n}$. $\chi(\vec{h}_0)$ is infinite if τ is an infinite sequence or if $\tau^N = \infty$. The following result is proved in [54].

Lemma 4.3 *Assumption 4.1 implies that* $\vec{f}_{\vec{q}}(\vec{x}_{\vec{q}}, t) = \left(f_{q_i}(\cdot)\right)_{i \in I}$ *is globally Lipschitz on* $\vec{x}_{\vec{q}}$ *for all* $\vec{q} \in Q_I$.

4.7.2 Proof of Lemma 4.1

$(\Rightarrow)$ Suppose that $\mathbf{H}^*$ is deterministic, and for any $\vec{h}_0$ $\chi^{\infty}(\vec{h}_0)$ is nonempty, but there is a $\vec{h} \in Reach_{\mathbf{H}^*} \bigcap Out_{\mathbf{H}^*}$ for which there is no $\vec{s} \in S_{\vec{q}}$ such that $\vec{h}$ satisfies $G_{\vec{q}}(\vec{s})$. Since $\vec{h} \in Reach_{\mathbf{H}^*}$ there is a finite execution $\chi(\vec{h}_0) \in \chi^F(\vec{h}_0, \mathcal{E}^*)$ such that $\tau = \{\tau^0, \tau^1, \ldots, \tau^N\}$ and $\vec{h} = (\vec{q}^N, \vec{x}^N(\tau^N))$.

$a)$ Suppose there exists another execution $\check{\chi}(\vec{h}_0)$ that extends $\chi(\vec{h}_0)$ such that $\chi(\vec{h}_0) \sqsubseteq \check{\chi}(\vec{h}_0)$ and $\check{\tau} = \{\tau^0, \tau^1, \ldots, \tau^{N-1}, \tau^N + \epsilon\}$ for some $\epsilon > 0$ (Lemma 4.3 makes this possible). Then there exists $t \in [0, \epsilon)$ such that $\psi(q_i, t) \in X_{q_i}$ for all $i \in I$, which violates $\vec{h} \in Out_{\mathbf{H}^*}$ leading to a contradiction.

$b)$ Suppose there exists another execution $\check{\chi}(\vec{h}_0)$ such that $\chi(\vec{h}_0) \sqsubseteq \check{\chi}(\vec{h}_0)$ and $\check{\tau} = \{\tau^0, \tau^1, \ldots, \tau^N, \check{\tau}^{N+1}\}$, then there exists $\mathbf{H}^*$ executes a discrete transition $\vec{s}^N$ at τ^N, therefore (Definition 4.2) there exists a $\vec{s} \in S_{\vec{q}^{N-1}}$ such that $\vec{h}^{N-1}(\tau^N)$ satisfy $G_{\vec{q}^{N-1}}(\vec{s})$, where $\vec{q}^n$ is the discrete state of $\vec{h}^n$. Note that this violates the above assumption that there is no $\vec{s} \in S_{\vec{q}}$ such that $\vec{h}$ satisfies $G_{\vec{q}}(\vec{s})$ leading to a contradiction.

$a)$ and $b)$ imply that $\chi(\vec{h}_0)$ is maximal. However by assumption $\chi^\infty(\vec{h}_0)$ is nonempty, therefore there exists an infinite execution $\tilde{\chi}(\vec{h}_0) \in \chi^\infty(\vec{h}_0)$. This execution is also maximal and different from $\chi(\vec{h}_0)$, which implies that $\chi^M(\vec{h}_0, \mathcal{E}^*)$ has at least two different elements contradicting that $\mathbf{H}^*$ is deterministic, which proves the $(\Rightarrow)$ part of our claim.

$(\Leftarrow)$ Suppose there is an initial condition $\vec{h}_0$ for which $\chi^\infty(\vec{h}_0)$ is empty, but for all $\vec{h} \in Reach_{\mathbf{H}^*} \bigcap Out_{\mathbf{H}^*}$ there is a $\vec{s} \in S_{\vec{q}}$ such that $\vec{h}$ satisfies $G_{\vec{q}}(\vec{s})$.

Since $\chi^\infty(\vec{h}_0)$ is empty, it is possible to find a finite, maximal execution $\chi(\vec{h}_0) \in \chi^F(\vec{h}_0) \bigcap \chi^M(\vec{h}_0)$ that maps $\vec{h}_0$ to $\vec{h}$. $\chi(\vec{h}_0) \in \chi^F(\vec{h}_0)$ implies that the last elements of $\mathbf{x}$ and $\mathbf{u}$ are defined over the compact interval $[\tau^{N-1}, \tau^N]$, i.e., $\vec{x}^N : [\tau^{N-1}, \tau^N] \to \prod_{i \in I} X_{q_i^N}$ and $\vec{u}^N : [\tau^{N-1}, \tau^N] \to \prod_{i \in I} U_{q_i^N}$.

By assumption $\vec{h}^N \in Reach_{\mathbf{H}^*}$. If $\vec{h}^N \notin Out_{\mathbf{H}^*}$, then there exists $\epsilon > 0$ such that for all $t \in [0, \epsilon)$ $\psi(q_i^N, t) \in X_{q_i}$ for all $i \in I$. This implies that $\chi(\vec{h}_0)$ can be extended to $\tilde{\chi}(\vec{h}_0)$ such that $\chi(\vec{h}_0) \sqsubseteq \tilde{\chi}(\vec{h}_0)$, with $\tilde{\tau} = \{\tau^0, \tau^1, ..., \tau^N + \epsilon\}$, and $\tilde{\vec{x}}^N$ and $\tilde{\vec{u}}^N$ defined over the interval $[\tau^{N-1}, \tau^N + \epsilon)$. Therefore $\chi(\vec{h}_0)$ is not maximal, which is a contradiction.

If $\vec{h}^N \in Out_{\mathbf{H}^*}$, then by assumption there is a $\vec{s} \in S_{\vec{q}}$ such that $\vec{h}$ satisfies $G_{\vec{q}}(\vec{s})$. Then $\chi(\vec{h}_0)$ can be extended through a discrete transition to $\tilde{\chi}(\vec{h}_0)$ such that $\chi(\vec{h}_0) \sqsubseteq \tilde{\chi}(\vec{h}_0)$ where $\tilde{\chi}(\vec{h}_0) = (\tilde{\tau}, \tilde{\mathbf{q}}, \tilde{\mathbf{s}}, \tilde{\mathbf{x}}, \tilde{\mathbf{u}}) = (\tau, \mathbf{q}, \mathbf{s}, \mathbf{x}, \mathbf{u}) : (\tilde{\tau}^{N+1}, \tilde{\vec{q}}^{N+1}, \tilde{\vec{s}}^{N+1}, \tilde{\vec{x}}^{N+1}, \tilde{\vec{u}}^{N+1})$ where

$\tilde{\vec{q}}^{N+1} = \vec{s} \in S_{\vec{q}^N}$ and $\tilde{\vec{x}}_{\tilde{\vec{q}}^{N+1}}(\tilde{\tau}^{N+1}) \in Z_{\vec{q}^N}(G_{\vec{q}^N}, \vec{s})$. Therefore $\chi(\vec{h}_0)$ is not maximal, again a contradiction.

4.7.3 Proof of Lemma 4.2

($\Leftarrow$) Suppose that $\chi^M(\vec{h}_0)$ contains at least two elements but all FT, DTG, and SMT conditions hold. Then there exist $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$ such that $\tilde{\chi}(\vec{h}_0) \neq \check{\chi}(\vec{h}_0)$ and both are maximal i.e. $\tilde{\chi}(\vec{h}_0), \check{\chi}(\vec{h}_0) \in \chi^M(\vec{h}_0)$.

Since both executions start at the same initial condition $\vec{h}_0$ there exists an IHE $\chi(\vec{h}_0)$ that is a maximal prefix of both $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$. Moreover $\chi(\vec{h}_0)$ is finite because $\tilde{\chi}(\vec{h}_0) \neq \check{\chi}(\vec{h}_0)$.

Let $\vec{h}^N$ be the state of $\mathbf{H}^*$ that is obtained from $\chi(\vec{h}_0)$ with initial condition $\vec{h}_0$. Since $\chi(\vec{h}_0)$ is finite, $\vec{x}^N$ and $\vec{u}^N$ are defined over the compact interval $[\tau^{N-1}, \tau^N]$. At this point the following cases are possible:

1. $\tau^N \notin \tilde{\tau}$ and $\tau^N \notin \check{\tau}$, therefore both $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$ evolve from $\vec{h}^N$ on the system's continuous dynamics (This case establishes the sufficiency part of Lemma 4.3). By definition (Definitions 4.1-4.2, and Lemma 4.3) and standard existence and uniqueness argument for continuous dynamical systems, there exists $\epsilon > 0$ such that for all $t \in [0, \epsilon)$ $\psi(q_i^N, t) \in X_{q_i}$ for all $i \in I$. Therefore there exists $\bar{\chi}(\vec{h}_0) = (\bar{\tau}, \bar{\mathbf{q}}, \bar{\mathbf{s}}, \bar{\mathbf{x}}, \bar{\mathbf{u}})$ where $\bar{\tau} = \{\tau^0, \tau^1, \ldots, \tau^{N-1}, \tau^N + \epsilon\}$, $\bar{\vec{x}}^N$ and $\bar{\vec{u}}^N$ are defined over $[\tau^{N-1}, \tau^N + \epsilon)$, AND $\chi(\vec{h}_0) \sqsubset \bar{\chi}(\vec{h}_0) \sqsubseteq \tilde{\chi}(\vec{h}_0)$ and $\chi(\vec{h}_0) \sqsubset \bar{\chi}(\vec{h}_0) \sqsubseteq \check{\chi}(\vec{h}_0)$ which contradicts discussion about $\chi(\vec{h}_0)$ being the maximal prefix of $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$.

2. $\tau^N \notin \tilde{\tau}$ and $\tau^N \in \check{\tau}$, therefore $\tilde{\chi}(\vec{h}_0)$ evolves from $\vec{h}^N$ on the system's continuous dynamics, while $\check{\chi}(\vec{h}_0)$ executes a discrete transition from $\vec{h}^N$ (This establishes sufficiency of the FT condition).

Since $\check{\chi}(\vec{h}_0)$ executes a discrete transition from $\vec{h}^N$ there exists $\vec{s} \in S_{\vec{q}^N}$ such that $\vec{q}^{N+1} = \vec{s}$, $\vec{h}^N(\tau^N)$ satisfies $G_{\vec{q}^N}(\vec{s})$. Then by the FT condition $\vec{h}^N \in Out_{\mathbf{H}^*}$. On the other hand, since $\tau^N \notin \tilde{\tau}$, (Definitions 4.1-4.2, Lemma 4.3, and standard existence and uniqueness argument for continuous dynamical systems) there exists $\epsilon > 0$ such that for all $t \in [0, \epsilon)$ $\psi(q_i^N, t) \in X_{q_i}$ for all $i \in I$, which implies that $\vec{h}^N \notin Out_{\mathbf{H}^*}$ contradicting the previous conclusion.

3. $\tau^N \in \tilde{\tau}$ and $\tau^N \notin \check{\tau}$. Symmetric to case 2.

4. $\tau^N \in \tilde{\tau}$ and $\tau^N \in \check{\tau}$, therefore both $\tilde{\chi}(\vec{h}_0)$ and $\check{\chi}(\vec{h}_0)$ execute a discrete transition from $\vec{h}^N$. (This case establishes the sufficiency of DTG and STM conditions).

   Since $\tilde{\chi}(\vec{h}_0)$ executes discrete transition, then from definition of IHE, there is $\tilde{\vec{s}} \in S_{\vec{q}^N}$ such that $\vec{h}^N(\tau^N)$ satisfies $G_{\vec{q}^N}(\tilde{\vec{s}})$. By assumption, there also exists a $\check{\vec{s}} \in S_{\vec{q}^N}$, such that $\vec{h}^N(\tau^N)$ satisfies $G_{\vec{q}^N}(\check{\vec{s}})$.

   Since $\vec{h}^N$ satisfies the guard conditions for both $\tilde{\vec{s}}$ and $\check{\vec{s}}$, then by DTG condition that $\tilde{\vec{s}} = \check{\vec{s}}$. Then by IHE definition $\tilde{\vec{q}}^{N+1} = \check{\vec{q}}^{N+1}$, which by the STM condition implies that $\tilde{\vec{x}}^{N+1}(\tau^N) = \check{\vec{x}}^{N+1}(\tau^N)$. Therefore $\chi(\vec{h}_0)$ can be extended to $\bar{\chi}(\vec{h}_0) = \chi(\vec{h}_0, \mathcal{E}^*) : (\bar{\tau}^{N+1}, \bar{\vec{q}}^{N+1}, \bar{\vec{s}}^{N+1}, \bar{\vec{x}}^{N+1}, \bar{\vec{s}}^{N+1})$ where $\chi(\vec{h}_0) \sqsubset \bar{\chi}(\vec{h}_0) \sqsubseteq \tilde{\chi}(\vec{h}_0)$, $\chi(\vec{h}_0) \sqsubset \bar{\chi}(\vec{h}_0) \sqsubseteq \check{\chi}(\vec{h}_0)$, which contradicts discussion about $\chi(\vec{h}_0)$ being the maximal prefix of both $\tilde{\chi}(\vec{h}_0)$, and $\check{\chi}(\vec{h}_0)$.

All previous cases prove the $(\Leftarrow)$ part of the claim.

$(\Rightarrow)$ Suppose for the sake of contradiction that $\chi^M(\vec{h}_0)$ contains at most one element, but that at least one of the FT, DTG, or STM conditions is not satisfied for $\vec{h}$. Since $\vec{h} \in Reach_{\mathbf{H}^*}$ there exists a finite execution $\chi(\vec{h}_0) \in \chi^F(\vec{h}_0)$ such that $\vec{h} = \vec{h}^N(\tau^N) = (\vec{q}^N, \vec{x}^N(\tau^N))$ where $\vec{q}^N$ and $\vec{x}^N$ are the last elements of $\mathbf{q}$ and $\mathbf{x}$ respectively, and $\vec{x}^N$ and $\vec{u}^N$ are defined over the compact interval $[\tau^{N-1}, \tau^N]$.

If the FT condition is violated, there exists $\vec{s} \in S_{\vec{q}^N}$ such that $\vec{h}^N(\tau^N)$ satisfies $G_{\vec{q}}(\vec{s})$, but $\vec{h}^N(\tau^N) \notin Out_{\mathbf{H}^*}$. Therefore $\chi(\vec{h}_0)$ can be extended with either a discrete transition or continuous evolution: In case of the discrete transition consider $\vec{\check{q}}^{N+1} = \vec{s}$ then there exists $\check{\chi}(\vec{h}_0) = (\tau, \mathbf{q}, \mathbf{s}, \mathbf{x}, \mathbf{u}) : (\check{\tau}^{N+1}, \vec{\check{q}}^{N+1}, \vec{\check{s}}^{N+1}, \vec{\check{x}}^{N+1}, \vec{\check{u}}^{N+1})$. In case of the continuous evolution there is an $\epsilon > 0$ such that for all $t \in [0, \epsilon)$, $\psi(q_i^N, t) \in X_{q_i^N}$ for all $i \in I$ and . Then there exists $\tilde{\chi}(\vec{h}_0) = (\tau^0, \vec{q}^0, \vec{s}^0, \vec{x}^0, \vec{u}^0), \ldots, (\tau^{N-1}, \vec{q}^{N-1}, \vec{s}^{N-1}, \vec{x}^{N-1}, \vec{u}^{N-1})$, $(\tau^N + \epsilon, \vec{q}^N, \vec{s}^N, \vec{x}^N, \vec{u}^N)$ where $\vec{x}^N$ and $\vec{u}^N$ are defined over $[\tau^{N-1}, \tau^N + \epsilon)$. Thus $\chi(\vec{h}_0) \sqsubset \check{\chi}(\vec{h}_0)$ and $\chi(\vec{h}_0) \sqsubset \tilde{\chi}(\vec{h}_0)$, and $\check{\chi}(\vec{h}_0) \neq \tilde{\chi}(\vec{h}_0)$ which implies that there is at least two maximal executions in $\chi^M(\vec{h}_0)$ which contradicts assumption, therefore FT must hold.

If the DTG condition does not hold, there exists $\vec{s}, \vec{s}\,' \in S_{\vec{q}^N}$ with $\vec{s} \neq \vec{s}\,'$ such that $\vec{h}^N$ satisfies both $G_{\vec{q}^N}(\vec{s})$ and $G_{\vec{q}^N}(\vec{s}\,')$ simultaneously. Therefore $\chi(\vec{h}_0^N)$ can be extended on two different discrete transitions $\vec{\check{q}}^{N+1} = \vec{s}$ and $\vec{\tilde{q}}^{N+1} = \vec{s}\,'$, where $\vec{\check{q}}^{N+1} \neq \vec{\tilde{q}}^{N+1}$. Then there exist two interconnected hybrid executions $\check{\chi}(\vec{h}_0^N) \neq \tilde{\chi}(\vec{h}_0^N)$ where $\check{\chi}(\vec{h}_0^N) = (\tau, \mathbf{q}, \mathbf{s}, \mathbf{x}, \mathbf{u}) : (\check{\tau}^{N+1}, \vec{\check{q}}^{N+1}, \vec{\check{s}}^{N+1}, \vec{\check{x}}^{N+1}, \vec{\check{u}}^{N+1})$ and $\tilde{\chi}(\vec{h}_0^N) = (\tau, \mathbf{q}, \mathbf{s}, \mathbf{x}, \mathbf{u}) : (\tilde{\tau}^{N+1}, \vec{\tilde{q}}^{N+1}, \vec{\tilde{s}}^{N+1}, \vec{\tilde{x}}^{N+1}, \vec{\tilde{u}}^{N+1})$. Note that $\chi(\vec{h}_0^N) \sqsubset \check{\chi}(\vec{h}_0^N)$ and $\chi(\vec{h}_0^N) \sqsubset \tilde{\chi}(\vec{h}_0^N)$ so there exist at least two maximal execution in $\chi^M(\vec{h}_0^N)$ contradicting our assumption. Thus the DTG condition must hold.

If the STM condition does not hold for $\vec{h} = \vec{h}^N(\tau^N)$, there exists $\vec{s} \in S_{\vec{q}^N}$ such that $\vec{h}$ satisfies $G_{\vec{q}}(\vec{s})$. Since $Z_{\vec{q}^N}(\vec{h}, \vec{s})$ contains at least two elements, $\chi(\vec{h}_0)$ may be extended to $\check{\chi}(\vec{h}_0) = (\tau, \mathbf{q}, \mathbf{s}, \mathbf{x}, \mathbf{u}) : (\check{\tau}^{N+1}, \vec{\check{q}}^{N+1}, \vec{\check{s}}^{N+1}, \vec{\check{x}}^{N+1}, \vec{\check{u}}^{N+1})$ as well as to $\tilde{\chi}(\vec{h}_0) = (\tau, \mathbf{q}, \mathbf{s}, \mathbf{x}, \mathbf{u}) : (\tilde{\tau}^{N+1}, \vec{\tilde{q}}^{N+1}, \vec{\tilde{s}}^{N+1}, \vec{\tilde{x}}^{N+1}, \vec{\tilde{u}}^{N+1})$ where $\vec{\check{q}}^{N+1} = \vec{\tilde{q}}^{N+1}$ but $\vec{\check{x}}^{N+1} \neq \vec{\tilde{x}}^{N+1} \in Z_{\vec{q}^N}(\vec{h}, \vec{s})$. This implies $\chi(\vec{h}_0) \sqsubset \check{\chi}(\vec{h}_0^N)$ and $\chi(\vec{h}_0) \sqsubset \tilde{\chi}(\vec{h}_0^N)$. Since $\check{\chi}(\vec{h}_0) \neq \tilde{\chi}(\vec{h}_0)$, $\chi^M(\vec{h}_0)$ contains at least two elements contradicting our assumption. Therefore STM must hold. This completes the proof of the sufficiency part.

Chapter 5

Randomized Algorithms for Optimal Control of Hybrid Systems

5.1 Introduction

The previous chapter provides tools to define the general structure of the discrete dynamics for our resource allocation problem such that the systems is well behaved in terms of existence and uniqueness of its executions. However our final goal is to design the dynamics of this system such that the agents converge to a point that is as close as possible to the optimal distribution of agents in the network. Towards this end we explore a randomized approach for the optimization of individual hybrid systems' first, and for the optimization problem.

The basic approach is to generate samples from the family of possible solutions, and to test them on the plant's model to evaluate their performance. This result is obtained by first presenting the general hybrid optimal control problem, and then converting it into an optimization problem within a statistical learning framework. The results are first applied to examples already existing in the literature in order to highlight certain operational aspects of the proposed methods, and then to the resource allocation problem discussed along this dissertation.

The motivation for exploring this technique for hybrid optimal control comes from the complexity of the resource allocation model: The analytical complexity of randomized optimization (a model-free technique) does not grow with the complexity of the systems model, which is not true for model based techniques. In fact in order to obtain hybrid extensions of model based techniques like gradient optimization [7, 8, 10, 18, 79], dynamic programming theory [26], or the Pontryagin's Minimum Principle [60, 63] various restrictions have been imposed in the hybrid model in order to obtain meaningful, and computationally feasible results.

One approach is to limit the control inputs to the discrete domain. Thus, a performance function is optimized by choosing the modal sequence and the corresponding switching times. This idea was proposed in [18, 79] restricting the attention to the switching times using a fixed modal sequence. The result in [18] was later modified [3] to vary the sequence by iteratively inserting new modes and optimizing a new fixed sequence until no further improvements were achieved. The same idea was pursued in [8], with the system restricted to two modes only, and in [7] with the attention concentrated on systems with linear continuous dynamics ([3, 8, 18, 79] dealt with nonlinear dynamics). Other methods use non-smooth optimization [52], switching surfaces optimization [10], model predictive control for optimization of continuous inputs [6], and game theoretic approaches [73].

An alternative to classical methods is provided by statistical learning and randomized algorithms [72, 75]. The objective of such methods is to optimize a performance measure *on average*, guaranteeing that the error between the obtained solution and the optimal one is *arbitrarily small* with probability *arbitrarily close to one*. This is achieved by *sampling* the set of potential solutions, and choosing the sample that yields the best performance. Randomized techniques have been previously used to solve some robust control problems that are NP-hard [34, 76] and for reachability analysis of hybrid systems [9]. The advantage of statistical methods is that they simplify the analysis and design tasks at the cost of not being able to guarantee the optimality of the solution.

We show in this chapter that randomized approaches can also be pursued to study optimal control of hybrid systems, as previously discussed in [15] using a simulation example. We present a procedure to use randomized algorithms, obtained in a manner similar to [76], to solve a class of optimization problems for hybrid systems that may include discrete and continuous control inputs, as well as autonomous or controlled transitions between modes. An expression that relates the desired properties of the solution to the number of samples needed to obtain them is provided as a mechanism for controlling the trade-off between the computational complexity (number of samples), and the performance of the solution. The algorithm is tested in simulations involving individual hybrid systems, and compared to results obtained via gradient procedures, showing that one advantage that the randomized approach (a model-free technique) has over model-based techniques (e.g. gradient techniques) is that the theoretical analysis is simpler, making it applicable for complex situations, at the price of obtaining a solution that is not guaranteed to be *the optimal*. The same technique is then applied to the design of the discrete dynamics of the agents showing that for an special case of the problem considered in Chapter 3, the agents are capable of obtaining better resources from the network by migrating between discrete locations.

5.2 A Hybrid Optimal Control Problem

The hybrid optimal control problem we consider here uses the complete model introduced in Chapter 2, and is similar to the one defined in [12]. Let $\{l^n\}_{n\in\langle\tau\rangle}$ be a family of continuous flow cost functions with $l^n : X_{q^n} \times U_{q^n} \to \mathbb{R}^+$. Let $\{c^n\}_{n\in\langle\tau\rangle}$ be a family of discrete jump costs with $c^n : G_{q^n} \times S_{q^n} \to \mathbb{R}^+$. Let c_f be the terminal cost function

$c_f : Q \times \{X_q\}_{q \in Q} \to \mathbb{R}^+$. Let the hybrid cost function $J(t_0, t_f, h_0, \mathcal{I}) = J(\mathcal{I})$ be:

$$J(\mathcal{I}) := \sum_{n=0}^{L} \Big(\int_{\tau^n}^{\tau^{n+1}} l^n(s)ds + c^n \Big) + c_f \tag{5.1}$$

where $[t_0, t_f]$ is the optimization interval (assumed finite), $L \leq N$ is the number of discrete transitions that occur during the optimization interval, h_0 is the initial condition of the system, and $\mathcal{I}$ is the hybrid input. The optimal control problem can then be stated as:

Problem 5.1 (Hybrid Optimal Control Problem (HOCP)) *Given a system* $\mathbf{H}$ *(Definition 2.1) with initial condition* h_0*, hybrid execution* $\chi(h_0)$ *(Definition 2.4), and optimization interval* $[t_0, t_f]$*, the Hybrid Optimal Control Problem (HOCP) is to minimize the total cost* (5.1)*, over the family of input trajectories* $\{\mathcal{I}\}$*. If a hybrid input* $\mathcal{I}^*$ *minimizes* $J(\mathcal{I})$*, then it is called a hybrid optimal control for* $\mathbf{H}$.

Note that the optimal control problem defined here is for a finite interval of time. Similar definition can be possed for an infinite time interval using discounted costs as in [12].

5.3 Statistical Learning Theory

Suppose you have a system with decision vector $y \in \mathcal{Y}$ and cost functional $J : \mathcal{Y} \to \mathbb{R}$, and consider the problem of estimating the best performance of the system $J^* = \inf_{y \in \mathcal{Y}} J(y)$. For this purpose N_s independent and identically distributed (i.i.d.) random samples $y_i;\ i = 1, ..., N_s$ are taken from $\mathcal{Y}$ according to a probability distribution $P(y)$. Then the sample minimum is defined as $J^0 = \min_{i=1,...,N_s} J(y_i)$, and the objective becomes that of making J^0 as close as possible to J^* [72]. This leads to:

Definition 5.1 (Probable Near Minimum) *Given* $J(y)$*,* $\delta \in (0,1)$*,* $\alpha \in (0,1)$*, a number* $J_0 \in \mathbb{R}$ *is said to be a probable near minimum of* $J(y)$ *to level* α *and confidence* $1 - \delta$ *if* *there exists a set $\tilde{\mathcal{Y}} \subseteq \mathcal{Y}$ with $Pr\{\tilde{\mathcal{Y}}\} \leq \alpha$ such that*

$$Pr\Big\{ \inf_{\mathcal{Y}} J(y) \leq J_0 \leq \inf_{\mathcal{Y}\setminus\tilde{\mathcal{Y}}} J(y) \Big\} \geq 1 - \delta \tag{5.2}$$

Loosely speaking, the level α describes a set of potential solutions that may not be represented in the samples taken for optimization. If this set is made small there will be a small probability of finding a better solution than the sample minimum. The confidence $(1 - \delta)$ describes the probability of obtaining the desired level. Definition 5.1 then implies that if α and δ are made small (but different than zero), the probability of finding a better solution will be small (α), with high confidence $(1 - \delta)$.

The mechanism to control the level and confidence of the sample minimum is the number of samples N_s taken from the set of possible solutions. In order to obtain a probable near minimum (Definition 5.1) from the samples, N_s can be obtained as a function of α and δ using the following result:

Theorem 5.1 (Minimum number of samples) *The minimum number of samples N_s that guarantee that J^0 is a probable near minimum to level α and confidence δ of $J(y)$ is*

$$N_s \geq \frac{\ln(1/\delta)}{\ln(1/(1-\alpha))} \tag{5.3}$$

In words, this result estates that for a sufficiently large number of samples N_s, the probability that the sample minimum J^0 is close to the optimal solution, will be close to one. Detailed coverage of theory on Randomized Algorithms or Statistical Learning can be found in [72, 75].

5.4 Randomized Optimization of Hybrid Systems

5.4.1 General randomized HOCP and solution

In order to use learning theory to solve optimization problems for hybrid systems, we restate the HOCP 5.1 as a randomized one. So instead of looking for a solution $\mathcal{I}^*$ that guarantees that the cost function (5.1) achieves its absolute minimum, we seek an approximate $\mathcal{I}^0$ that evaluates (5.1) arbitrarily close to its minimum with probability almost equal to one. The Randomized HOCP can be stated as:

Problem 5.2 *Given a level $\alpha \in (0,1)$ and a confidence $\delta \in (0,1)$, the Randomized HOCP is to find a hybrid input $\mathcal{I}^0$ such that $J(\mathcal{I}^0)$ is a probable near minimum (Definition 5.1) to level α and confidence $1-\delta$ of $J(\mathcal{I})$.*

Denote by $\{\hat{\mathcal{I}}\}$ the set of the input samples $\{\hat{\mathcal{I}}_1, ...\hat{\mathcal{I}}_{N_s}\}$ that will be used to estimate the optimal hybrid input. Let

$$J^* = J(\mathcal{I}^*) = \inf_{\mathcal{I} \in \{\mathcal{I}\}} J(\mathcal{I}) \tag{5.4}$$

be the minimum cost value for the system over the complete family of hybrid inputs $\{\mathcal{I}\}$ (exact optimum), and let

$$J^0 = J(\mathcal{I}^0) = \min_{1 \le i \le N_s} J(\hat{\mathcal{I}}_i) \tag{5.5}$$

be the minimum cost value for the system over the set of input samples $\{\hat{\mathcal{I}}\}$. The general algorithms use to solve Problem 5.2 is given by Algorithms 1.

An important part of the algorithm is the generation of each input sample $\mathcal{I}_i$, which is problem dependent. Two important cases we study with numerical examples here are the generation of an open loop input sequence on a system with controlled discrete dynamics, and of a closed loop hybrid input on a system with autonomous discrete dynamics.

Algorithm 1 Randomized optimization of hybrid input

```
Require: $\alpha$, $\delta$, and $\mathbf{H}$ (Definition 2.1).
Ensure: $J^0$ and $\mathcal{I}^0$.
1: $i \Leftarrow 1$
2: Compute $N_s$ according to Theorem 5.1.
3: while $i \leq N_s$ do
4:     Generate i.i.d. hybrid input sample $\hat{\mathcal{I}}_i \in \{\mathcal{I}\}$ according to $P(\mathcal{I})$.
5:     Test input sample $\hat{\mathcal{I}}_i$ in the plant and calculate its performance $J(\hat{\mathcal{I}}_i)$.
6:     $i \Leftarrow l + 1$
7: end while
8: Choose $J^0$ from (5.5) and $\mathcal{I}^0 = \arg\min_{\hat{\mathcal{I}_1},\hat{\mathcal{I}_2},\ldots\hat{\mathcal{I}_{N_s}}} J(\hat{\mathcal{I}}_i)$.
```

5.4.2 Open loop hybrid input sequence

An important case of the controlled discrete dynamics that is studied in the literature [3, 8, 79] is when no continuous control signal is considered and the hybrid switching sequence (τ, s) can be regarded as control input.

The procedure to sample hybrid switching sequences (τ, s) may vary according to each particular problem. If the switching guards and the state space of each mode coincide for all possible discrete transitions i.e. $G_q(s) = X_q$ for all $s \in S_q$ for all $q \in Q$, and every mode can be reached from any other mode with a single transition (which is the case studied in [3, 8, 79]), and the switching labels are controllable then the hybrid switching sequence can be regarded as discrete control input. In this case, a promising procedure to obtain a hybrid switching sequence is composed of two steps: 1) Randomly select a desired number of transitions N for the sample, and then pick a random switching sequence (τ, s) of $N+1$ elements, with the times inside the optimization interval $[t_0, t_f]$, and the transition labels chosen from S. Note that one can pick any distribution for number of transitions or the switching sequence, while keeping the number of transitions bounded.

If a continuous control input is considered in the hybrid input, and since on each mode interval the continuous part of the hybrid input must be continuous, one could generate a large number of classes of input signals using a set of parameterized basis functions (e.g. polynomials, splines), such that during the sampling, their parameters and their weights are chosen to obtain the final input signal.

We provide an example to test and clarify these ideas. For comparison purposes, we use an example that already exists in the literature. In this example we optimize the performance of a dynamical system by choosing the hybrid switching sequence. There is no continuous component in the hybrid input of the existing example, which was reported in the submitted version of [4]. The system is composed of two tanks, where the objective is

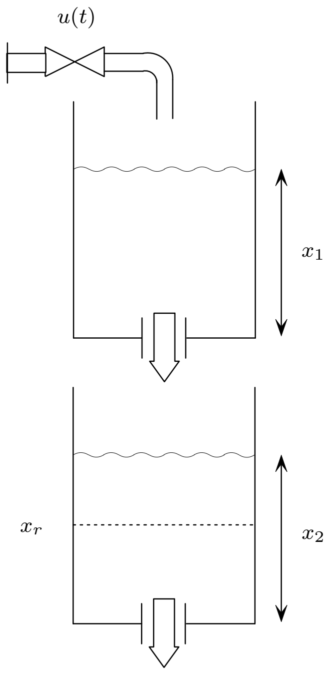

Figure 5.1: Graphical description of the tank system.

to control the fluid level of the second tank via the input flow rate to the first tank (Figure 5.1). The input variable $u(t)$ is the input flow rate to the first tank, and the state variables

$x_1(t)$ and $x_2(t)$ are the fluid levels at the first and second tanks respectively. Defining $x(t) = [x_1(t), x_2(t)]^T$ the system's equations (by Torricelli's principle) are:

$$\dot{x} = f(x,u) = \begin{bmatrix} -\gamma_1\sqrt{x_1} + u \\ \gamma_1\sqrt{x_1} - \gamma_2\sqrt{x_2} \end{bmatrix} \tag{5.6}$$

where $\gamma_1, \gamma_2 > 0$ are fixed constants. The control input is constrained to three operating states: fully open, half open, and fully closed, i.e, $u(t) \in \{u_{max}, \frac{1}{2}u_{max}, 0\}$, for some $u_{max} > 0$. Since these are discrete values, each operating state generates a discrete mode $q \in Q$ in a hybrid model (Definition 2.1). If the set of locations is $Q = \{1, 2, 3\}$, the dynamical systems $\Sigma_1,\ \Sigma_2,\ \Sigma_3$ are described as: $f_1(x,u) = f(x, u_{max})$, $f_2(x,u) = f(x, \frac{1}{2}u_{max})$, and $f_3(x,u) = f(x,0)$, $X_1 = X_2 = X_3 = \{x \in \mathbb{R}^2 : x_1 \geq 0, x_2 \geq 0\}$, and $U_1 = U_2 = U_3 = \{\}$. $S_q = Q$ for all $q \in Q$ (no restrictions on the mode transitions), $G_q^C = X_q$ for all $q \in Q$ (a discrete transition is possible for any value of x), and Z_q leaves the continuous state unchanged for all $q \in Q$.

Given an initial condition $x_0 = x(0)$ and a final time $T > 0$, the objective is to select a switching sequence that drives the error between $x_2(t)$ and a reference signal $x_r(t) \in \mathbb{R}$ to zero. To this end, the cost function is defined as

$$J = K\int_0^T (x_2(t) - x_r(t))^2 dt \tag{5.7}$$

for some $K > 0$. Comparing (5.1) to (5.7), the final cost and the transition costs are zero, and the flow cost is a continuous function (because the state flows are continuous).

The chosen parameters for the simulation (from [4, Submitted]) are $x_0 = [0.4, 0.4]^T$, $T = 5$, $K = 10$, $\gamma_1 = \gamma_2 = 1$, $u_{max} = 1$, and $x_r(t) = 0.5 + 0.25\frac{t}{T}$. The level and confidence parameters are $\alpha = \delta = 0.002$ yielding $N = 3105$ samples, generated according to our discussion above with the number of transitions distributed according to a uniform distribution $U[1, 10]$, the transition times according to $U[0, T]$ and the modes according to $U(1, 2, 3)$. The approximate optimal costs values in three different simulations were 0.1072, 0.1090 and 0.1058. The switching sequence and the state trajectories for $J^0 = 0.1058$ are shown in Figures 5.2 and 5.3. Comparing this result to that reported in the submitted version of [4], the performance obtained using the proposed approach is very close to that obtained using gradient techniques, where the reported final costs, under the same conditions, were 0.107 and 0.105 [4]. The continuous state trajectories also look similar, even though the mode schedules are different.

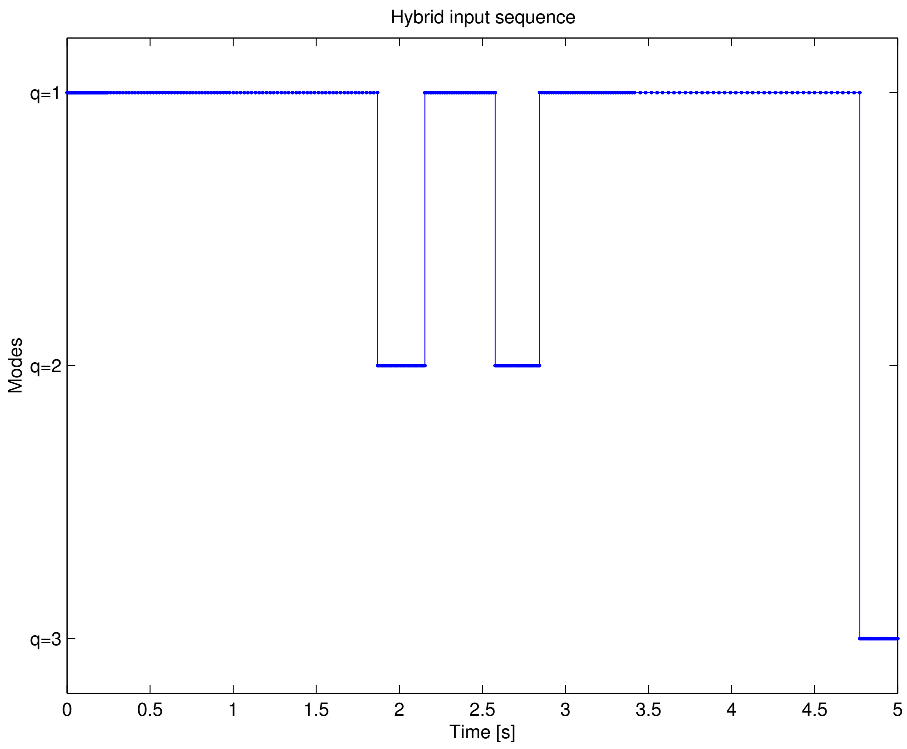

Figure 5.2: Optimal mode sequence for tanks problem using the proposed randomized approach.

5.4.3 Closed loop hybrid input

Another important case in the optimal control of hybrid systems is when the discrete dynamics are purely autonomous with forced transitions. In this case there is no direct ac-

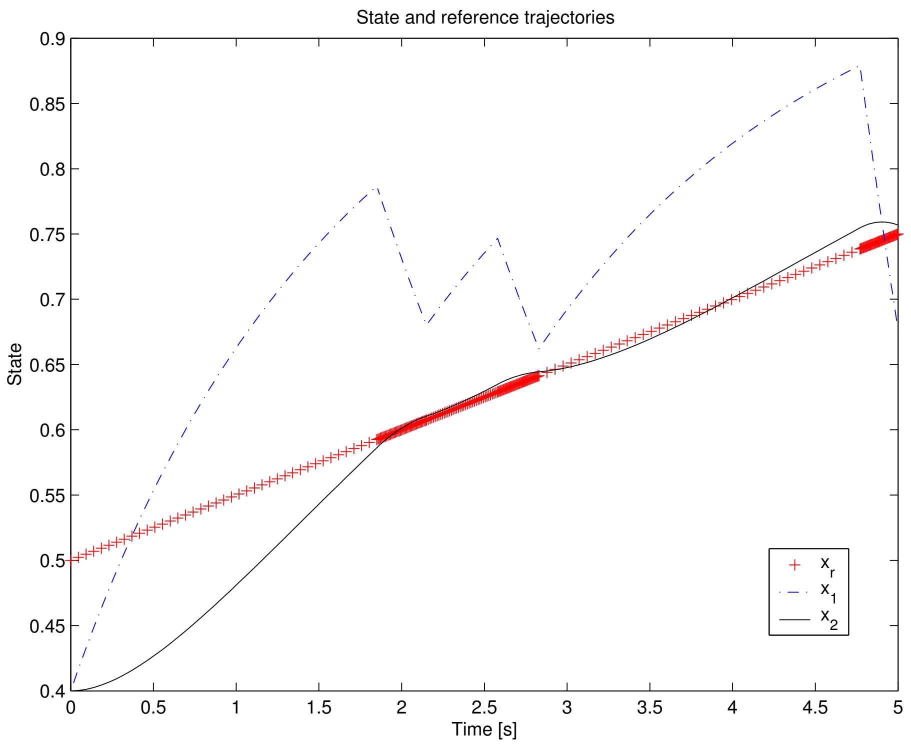

Figure 5.3: Optimal state trajectories and reference signal for tanks problem using the proposed randomized approach ($x_r(t)$: Reference, $x_1(t)$ and $x_2(t)$: State trajectories).

cess to the discrete transition labels. Instead, the transitions are governed by the transition guards i.e., whenever a transition guard is satisfied, the corresponding transition occurs immediately. This implies that the hybrid input sequence is determined by feedback functions of the continuous state of the system: the transition guards.

If one modifies the transition guards, it is possible to indirectly generate different hybrid input sequences. This may be done by considering the transition guards as parameterized functions of the state of the system. This could also be done to the continuous control function if available. Thus Theorem 5.1 and Algorithm 1 may be applied to autonomous discrete dynamics if instead of sampling hybrid inputs directly, one generates samples for the switching guards and feedback functions of continuous state of the system, generating hybrid input samples indirectly.

The example we provide to illustrate this ideas was previously presented in [10]. The problem consists of a unicycle mobile robot, that has to reach a point in the plane departing from an initial condition while avoiding a point obstacle. The robot may be controlled by switching between two discrete behaviors, “approach goal” (mode 1) and “avoid obstacle” (mode 2). The obstacle is surrounded by two circular guards (centered at the obstacle position) that determine which modes are active. If the robot is in mode 1 and reaches the inner guard, it switches to mode 2. If it is in mode 2 and reaches the outer guard, it switches to mode 1. The optimization problem is to choose the best radii for these two circular guards such that the robot gets as close as possible to the goal without hitting the obstacle.

The robot is described using its kinematic model as:

$$\dot{x} = v \cos\phi \tag{5.8a}$$

$$\dot{y} = v \sin\phi \tag{5.8b}$$

$$\dot{\phi} = \omega \tag{5.8c}$$

where the position of the robot (x, y) and its orientation ϕ form the continuous state of the system, and v and ω are its linear and angular speeds. The goal is located at $(x_g, y_g) \in \mathbb{R}^2$, and the obstacle at $(x_o, y_o) \in \mathbb{R}^2$. v has a constant value while ω is the feedback control input that changes according to each mode: In mode 1, ω is given by $\omega_1 = C_1(\phi_g - \phi)$ where $\phi_g = \arctan((y_g - y)/(x_g - x))$ and $C_1 > 0$ is a constant. In mode 2, ω is given by $\omega_2 = C_2(\phi - \phi_o)$ where $\phi_o = \arctan((y_o - y)/(x_o - x))$ and $C_2 > 0$ is a constant.

The procedure to map the robot model (5.8) to the hybrid system (Definition 2.1) is similar to that on the previous example. However the parametrization of the switching guards and reset maps is different (similarly to [10] we assume a fixed continuous control input (C_1, C_2) that is not considered as an optimization variable). Note from the nature of the system that the reset maps Z_q leave the continuous state unchanged for all $q \in Q$, while the transition guards must be parameterized according to the circular guards described in the problem. Therefore, let

$$G_1^A(v_1 = 2, r_{in}) = (x_o - x)^2 + (y_o - y)^2 - r_{in}^2 \tag{5.9a}$$

$$G_2^A(v_2 = 1, r_{out}) = (x_o - x)^2 + (y_o - y)^2 - r_{out}^2 \tag{5.9b}$$

where the G_1^A is the guard to jump from mode 1 to mode 2, G_2^A is to jump from mode 2 to mode 1, and r_{in} and r_{out} are the radii of the inner and outer guards respectively. The cost functional is defined as

$$J = \int_0^T \left[(x_g - x)^2 + (y_g - y)^2 + \beta \mathrm{e}^{-\xi[(x_o - x)^2 + (y_o - y)^2]} \right] dt.$$

The simulations parameters are $v = 1$, $C_1 = 2$, $C_2 = 0.4$, $T = 3$, $\xi = 10$, $\beta = 5$, $(x_g, y_g) = (2.25, 2)$, $(x_o, y_o) = (1, 1)$, and $(x(0), y(0), \phi(0)) = (0, 0, 0)$. The level and confidence parameters are $\alpha = 0.02$ and $\delta = 0.02$ yielding $N = 194$ (note that because the sampling space is simple, low values of level and confidence parameters can yield to good results). The resulting approximate optimal cost was $J^0 = 10.4695$ with optimal $r_{in}^0 = 0.4807$, $r_{out}^0 = 0.5006$, while an optimization performed using the algorithm reported in [10] with the same values yielded $J^* = 10.4609$, $r_{in}^* = 0.4963$, $r_{out}^* = 0.4963$. A comparison of the trajectories and guards obtained via the randomized approach (Figure 5.4), and those obtained using the gradient descent approach [10] (Figure 5.5) shows that both results are almost identical. (Figure 5.4 shows the approximate optimal result while Figure 5.5 shows the evolution of the optimization algorithm of the guards and trajectories converging to the optimum that is where the two guards collide).

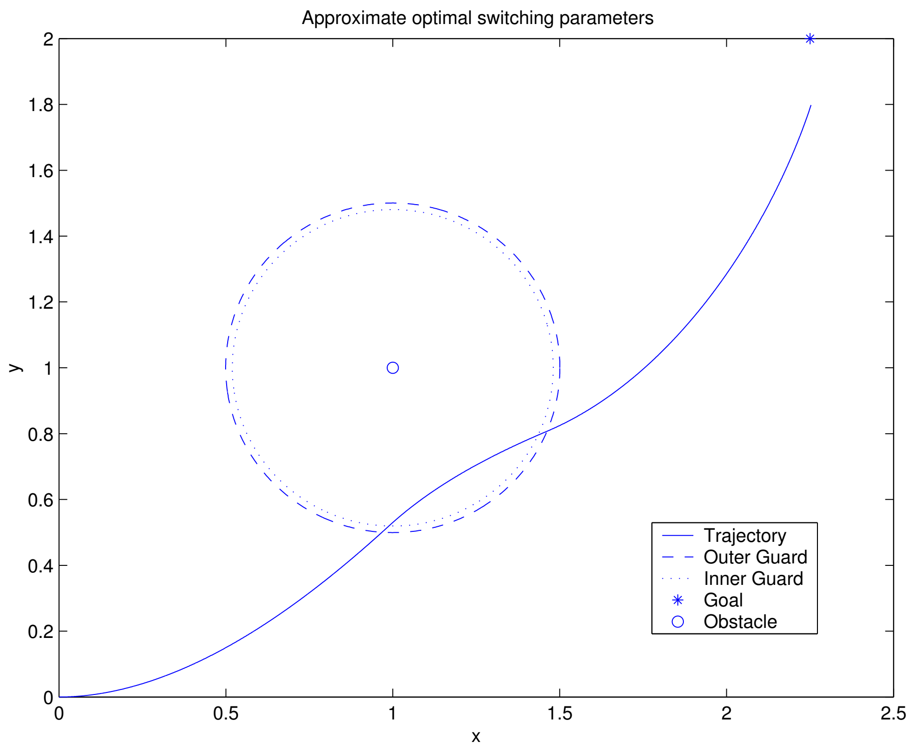

Figure 5.4: Robot trajectory and optimal guards using the proposed randomized approach

5.5 Randomized optimization of the agents' discrete dynamics

This section discusses the application of the ideas in this chapter to the resource allocation problem discussed along this dissertation. It is important to note that the general discussion in Sections 5.2 and 5.4 also applies to the Interconnected Hybrid Systems defined in Chapter 4 with minor notation changes. Therefore it is straightforward to extend the results in this chapter to multi-agent systems with hybrid interacting dynamics.

The problem we consider here is an special case of the original problem discussed in Chapter 3. We consider agents moving among discrete locations in a network with fixed

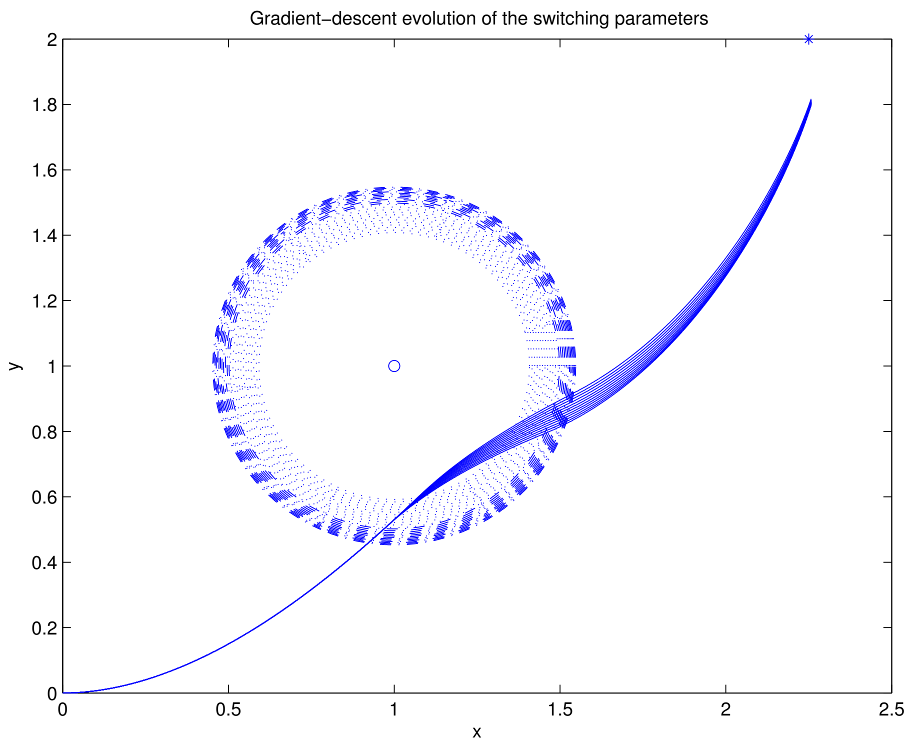

Figure 5.5: Evolution of the robot trajectory and switching guards using a gradient descent approach. The optimum configuration is where the two guards collide.

topology and nodes with fixed amounts of resources. We restrict ourselves to these cases because the discrete dynamics of the agents are designed based on the general structure proposed in Chapter 4, which is only applicable to systems with autonomous discrete dynamics.

5.5.1 Simulation set-up

In this numerical example we consider a network of 5 nodes that are arbitrarily connected as shown in Figure 5.6, and host 12 agents total in the whole network. Both agents and nodes are identified by natural numbers. The network only has one type of resource for simplicity. The available resource in each node is $(r_1, \ldots, r_5) = (2, 4, 3, 4, 5)$. The network topology and resources in each node are fixed as discussed above.

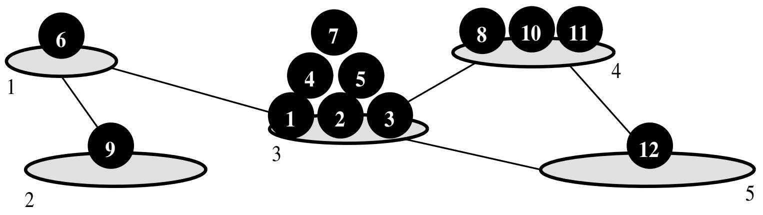

Figure 5.6: Network configuration for numerical example. Nodes and agents are identified by natural numbers. The agents initial locations in the networks are shown.

The benefit function of the agents has the same form to that in the numerical example of Chapter 3 (Section 3.6):

$$W_k(x_k) = \nu_k \ln(x_k)$$

The continuous dynamics of both agents and nodes are also kept identical to Section 3.6:

$$f_{q_i} = \left[L_{q_i}(p_{q_i})\Big(\sum_{\{k \in \mathcal{A} : q_k = i\}} x_{q_k} - r_{q_i}\Big)\right]^+_{p_{q_i}}, \ \forall i \in \mathcal{V}, \tag{5.10a}$$

$$f_{q_k} = K_{q_k}(x_{q_k})\Big(\frac{\nu_k}{x_{q_k}} - p_{q_{i=q_k}}\Big), \ \forall k \in \mathcal{A} \tag{5.10b}$$

where $\mathcal{V} = \{1, \ldots, 5\}$, $L_{q_i}(p_{q_i}) = \tanh(p_{q_i}) + 1$, $\forall q_i \in Q_i \ \forall i \in \mathcal{V}$, and $K_{q_k}(x_{q_k}) = 50x_{q_k} \ \forall q_k \in Q_k \ \forall k \in \mathcal{A} = \{1, \ldots, 12\}$. The weights of the benefit function in this case are given by (subindexes next to the number are to ease identification):

$$(\nu_1, \ldots, \nu_{12}) = (0.5_1, 0.6_2, 0.1_3, 0.3_4, 0.4_5, 0.9_6, 0.4_7, 0.3_8, 0.2_9, 0.1_{10}, 0.5_{11}, 0.8_{12}) \tag{5.11}$$

The nodes and agents' discrete dynamics are designed as discussed in Section 4.5. The nodes update their discrete state depending on the agents located in it. Therefore, as soon as agent k migrates from node i to node i', a transition in node i occurs removing agent k

from the sum in (5.10a) for i, and a transition in node i' occurs adding agent k to the sum in (5.10a) for i'.

The discrete transitions in the agents dynamics represent the location change of the agent in the network. They are governed by transition guards that allow them to compare the benefit they get at their current location to the potential benefit they could get at a new location. As discussed in Section 4.5 a transition guard $G_{q_k=i}(s = i')$ (transition guard for agent k to migrate from node i to node i') is satisfied if the following conditions are satisfied (Eq. (4.2)):

$$\mathrm{mig}_k(p_{i'}, i') < \mathrm{nmig}_k(p_i, i) \;\text{ and} \tag{5.12a}$$

$$\mathrm{mig}_k(p_{i'}, i') < \mathrm{mig}_k(p_j, j), \;\; \forall j \neq i'; j \in V(q_k) \tag{5.12b}$$

where for this particular example we make

$$\mathrm{nmig}_k(p_i, i) = p_i \tag{5.13a}$$

$$\mathrm{mig}_k(p_{i'}, i') = \zeta \frac{p_{i'}}{\nu_k} + \eta \tag{5.13b}$$

where ζ and η are real parameters and $\zeta > 0$. Functions $\mathrm{mig}(\cdot)$ and $\mathrm{nmig}(\cdot)$ are chosen in this form because they are monotonically increasing as required in Section 4.5 and they are the simplest special case of a polynomial function of p, which makes the decision vector small but flexible enough to control the agents in the network. Note that ζ, η, are inversely related to the *willingness* of all the agents in the network to execute a transition, while ν_k (the weight of the agent's benefit function) is directly related to the willingness of agent k to execute a transition. The higher the values of ζ and η the lower is the likelihood that the agents in the network perform a transition, while the higher the value of ν_k the higher is the likelihood that agent k performs the transition. This implies that ζ and η may be used to control the number of transitions that the agents in the network may execute, but that the willingness to transition of each agent k also depends on the importance of the resource (determined by ν_k) for that agent.

In this numerical example we randomly optimize the values of ζ and η using Algorithm 1. The simulation time for the system is 12 seconds. The agents are only allowed to execute transitions every 2 seconds in a synchronized form, enforcing a dwell time such that the system is not destabilized due to agents switching between two nodes too fast. The discrete dynamic parameters ζ and η are allowed to have different values for each transition time at the end of a two-second interval, so the decision vectors become $\vec{\zeta} = (\zeta_1, \ldots, \zeta_5)$ and $\vec{\eta} = (\eta_1, \ldots, \eta_5)$. Each ζ_c (with $c = 2, 4, \ldots, 12$) is assumed to be uniformly distributed between 0 and 4, i.e. $U[0, 4]$, and each η_c is uniformly distributed between 0 and 1 i.e. $U[0, 1]$. Each component in $\vec{\zeta}$ and $\vec{\eta}$ is generated independently of the other components of the vector.

The total benefit function to maximize in this simulation is given by

$$J(\vec{\zeta}, \vec{\eta}) = \sum_{c=2,4,\ldots,12} \left[\sum_{k\in\mathcal{A}} W_k\big(x_k(t)\big)|_{t=c} \right] \tag{5.14}$$

This benefit function resembles the objective that we stated in Problem 3.1, which required that the agents converged to a point that coincides with the maximum of $\sum_{k\in\mathcal{A}} W_k(x_k)$. In this case we include the sum of the network's total benefit every 2 seconds because we would like that the agents improve the overall benefit on the network every time they execute a transition (every 2 seconds), and that at the end they converge to a point that is close to the optimum configuration.

Algorithm 1 performs the following steps for this particular example:

1. Compute N_s based on α and δ, according to Theorem 5.1.

2. Generate an i.i.d. sample of $\vec{\zeta}$ and $\vec{\eta}$ to be simulated.

3. Simulate system to evaluate performance. Note that every 2 seconds every agent evaluates the functions nmig and mig for its current location, and all other nodes in its neighborhood, and decides to either stay at its current location or migrate to

the best node it found depending on whether a transition guard was satisfied, which depends on the values of the corresponding component of $\vec{\zeta}$ and $\vec{\eta}$. After a transition occurs (or not), the continuous dynamics (5.10) continue its evolution as described in the simulations of Section 3.6 until a new decision time (2 seconds after) is reached.

4. Go back to step 2 unless tested samples match N_s.

5.5.2 Overall behavior of the algorithm

The first simulation is aimed to study the overall performance of the optimization algorithm with respect to the variations in level (α) and confidence (δ) parameters. We are interested in observing what is the influence of these parameters in the capability of the randomized algorithm for obtaining a good performance in our particular system.

This simulation is configured to test the system described above with seven different combinations of values for α and δ that are summarized in Table 5.1. These values cause the number of samples to change according to Theorem 5.1.

α	0.01	0.01	0.005	0.005	0.005	0.002	0.002
δ	0.01	0.005	0.01	0.005	0.002	0.005	0.002
N_s	459	528	919	1058	1240	2647	3105

Table 5.1: Values for α and δ used in the first numerical test.

In order to see the consistency of the performance of Algorithm 1 applied to this system, we execute the optimization algorithm 10 times for each $\alpha - \delta$ combination. We then compute the mean, standard deviation, and the best performance obtained among each group of 10 simulations with common $\alpha - \delta$ values. These result are summarized in Figure 5.7.

As expected, as the number of samples grow ($\alpha - \delta$ reduce) the behavior of the algo-

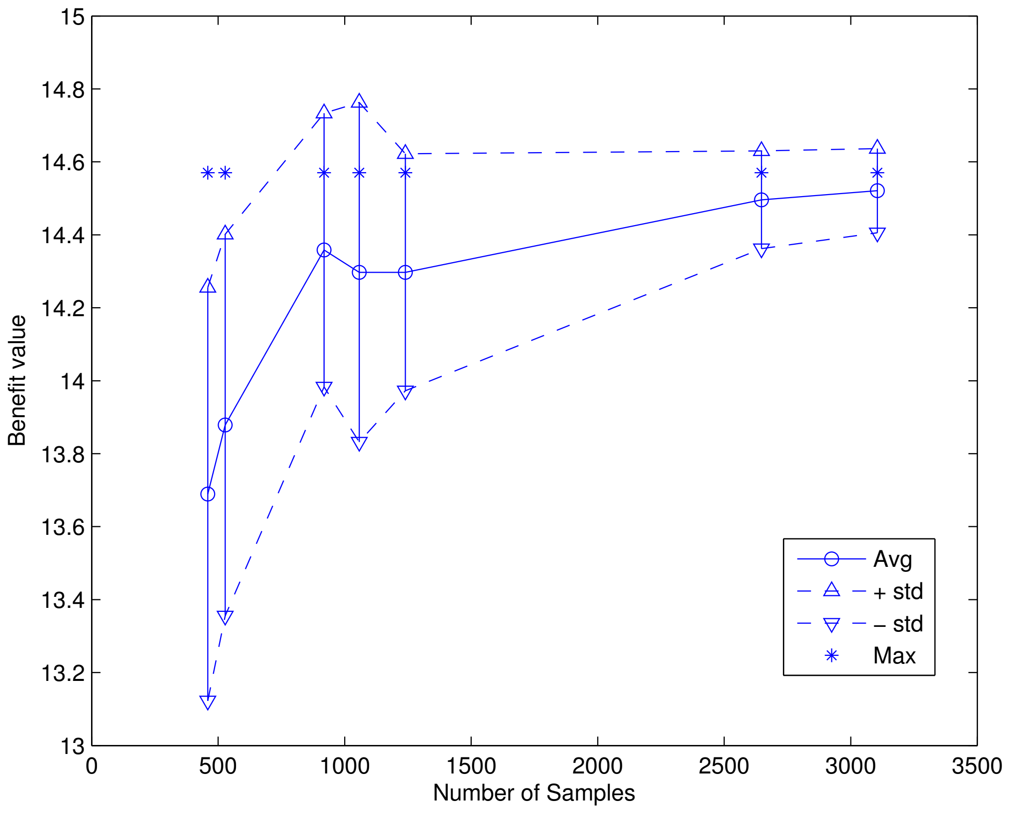

Figure 5.7: Overall performance of randomized optimization in the multi-agent system. The system is optimized ten times for each $\alpha - \delta$ combination in Table 5.1. Then the average, standard deviation, and maximum performance are computed for each group of ten results. The average is represented by $-o-$, the average plus one standard deviation by $-\triangle-$, the average plus one standard deviation by $-\nabla-$, and the maximum among the ten tests by $- * -$. Note that taking the mean and standard deviation among ten instantiations of the algorithm may not be enough to provide a precise statistical characterization of the behavior of the system but we believe it is helpful to illustrate the general behavior of the randomized algorithm. Also note that the true optimum is unknown because it is difficult (or impossible) to compute with the currently available model-based optimization algorithms for hybrid systems.

rithm becomes more consistent. The average value gets closer to the maximum obtained, and the standard deviation is reduced. Note however that even with the lowest number of samples in the plot (459) the maximum the algorithm ever achieved was obtained. Moreover the algorithm was capable of obtaining that maximum (close to 14.6) with any of the $\alpha - \delta$ combinations tested. However the likelihood of obtaining that maximum is lower with low number of samples.

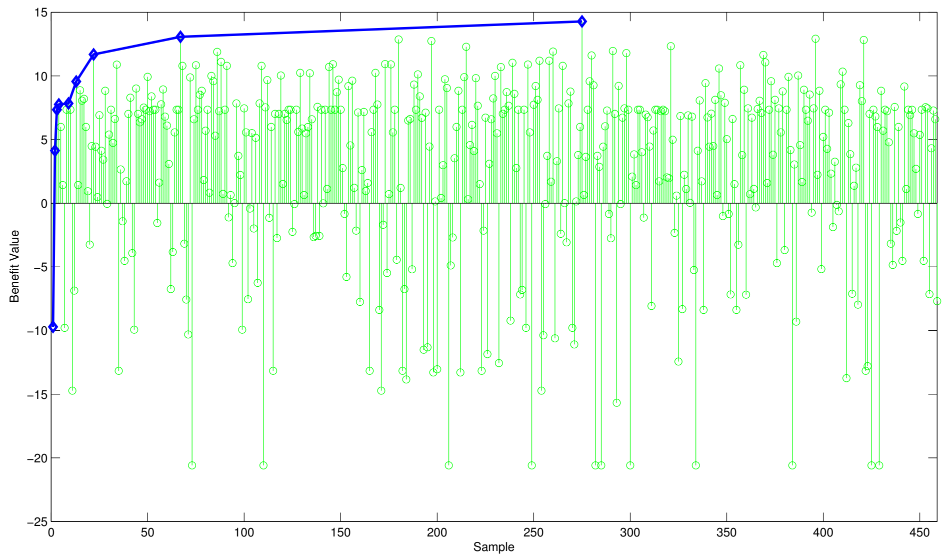

Figure 5.8: Optimal benefit evolution during one optimization run. The optimization algorithm is programmed to test the samples iteratively. Each point in the plot is the benefit value obtain with the corresponding sample. The solid curve highlights the samples where the benefit value improved compared to the previous one.

During this test, we also recorded the evolution of Algorithm 1 during one optimization run, which is shown in Figure 5.8. This only makes sense because the algorithm is programmed to test the random samples iteratively. As expected the samples are independent from each other, and the improvements occur in a random pattern, even though they become more spaced as the benefit value gets closer to the optimum.

5.5.3 Agents' optimal behavior

In this part we explore the optimal behavior of the system as obtained by the randomized algorithm, and also, the behavior of the system during one optimization run. We are interested in the benefit of optimal sample, the behavior of the total benefit of the network on time[1], the trajectories of the agents in the network, the continuous trajectories of agents and nodes, and the final distribution the agents achieve. We perform one optimization with $\alpha = \delta = 0.02$, which yield 459 samples. The probable near maximum benefit obtained for this case is $J^0 = 13.07$. The optimal control vectors are

$$\vec{\zeta}^0 = (1.33, 0.34, 1.58, 1.88, 3.92), \text{ and}$$

$$\vec{\eta}^0 = (0.24, 0.57, 0.50, 0.41, 0.92).$$

The optimal agent discrete trajectories are shown in Figures 5.9, and 5.10. These two figures are related in that the discrete states of the agents and the discrete locations of the agents are labelled identically, e.g if the discrete of agent k is 3 then it is located at node 3. The optimal trajectories of the agents continuous states (captured resources) are shown in Figure 5.11, while the nodes continuous trajectories (prices) are shown in Figure 5.12. The continuous dynamics will not be discussed in detail here because similar cases were discussed in the simulations of Section 3.6. These results are provided here for completeness.

The agents start at the initial discrete location shown in Figure 5.10-top (this can also be seen in Figure 5.9) and stay in that position until 2 seconds, providing enough time for the continuous dynamics to converge to the equilibrium for this particular distribution of agents (see Figures 5.11 and 5.12). Note that there is a large number of agents concentrated at node three, which causes a suboptimal allocation of resources. At 2 seconds (see Figures

[1]The benefit of the (optimal) sample is that given by (optimal solution of) (5.14), while the total benefit of the network at each particular time is the sum of the benefit of all the agents in the network at that particular time.

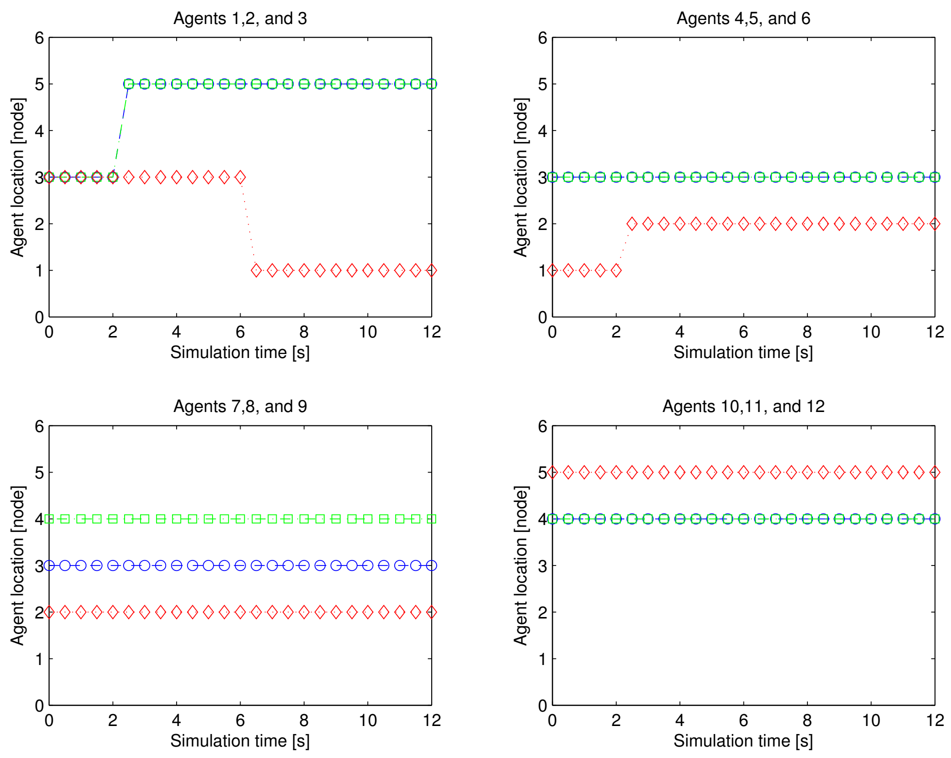

Figure 5.9: Optimal agent discrete state trajectories. The top left plot shows agents 1, 2, and 3. The top right shows agents 4, 5, and 6. The bottom left shows agents 7, 8, and 9. The bottom right shows agents 10, 11, and 12. Within each figure, the first agent is identified by a blue $-o-$, the second by a green $-\square-$, and the third by a red $-\diamondsuit-$.

5.10-middle and 5.9), the agents execute some transitions to improve the total benefit of the network. Agents 1 and 2 migrate to node 5, which is the node with greater quantity of resources in the network and that is only occupied by one agent. Note that agents 1 and 2 have the highest weights (0.5_1 and 0.6_2 on equation (5.11)) compared to the other agents in node 3 (0.1_3, 0.3_4, 0.4_5, and 0.4_7). At the same time agent 6 (with weight equal to 0.9) migrates from node 1 to node 2, seeking more resources as well, since node 2 has twice as much resources as node 1 and is only occupied by agent 9 who has a very low weight (0.2), which implies that agent 6 will get more resources at node 2 (sharing them

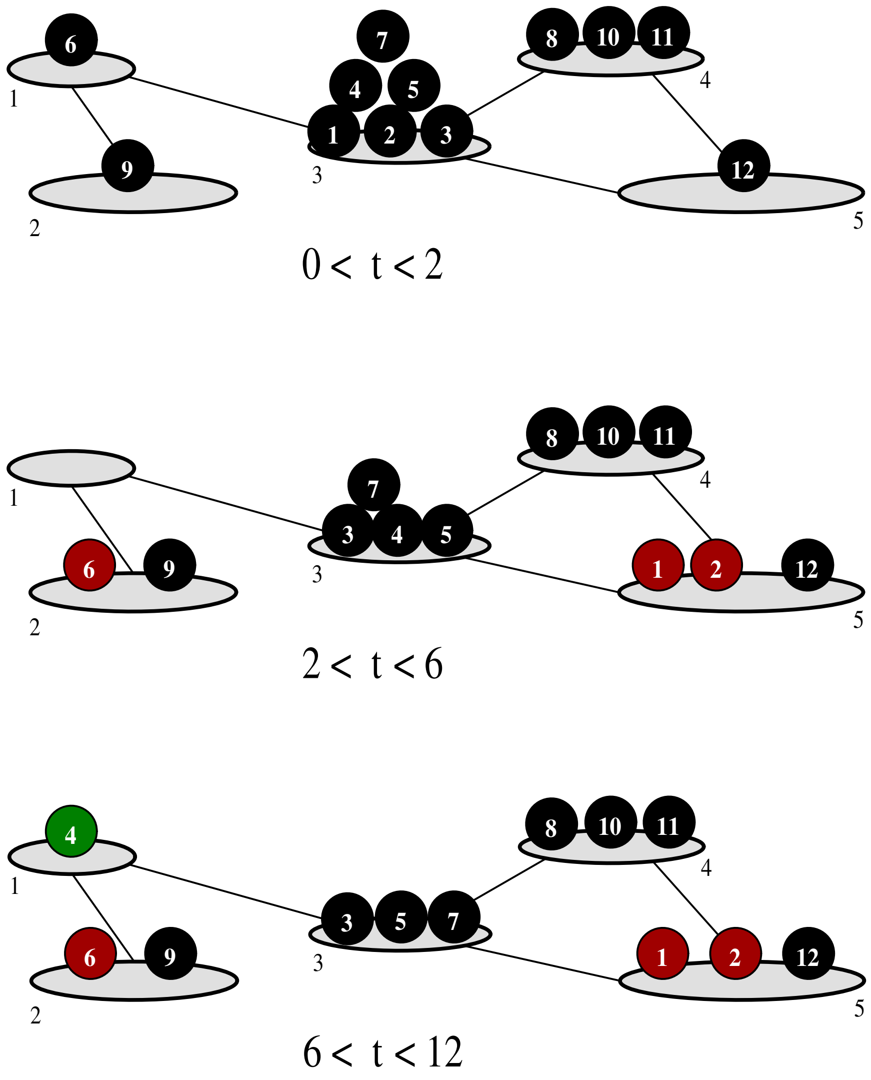

Figure 5.10: Optimal agents' trajectory in the network. The agents (black or colored circles) and nodes (gray ovals) are identified by natural numbers. The agents start at the initial condition shown at the top and remain in that discrete position until 2 seconds. At 2 seconds, agents 1, 2, (highlighted in red) migrate from node 3 to node 5, while agent 6 migrates from node 1 to node 2 (middle picture). The agents then remain in that position until 6 seconds when agent 4 (highlighted in green) migrates from node 3 to node 1, which is the final position of the agents in the network (bottom picture).

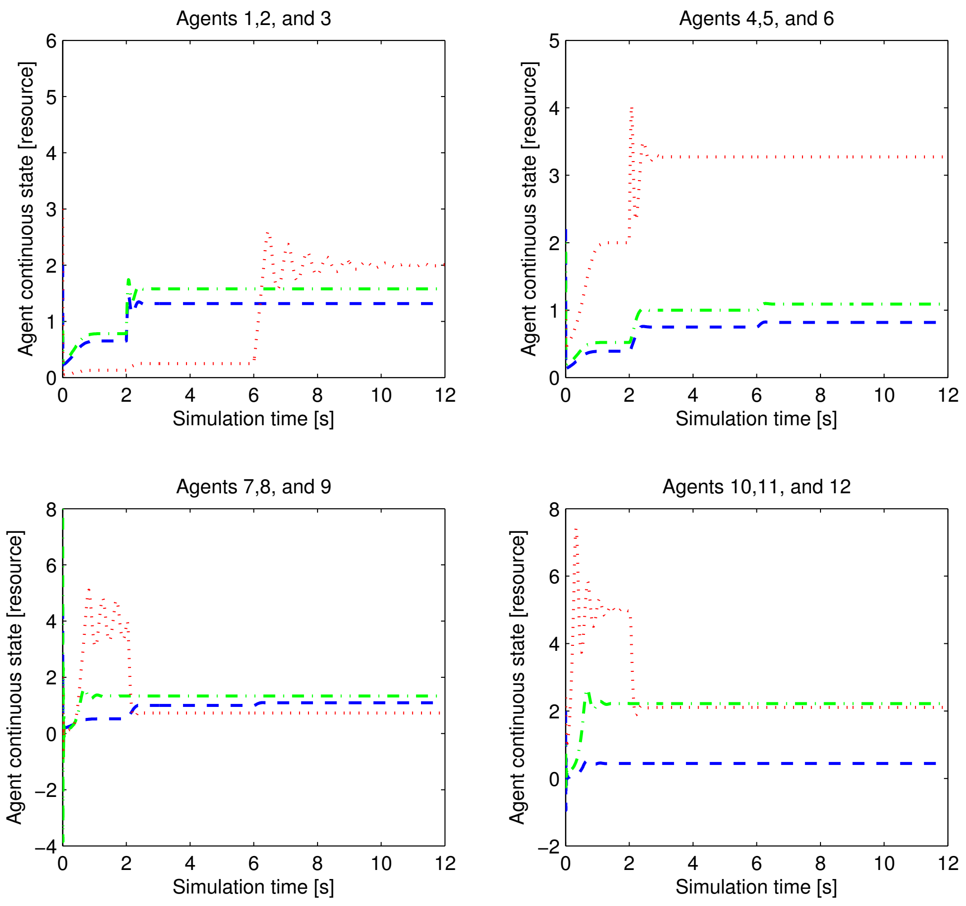

Figure 5.11: Optimal continuous state trajectories of the agents. The top left plot shows agents 1, 2, and 3. The top right shows agents 4, 5, and 6. The bottom left shows agents 7, 8, and 9. The bottom right shows agents 10, 11, and 12. Within each figure, the first agent is identified by a blue (— —) line, the second by a green (— ·) line, and the third by a red (· · ·) line.

with agent 9) than it gets at node 1 (all resources for itself).

Then the network stays at the configuration shown in Figure 5.10-middle for 4 seconds, allowing the continuous dynamics to converge to a new equilibrium for the current network configuration (see Figures 5.11 and 5.12). At 6 seconds agent 4 migrates from node 3 to

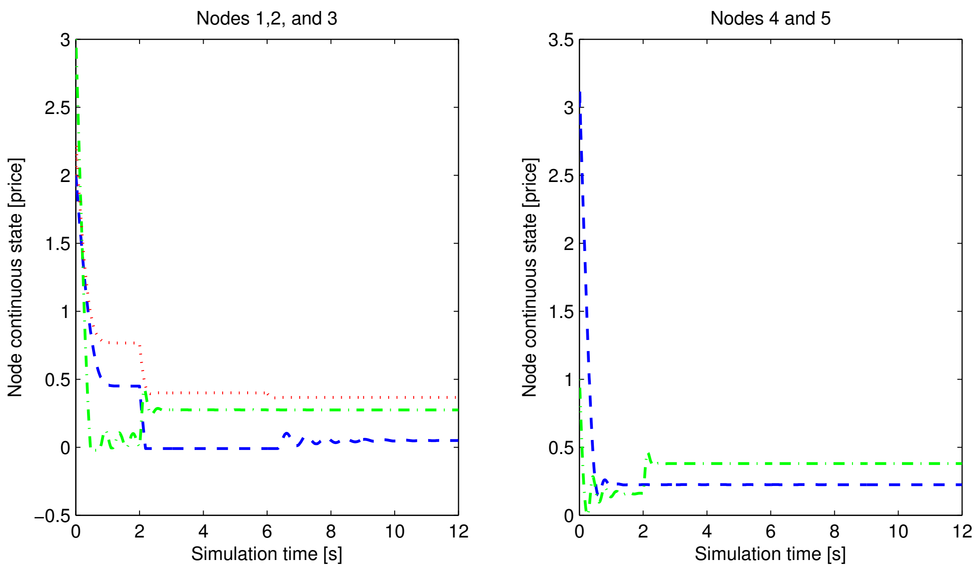

Figure 5.12: Optimal continuous state trajectories of the nodes. The left plot shows nodes 1, 2, and 3. The right shows nodes 4, and 5. Within each figure, the first node is identified by a blue $(--)$ line, the second by a green $-\cdot$ line, and the third by a red $(\cdots)$ line (where applicable).

node 1 following a similar logic to that in the previous paragraph. The network stays at that configuration (Figure 5.10-bottom) for the remainder of the simulation.

The end configuration seems to have a better allocation of resources than the initial one. Node 5 which is the one with greatest amount of resources is occupied by agents with high resource weights $(0.5_1,\ 0.6_2,\ 0.8_{12})$. Node 2 that also has high amount of resources is occupied by two agents only, but one of the (agent 6) is the agent with highest weight in the network (0.9). Node 1, which has the fewest amount of resources is only occupied by one agent (agent 4) with an average weight (0.4), while nodes 3 and 4 share the rest of the agents that all have weights lower than 0.5.

Figure 5.13 supports the previous discussion showing the quantitative improvement of the total benefit of the network on time. Each curve in Figure 5.13 represents the evolution

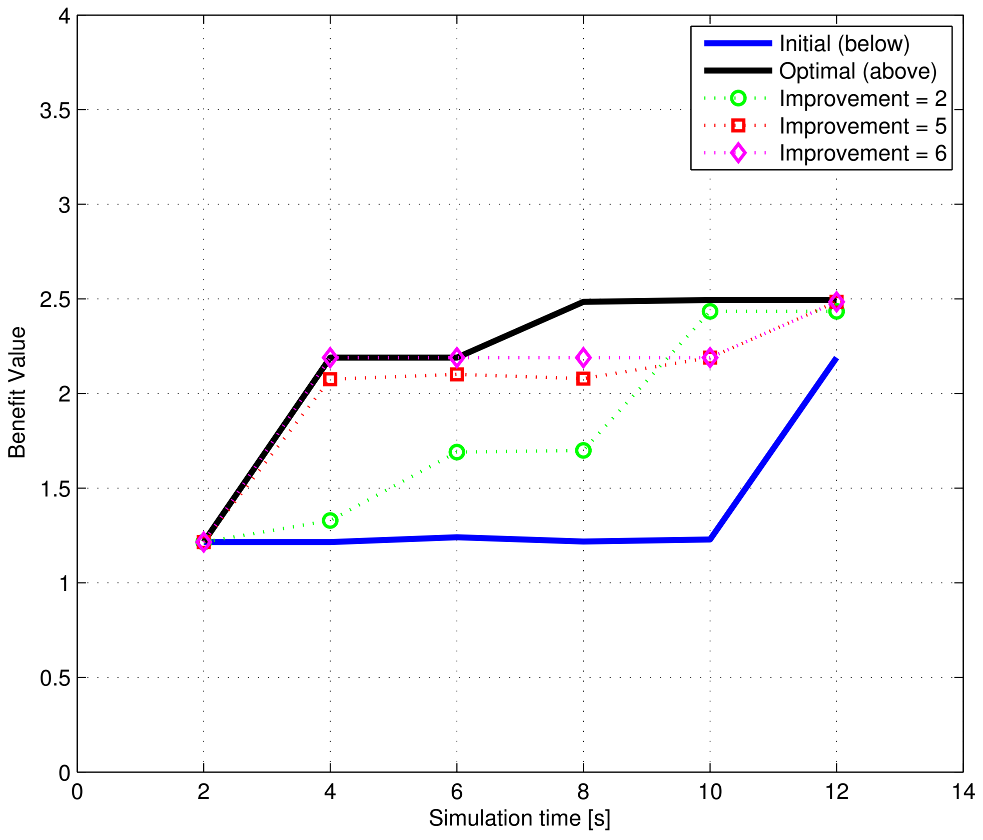

Figure 5.13: Time evolution of the total benefit of the network for different suboptimal samples. Each curve represents the total benefit of the network on time during one numerical simulation of the system using a different set of suboptimal control parameters ($\vec{\zeta}$ and $\vec{\eta}$. The solid blue curve represents the first sample tested. The dashed (green, red, and cyan) curves represent suboptimal samples where some improvement was achieved. The solid black curve represents the optimal trajectory in this optimization.

on time of the total benefit of the network for different samples during one optimization run. The solid blue curve corresponds to the first tested sample. The dashed (green, red, and cyan) curves correspond to suboptimal samples where some improvement on the total benefit function of the optimization problem was achieved. The solid black curve corresponds to the optimal solution of this optimization.

Note that the total benefit of the network for the optimal trajectory (black curve in Figure 5.13) improves every two seconds, or at least remains unchanged. So the network starts with a total benefit at 2 second of 1.2. This improves at 4 second to 2.2, which stays the same at 6 seconds, and then improves to 2.5 at 6 seconds where it is kept until the end of the simulation. This approximately yields (summing the benefits of all the two-second intervals as done in (5.14)) the optimal optimization benefit 13.1, which is close to the exactly computed $J^0 = 13.07$.

Compare this optimal trajectory to any other one int Figure 5.13. Take for example the initial sample (solid blue). The total benefits of the network for every two-second interval are $1.2,\ 1.2,\ 1.2,\ 1.2,\ 1.2,\ 2.2$, which yields a benefit value for this sample of about 8.2. Alternatively take the curve of improvement 5 (red squares), where the total benefits of the network every two-second interval are $1.2,\ 2.1,\ 2.1,\ 2.1,\ 2.2,\ 2.5$ which yields a benefit value for this sample of 12.2. Note that both of this curves present improvements and degradations in the total benefit of the network among two-second intervals. These degradations are not observed in the optimal sample.

This allows us to conclude that the benefit functional (5.14), the decision policies that govern the agents discrete transition, and the randomized optimization algorithm make the system improve the total benefit of the network every time the agents are allowed to change locations, and also make the agents converge to a discrete configuration that comes close to the the optimal configuration.

5.6 Conclusions

We have presented a randomized approach for optimal control of hybrid systems. We defined a general hybrid optimal control problem, and stated an equivalent problem in the randomized framework. We provided an expression that relates the requirements of the desired solution (level α and confidence $1-\delta$) with the computational complexity (number samples) needed to guarantee such requirements. In this form the performance/ computational complexity trade-off can be controlled. We tested our approach on two different examples and obtained comparable results to those available in the literature using model-based approaches. The advantage of the proposed approach is that the theoretical analysis is simplified due to its model-free nature, making it attractive for complicated systems where model-based techniques may face difficulties. However, the price to pay for this problem simplification is that the proposed approach does not guarantee the optimality of the final solution.

We applied the randomized optimization approach to the design of the discrete dynamics on the resource allocation problem discussed along this dissertation. The problem considered in this chapter is an special case of the general problem defined in Chapter 3. We consider a network with fixed topology and fixed amount of resources in the network because the discrete dynamics are based on the discussion in Chapter 4, which is only applicable to this particular case. The extension of the optimization algorithm to the IHS case is straightforward. The performance we obtained from the numerical examples makes the system approach an optimal solution as expected. Moreover, the optimal behavior of the agents yields an improvement of the total resource of the network every time the agents make a transition between nodes.

The results obtained so far are useful for special subclasses of the general hybrid system defined in this chapter. Possible extensions include the application of the proposed algorithm to more general types of dynamics, and the inclusion of uncertainty in the system. These extensions would be particularly interesting in the resource allocation problem, because they would allow us to consider a time varying network topology and changes in the nodes' resources. In particular the control of uncertain systems could be achieved using statistical learning in similar form as previously discussed for continuous systems in [34, 72, 76].

Chapter 6

Discrete Asymptotic Abstractions of Hybrid Systems

6.1 Introduction

Motivated by the complexity of the multiagent system studied here, we explore abstraction procedures for model simplification of hybrid systems. In particular we introduce the notion of Finite Time Mode Abstraction to relate a hybrid system to a timed automaton that preserves the stability and reachability properties of the former. The abstraction procedure discards the continuous dynamics of each mode in the hybrid automaton completely, keeping only the information about the maximum time in which the continuous state makes a discrete jump. This information is used to construct a timed automaton, based on the original hybrid system, and to prove that the stability and reachability properties of the original system are retained in the abstract timed automaton. In the process of abstracting a hybrid to a timed automaton we introduce a new notion of hybrid distance metric, which provides information about both the number of discrete transitions that a system would have to make to go from one hybrid state to another, and the distance between the continuous parts of such hybrid states.

The interactions between continuous and discrete dynamics in hybrid systems make the analysis, controller synthesis, and performance optimization tasks difficult. This complexity is further increased if there is multiple interacting hybrid systems, because the dynamics of the component agents may be influenced at both the continuous and discrete levels by the continuous and/or discrete dynamics of other agents. This modeling complexity motivates the need for alternative techniques to simplify control related tasks. In particular, in the multi-agent problem we are interested in, stable continuous dynamics been designed for all agents in Chapter 3. This makes it possible to think that the continuous evolution history may be irrelevant for the system's overall performance, while the information about the asymptotic behavior of the system i.e. its equilibrium points, and the time it takes to reach those may be crucial for such performance evaluation. Thus the objective is to discard as much irrelevant information as possible from the system's description without loosing the key components for the task of interest.

The procedure that allows such model simplification is called abstraction. Loosely speaking, system's abstraction is the selective retention of information pertinent to a specific task or objective. Two well known abstraction procedures are based on *bisimulation* and *simulation* relations [40]. A bisimulation relation generates a simplified system, whose state space is a partition of the original one, but with a model that is input-output equivalent with respect to a given property (e.g. reachability). Therefore checking a property on a bisimilar quotient system is equivalent to checking it on the original model. A simulation relation also generates a simplified system, however in this case the simplified system may have richer behavior that the original. Therefore checking a desired property on a similar quotient system is sufficient to claim that such property holds on the original system, without the converse being necessarily true.

Abstractions for continuous dynamical systems using bisimulation ideas have been explored for linear systems in [47,48] and for nonlinear systems in [49,50,67,74]. However, the link between this form of abstraction and bisimulation relations is only recognized in the later results [47, 67, 74]. In [66, 68] the authors present abstraction of nonlinear control systems based on category theory with the ultimate objective of enabling correct by design embedded systems' synthesis.

Bisimulation was proven to be a restrictive concept in the survey paper [2], where it is demonstrated that in order to obtain bisimulations for hybrid systems, one has to restrict either the discrete logic that governs the transitions, or the type of continuous dynamics. Less restrictive idea of simulation-based abstraction was explored for linear system in [71], using a similar framework to that of [47]. Other interesting, and less restrictive concepts for abstraction are the approximate bisimulations [23], and approximate simulations [22, 65]. Approximate (bi)simulations are (bi)simulation relations that instead of having exact observation correspondence, these are allowed to be with in a certain distance controlled by a predefined precision parameter.

Applications of abstraction to multi-agent systems were explored in [5, 33]. The authors in [5] use abstractions to map the combined state space of a group of robots to a state space that only contains information about group position and shape. Then in [33] these ideas are exploited using hierarchical abstractions for model simplification of robotic swarms and temporal logics for motion planning and control (at the level of natural language) of the swarm.

The abstraction method we propose here is still less restrictive than approximate simulation relations. We are interested in studying the asymptotic behavior of the system, and possibly how much time the system takes to reach a neighborhood of an asymptotic equilibrium set, being willing to sacrifice knowing how the system got there. We therefore propose a method to discard the continuous dynamics of the hybrid system almost completely, substituting this information by clocks that will tell us how much time the system expends to get to the equilibrium set. This results in mapping the hybrid system to a timed automaton. The assumption we make for this purpose is that the system contains a finite number of disjoint limit sets, and that these are contained in the transition guard of each discrete mode. In our way to formulate this abstraction procedure, we also propose a new hybrid metric that as opposed to the metric use in [37], provides enough information in a single number to determine how many jumps the system need to reach one discrete mode from another, and how far are two continuous states from each other. Finally we discussed the applicability of these concepts to the resource allocation problem at hand.

6.2 Preliminary Concepts and Assumptions

In this chapter we restrict our attention to autonomous hybrid systems. Therefore we do not consider any type of input in either of the continuous or discrete dynamics in Definition 2.1:

Assumption 6.1 *The following items are modified in Definition 2.1:*

- *The continuous dynamical systems are autonomous. Therefore $\Sigma_q = (X_q, f_q, \mathbb{R}^+)$ for all $q \in Q$, where f_q is a smooth map and $X_q \in \mathbb{R}^m$.*
- *The discrete transitions are only of autonomous type. Therefore S_q only contain autonomous transition labels, $G_q = G_q^A$ for all $q \in Q$, and Z_q are only of autonomous type for all $q \in Q$.*

Let $\psi_q(x,t)$ be the flow of the vector field f_q starting at $x \in X_q$. We assume that flows are ultimately bounded in the following sense:

Assumption 6.2 *For all $x \in X_q$ for all $q \in Q$, and for all $t \in \mathbb{R}^+$, $\sup_{t\in\mathbb{R}^+} \|\psi_q(x,t)\| < \infty$.*

The norm $\|\cdot\|$ on X_q is assumed to be one of the typical norms on $\mathbb{R}^n$. We now recall the concept of the positive limit set of the trajectories of a continuous dynamical system [30]:

Definition 6.1 (Positive limit set) *Let $\psi_q(x,t)$ be a flow of the system Σ_q starting from $x \in X_q$. Then $y \in X_q$ is said to be a positive limit point of $\psi_q(x,t)$ if there is a sequence $\{t_n\}$, with $t_n \to \infty$ as $n \to \infty$, such that $\psi_q(x,t_n) \to q$ as $n \to \infty$. The set of all limit points of $\psi_q(x,t)$, $\forall x \in X_q$ is called the positive limit set of $\psi_q(t)$.*

We define the distance of a point to a set as [30]:

Definition 6.2 (Distance to a subset of X_q) *The distance of a point x of the state space X_q to a subset $Y \subset X_q$ is defined as $\operatorname{dist}\bigl(x, Y\bigr) \triangleq \inf_{y\in Y} \|x-y\|$.*

We study a subset of the hybrid system given by Definition 2.1 and Assumption 6.1.

Assumption 6.3 *Consider the system given by Definition 2.1 and Assumption 6.1. We assume that*

- *$G_q(s) \neq \emptyset$, $\forall s \in S_q$ for all $q \in Q$;*
- *$Z_q(x,s) \neq \emptyset$, $\forall x \in G_q(s)$, for all $s \in S_q$ and for all $q \in Q$;*
- *For each $q \in Q$, the positive limit set L_q^+ of the $\psi_q(t)$ satisfies $L_q^+ \subseteq G_q$, where $G_q = \bigcup_{s\in S_q} G_q(s)$.*

The last condition implies that the positive limit sets of the flows are contained in the guards. Note that we do not assume global Lipschitz continuity of f_q; instead, we use the boundedness condition of Assumption 6.2, which also ensures the existence of a positive limit set L_q^+ for the flows of f_q for all $q \in Q$.

Given a $q \in Q$, the positive limit set of $\psi_q(t)$, L_q^+, may be disconnected. For a given discrete state $q \in Q$, let $L_q^+(i)$ $i = 1,\ldots,\ell$ be a disconnected component of L_q^+. We assume that $\ell < \infty$, considering the verification of this condition a control design issue to be addressed in the future. Whenever a state space X_q contains multiple disconnected components of L_q^+, (and given that each component belongs to a different guard) we partition the given X_q into regions that have a single, common component $L_q^+(i)$ as shown in Figure 6.1.

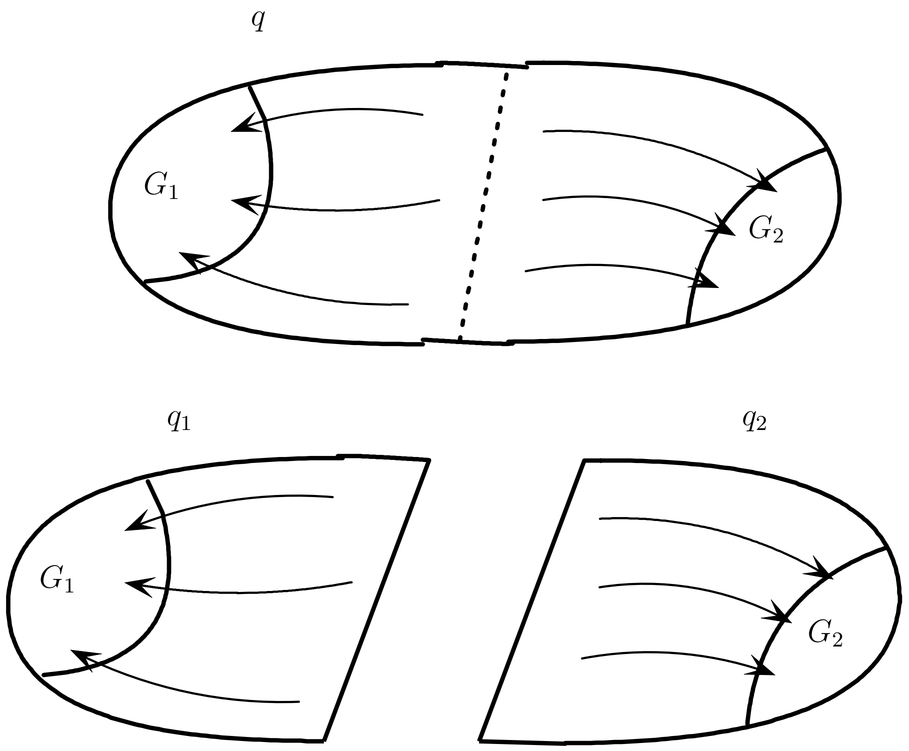

Figure 6.1: A domain is partitioned according to the inclusion of connected components of the positive limit set within the guards. Note that the boundary between the regions of attraction is unstable, which implies that any perturbation on the system could force the dynamics to jump between regions of attraction if the state of the system is close enough to the boundary. A possible solution to this may be to create a third partition that contains this boundary and leads to both transition guards.

This refinement also guarantees that the flows of $f_q(x,t)$ in X_q do not activate any other guards before reaching the one where L_q^+ is contained. However note that the boundary between two adjacent regions of attraction of two different transition guards is an unstable equilibrium set. This implies that if the continuous state of the system is close enough to the boundary, any perturbation on the continuous dynamics could force the system to move between the adjacent regions of attraction. A possible modification to overcome this limitation may be to add a third partition to the refinement in Figure 6.1 that contains the boundary between the two original partition sets. Therefore any continuous flow starting in this partition could lead to any transition guard in Figure 6.1 creating nondeterminism in the region of the continuous space of the system that is close to the unstable boundary.

A timed automaton is defined here as follows:

Definition 6.3 (Timed Automaton [2]) *A Timed Automaton is a hybrid system given as Definition 2.1 and Assumption 6.1 that satisfies the following properties:*

- *For every* $(q_0, x_0) \neq (\hat{q}_0, \hat{x}_0) \in \text{Init}$*,* $q_0 \neq \hat{q}_0$.
- *The set* X_q *is a rectangular set and the vector field is given by* $f_q(x,t) = 1$ *for all* $q \in Q$
- *For each discrete transition* $s \in S_q$ *for all* $q \in Q$*, the set* $G_q(s)$ *is a rectangular set.*
- *For every discrete transition* $s \in S_q$ *for all* $q \in Q$*, and for all* $x \in G_q(s)$*,* $Z_q(x,s) = \{y \in X_s | y = x$ *or* $y = c$*, where* c *is a constant vector.*

6.3 A New Hybrid Metric

We define in this section a notion of a hybrid distance that provides information about both the continuous and the discrete distances between two hybrid states. Since a graph $\mathcal{G}_{\mathbf{H}}$ is directly associated with a hybrid system $\mathbf{H}$, we identify its nodes with the modes of $\mathbf{H}$, which represent a distinct behavior of the underlying dynamical system.

Definition 6.4 (Discrete Distance) *Let the distance between two discrete states of a hybrid system q_1 and q_2 be the length of the shortest path*[1] *from mode q_1 to mode q_2 in the directed graph $\mathcal{G}_{\mathbf{H}}$, associated with the hybrid system $\mathbf{H}$. This distance is denoted by $d_D(q_1, q_2)$.*

Definition 6.5 *Let $A = A(\mathcal{G}_{\mathbf{H}})$ be the adjacency matrix of the directed graph $\mathcal{G}_{\mathbf{H}}$ associated with $\mathbf{H}$. The entries of A have their rows and columns indexed by the pair $(q_i, q_j) \in Q \times Q$. Each entry (q_i, q_j) will be 1 when a transition is possible from q_i to q_j (a label $s_{q_i} = q_j$) and 0 otherwise.*

The adjacency matrix has the property that its r power will give as an entry at position (q_i, q_j) the number of directed paths from q_i to q_j of length r [24]. Based on this property we propose a procedure to calculate the discrete distance between to discrete modes in a hybrid system $\mathbf{H}$:

Lemma 6.1 *The discrete distance $d_D(q_1, q_2)$ can be calculated as follows:*

$$d_D(q_1, q_2) = \begin{cases} \min_{r \in \mathbb{N}} \{r : (A^r)_{(q_1,q_2)} \neq 0\} & q_2 \in Reach(q_1) \\ \infty & \text{otherwise} \end{cases} \tag{6.1}$$

where $Reach(q) = Reach(h)$ such that the discrete state of h is q.

Definition 6.6 (Hybrid Distance) *Let the distance between two hybrid states h_1 and h_2 be $d_H(h_1, h_2) = \tanh(\|x_1 - x_2\|) + d_D(q_1, q_2)$, where $h_i = (q_i, x_i)$ for $i = 1, 2$ and $\|.\|$ is the norm on $\mathbb{R}$.*

Using the $\tanh(\cdot)$ function of the norm in the distance expression gives different weight to the discrete part of the hybrid state; (hybrid) states in different discrete modes are considered to be much further apart than any continuous states in the same mode.

[1]For a definiton of a path, see [24].

The metric provided by Definition 6.6 as a measure of the distance between two hybrid states is composed by two completely separable parts: an integer part ($d_D(q_1, q_2)$) that is a function of the number of discrete transitions that H has to make to go from h_1 to h_2; and a fractional part ($0 \leq \tanh(\|x_1 - x_2\|) \leq 1$ as proven below) that is a function of the (induced) distance between the continuous components of h_1 and h_2. Also note that this distance notion can be extended to sets $M_1, M_2 \subseteq Q \times X$ by defining $d_H(M_1, M_2) = \inf\{d_H(h_1, h_2) : h_1 \in M_1, h_2 \in M_2\}$.

In what follows, we show that the proposed function can serve as a metric on the space $Q \times X$, with the exception of symmetry: the existence of a path from q_1 to q_2 does not imply the existence of a path of the same length from q_2 to q_1. This distinction is not made in the related constructions found in [37, 51].

Proposition 6.1 *The hybrid distance $d_H(h_1, h_2)$ is zero if and only if $q_1 = q_2$ and $x_1 = x_2$.*

Proof: First note that the continuous portion of the hybrid distance $\tanh(\|x_1 - x_2\|)$ will only be zero when the argument of $\tanh(.)$ is zero and this will happen only when $x_1 = x_2$. Second note that, by definition 6.4, the discrete part of the hybrid distance $d_D(q_1, q_2)$ will be zero only when $q_1 = q_2$ which proves the proposition. ■

Proposition 6.2 *The hybrid distance $d_H(h_1, h_2) \geq 0$ for all q_1, q_2, x_1, and x_2.*

Proof: The $\tanh(\cdot)$ function is positive for positive arguments and zero if the argument is null. Since $\|x_1 - x_2\|$ is positive for all $x_1 \neq x_2$ and zero for $x_1 = x_2$ then $\tanh(\|x_1 - x_2\|)$ will be positive for all $x_1 \neq x_2$ and zero for $x_1 = x_2$. On the discrete part of the hybrid distance r represents the number of jumps that an state would have to take to reach another state. Since this variable is always nonnegative, and zero only for $q_1 = q_2$, $d_D(q_1, q_2)$ will always be nonnegative proving the proposition. ■

Proposition 6.3 *The hybrid distance $d_H(h_1, h_2)$ satisfies the triangle inequality*

$$d_H(h_1, h_3) \leq d_H(h_1, h_2) + d_H(h_2, h_3)$$

for all q_1*,* q_2*,* q_3*,* x_1*,* x_2*, and* x_3*.*

To prove the above we will need the following Lemmas:

Lemma 6.2 $d_D(q_1, q_3) \leq d_D(q_1, q_2) + d_D(q_2, q_3)$ *for all* q_1*,* q_2*, and* q_3*.*

Proof: Consider a directed graph that contains $q_1, q_2, q_3 \in Q$ and analyze three cases:

1. If $q_1 = q_3$, then $d_D(q_1, q_3) = 0$ by proposition 6.1. Moreover it has been proven (proposition 6.2) that for every pair of modes $q_m, q_n \in Q$ the distance $d_D(q_m, q_n) \geq 0$. So $d_D(q_1, q_3) = 0 \leq d_D(q_1, q_2) + d_D(q_2, q_3)$.

2. If $q_1 \neq q_3$ and $q_3 \notin \text{Reach}(q_1)$ then $d_D(q_1, q_3) = \infty$, because there does not exist any path from q_1 to q_3. This implies that there will not exist any path between at least one of the pairs q_1, q_2 or q_2, q_3 causing at least one of the distances $d_D(q_1, q_2)$ or $d_D(q_2, q_3)$ to be infinite. Thus $d_D(q_1, q_3) = \infty = d_D(q_1, q_2) + d_D(q_2, q_3)$.

3. If $q_1 \neq q_3$ and $q_3 \in \text{Reach}(q_1)$ then $d_D(q_1, q_3) < \infty$. So assume without loss of generality that $q_3 \in \text{Reach}(q_2)$ and $q_2 \in \text{Reach}(q_1)$ (If any of this conditions is not satisfied then the lemma is trivially satisfied because at least one of the distances in the right hand side of the inequality would be infinite). Note that the minimum number of transitions to go from q_i to q_j for all $i = 1, 2 \text{ and } j = 2, 3 : i \neq j$ is given by $d_D(q_i, q_j)$. So if the minimum path from q_1 to q_3 included q_2 then $d_D(q_1, q_3) = d_D(q_1, q_2) + d_D(q_2, q_3)$. Otherwise, if the minimum path between q_1 and q_3 did not include q_2 then moving the discrete state from q_1 through q_2 to q_3 would create a path with more jumps than going directly from q_1 to q_3, i.e. $d_D(q_1, q_3) < d_D(q_1, q_2) + d_D(q_2, q_3)$.

These three cases together prove that $d_D(q_1, q_3) \leq d_D(q_1, q_2) + d_D(q_2, q_3)$ for every $q_1, q_2, q_3 \in Q$. ■

Lemma 6.3 $\tanh(\|x_1 - x_3\|) \leq \tanh(\|x_1 - x_2\|) + \tanh(\|x_2 - x_3\|)$ *for all* x_1*,* x_2*, and* x_3.

Proof: It follows directly from the properties of the $\tanh(\cdot)$ function. ■

We now prove Proposition 6.3:

Proof: The triangle inequality in Proposition 6.3 can be rewritten as follows $\tanh(\|x_1 - x_3\|) + d_D(q_1, q_3) \leq \tanh(\|x_1 - x_2\|) + d_D(q_1, q_2) + \ldots + \tanh(\|x_2 - x_3\|) + d_D(q_2, q_3)$. Note that if $a \leq b$ and $c \leq d$ then $a + c \leq b + d$. Thus the proof follows from this fact and Lemmas 6.2 and 6.3. ■

6.4 Hybrid Notions of Stability

Definition 6.7 (Invariant Set [37]) *A set* $\mathrm{Inv} \subseteq \mathit{Reach}_{\mathbf{H}}$ *is invariant if* $\forall h_o \in \mathrm{Inv}$*, all* $(\tau, \mathrm{q}, \mathrm{x}) \in \chi^S(h_0)$*, all* $n \in \langle\tau\rangle$*, and all* $t \in [\tau^{n-1}, \tau^n)$*,* $(q^n, x^n(t)) \in \mathrm{Inv}$.

Definition 6.8 (Stable Invariant Set [37]) *An invariant set* Inv *is called*

- *stable if for all* $\epsilon > 0$ *there exists a* $\delta > 0$ *such that for all* $(h_0) \in \mathit{Reach}_{\mathbf{H}}$ *with* $d_H((h_0, \mathrm{Inv}) < \delta$*, all* $(\tau, \mathrm{q}, \mathrm{x}) \in \chi^S(h_0)$*, all* $n \in \langle\tau\rangle$*, and all* $t \in [\tau^{n-1}, \tau^n)$*,* $d_H((q^n, x^n(t), \mathrm{Inv})) < \epsilon$*;*
- Inv *is called asymptotically stable if it is stable and in addition there exists a* $\Delta > 0$ *such that for all* $(h_0) \in \mathit{Reach}_{\mathbf{H}}$ *with* $d_H(h_0, \mathrm{Inv}) < \Delta$ *and all* $(\tau, \mathrm{q}, \mathrm{x}) \in \chi^\infty(h_0)$*,* $\lim_{t \to |\tau|} d_H((q^n, x^n(t)), \mathrm{Inv}) = 0$.

Note that positive limit sets L_q^+ are invariant but not necessarily stable. The existence of L_q^+ merely suggests that the hybrid trajectory will approach it *in time,* not that it will stay in its neighborhood. We use the positive limit sets to ensure that a transition between discrete modes will occur in finite time. Stability of a hybrid system $\mathbf{H}$, is understood as convergence to a asymptotically stable invariant set Inv. For simplicity, we will assume that $\mathbf{H}$ has only one (globally) asymptotically stable invariant set Inv:

Assumption 6.4 *The hybrid system $\mathbf{H}$ has only one asymptotically stable invariant set denoted (Q_{eq}, X_{eq}). In addition assume that every $q \in Q$ there exists a unique possible discrete jump $s \in S_q$, and that the associated guard $G_q(s)$ containing a connected component of L_q^+ is "forced" (the transition must occur).*

6.5 Finite Time Abstraction for Continuous Dynamics

For this section consider an autonomous continuous dynamical system $\Sigma = (X, f, \mathbb{R}^+)$. We say that two points x_1 and x_2 in X are *asymptotically equivalent* (denoted $x_1 \sim x_2$) if their positive limit points belong to the same limit set. However a finer partition of the state space could be obtained by comparing the distances of the flows of two points, x_1 and x_2, to the same connected component L_k^+ at time T. We formally define this idea as *finite time abstraction*:

Definition 6.9 (Finite-time Equivalence relations) *Consider an autonomous system Σ, where X is a compact subset of $\mathbb{R}^m$, and let the flows of Σ belong in X for all $t \in \mathbb{R}^+$. Let $L^+ = \bigcup_{k=1}^{\ell} L^+(k)$ be the positive limit of Σ, where each $L^+(k)$ is simply connected. We define an equivalence relation $\sim_T$ on X as follows: Two points x_1, $x_2 \in X$ are said to belong to the same T-equivalence class, denoted $x_1 \sim_T x_2$, if*

1. *$x_1 \sim x_2$, and*
2. *if for some k, $\lim_{t\to\infty} \operatorname{dist}\big(\psi(x_1, t), L^+(k)\big) = \lim_{t\to\infty} \operatorname{dist}\big(\psi(x_2, t), L^+(k)\big) = 0$, then $\operatorname{dist}\big(\psi(x_1, T), L^+(k)\big) = \operatorname{dist}\big(\psi(x_2, T), L^+(k)\big)$.*

The first condition excludes the possibility of one point belonging into different T-equivalence classes. A finite time abstraction partitions the state space according to the distance of the flows of the points at time T, to the component $L^+(k)$ of the positive limit set which they converge to. We use $L^+(k) + \mathcal{B}_d$ to denote the set $\{x + y \mid x \in L^+(k), y \in \mathcal{B}_d\}$, where $\mathcal{B}_d \subseteq \mathbf{R}^m$ is the open ball of radius d centered at the origin.

Definition 6.10 (Finite Time Abstraction) *Consider a system Σ, where X is a compact subset of $\mathbb{R}^m$, and let $\psi(x,t)$ be the flow of f from $x \in X$. Suppose that the flows of Σ belong in X for all $t \in \mathbb{R}^+$ and that Σ has a positive limit set $L^+ = \bigcup_{k=1}^{\ell} L^+(k)$. The finite-time T-abstraction of Σ is a (set valued) map, that associates each point $x \in X$ to the set $L^+(k) + \mathcal{B}_d$, where k is such that $\lim_{t\to\infty} \operatorname{dist}\big(\psi(x,t), L^+(k)\big) = 0$, $d = \operatorname{dist}\big(\Phi(x,T), L^+(k)\big)$, and $\mathcal{B}_d$ is the ball of radius d centered at the origin.*

In this sense, a finite-time T-abstraction will contain information about "how close to destination" the flows from different points will be, in time T.

6.6 Discrete Asymptotic Abstraction

Consider a hybrid system $\mathbf{H}$ satisfying the conditions of Assumption 6.3, and let $\psi_q(y,t)$ be the flow of f_q from $y \in X_q \setminus G_q(s,q')$ where $s \in S_q$. Given that $L_q^+ \subset G_q(s,q')$, there will be a (finite) upper bound on the time needed for the flow of f_q to reach $G_q(s,q')$ from any point $x \in X_q$. We denote this bound Θ_q. The existence of Θ_q is guaranteed by the definition of the positive limit set L_q^+, and the fact that the latter is completely contained in the guard.

Definition 6.11 (Finite Time Mode Abstraction) *The Θ_q-abstraction of the continuous dynamical system Σ_q in mode q is given as the image of the constant map: $(X_q, f_q) \to Q \times \mathbb{R}^+ : (x, f_q(x,t)) \mapsto (q, \Theta_q)$.*

In this way, the continuous dynamics are dropped completely. All the information that remains is an indication of "how long it takes to reach the guard." This is the concept that allows us to abstract the continuous dynamics of the hybrid system $\mathbf{H}$ into a the clock dynamics of a timed automaton.

In abstracting $\mathbf{H}$ into a Timed Automaton $\tilde{\mathbf{H}}$, we have to consider the equilibrium hybrid set (Q_{eq}, X_{eq}) separately. An ϵ-neighborhood of X_{eq} (for an arbitrarily small ϵ,) will define the continuous space that is mapped to this new "final mode" during the abstraction procedure. Since the system's flows stay in the ϵ-neighborhood of X_{eq} once this has been reached we let $\Theta_{\tilde{q}_f} = c$ for any $c > 0$. This is possible because, as can be seen in Definition 6.12, once the system reaches $\tilde{q}_f$, it will either stay in that state by continuous evolution (if $\Theta_{\tilde{q}_f} = \infty$) or will periodically execute a discrete transition the same state $\tilde{q}_f$ (if $\Theta_{\tilde{q}_f} < \infty$).

Based on Assumption 6.4, one can obtain the following constructive process for defining the Timed Automaton $\tilde{\mathbf{H}}$ that captures the asymptotic behavior of $\mathbf{H}$. A pictorial representation of this abstraction is shown in Figure 6.2.

Definition 6.12 (Abstract timed automaton) *Let the abstract timed automaton $\tilde{\mathbf{H}}$ be a hybrid system as given in Definition 2.1 and Assumption 6.1 such that:*

- $\tilde{Q} = Q \bigcup \{\tilde{q}_f\}$, *where $\tilde{q}_f$ is a new mode that represents $X_{eq} \in X_{Q_{eq}}$.*

- $\tilde{\Sigma}_{\tilde{q}} = (\tilde{X}_{\tilde{q}}, \tilde{f}_{\tilde{q}}, \mathbb{R}^+)$, *where $\tilde{X}_{\tilde{q}} = \{(\lambda, \gamma)^T ; \lambda \leq \Theta_{\tilde{q}}\}$ for all $\tilde{q} \in \tilde{Q}$, and $\tilde{f}_{\tilde{q}}(\lambda, \gamma) = \mathbf{1}$ for all $\tilde{q} \in \tilde{Q}$ (clock dynamics).*

- $\tilde{\mathbf{S}} = \{\tilde{S}_{\tilde{q}}\}_{\tilde{q} \in \tilde{Q}}$ *where $\tilde{S}_{\tilde{q}} = S_q$ for all $\tilde{q} \in \tilde{Q}$ $\{\tilde{q}_f\}$, and $S_{\tilde{q}_f} = \{\tilde{s}_f\}$, where $\tilde{s}_f = \tilde{q}_f$ (A discrete transition from $\tilde{q}_f$ leads to itself).*

- $\tilde{\mathbf{G}} = \{\tilde{G}_{\tilde{q}}\}$ *where $G_{\tilde{q}_f}(s) = \{(\lambda, \gamma)^T ; \lambda \geq \Theta_{\tilde{q}}\}$, for all $\tilde{q} \in \tilde{Q}$.*

- $\tilde{\mathbf{Z}} = \{\tilde{Z}_{\tilde{q}}\}_{\tilde{q} \in \tilde{Q}}$ *where $\tilde{Z}_{\tilde{q}} : \tilde{G}_{\tilde{q}} \times \tilde{S}_{\tilde{q}} | (\lambda^{n+1}(t), \gamma^{n+1}(t))^T = (0, \gamma^n(t))^T$.*

where $(\tilde{q}, \tilde{x}) \in \bigcup_{\tilde{q} \in \tilde{Q}} \tilde{X}_{\tilde{q}} \times \tilde{Q}$ is the state of $\tilde{\mathbf{H}}$.

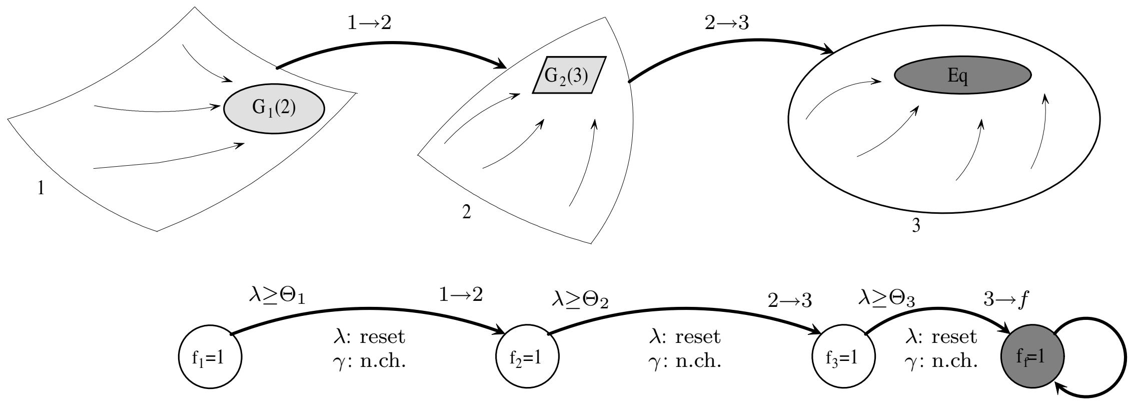

Figure 6.2: Pictorial representation of hybrid system's abstraction. Each discrete mode on the hybrid system (top) has a corresponding discrete mode in the abstract timed automaton (bottom). The timed automaton has an extra discrete mode that represents the equilibrium set of the hybrid system. The timed automaton executes the discrete transitions after the maximum time (Θ_q) the continuous dynamics in the hybrid system take to converge to the transition guards.

We now present the two main results of this chapter. The goal of these two theorems is two study the stability, and the reachability of a hybrid system using an abstract version of it. We do this by abstracting most of the continuous dynamics (Definition 6.12) of the hybrid system keeping only the relevant information to preserve the stability and reachability properties of the system.

Theorem 6.1 (Asymptotic Stability of $\mathbf{H}$) *If $\mathbf{H}$ is asymptotically stable (AS) with a L being an ϵ-neigborhood of its AS invariant set (Q_{eq}, X_{eq}), then the timed automaton $\tilde{H}$ constructed as in Definition 6.12 is asymptotically stable in the sense of Definition 6.8, with $(\tilde{q}_f, \cdot)$ its asymptotically stable invariant state.*

Proof: If $\mathbf{H}$ is asymptotically stable (AS) then by definition there exists a $\delta > 0$ for all $\xi > 0$, such that for all $h_0 \in Reach_{\mathbf{H}}$ with $d_H(h_0, L) < \delta$, every execution $(\tau, \mathbf{q}, \mathbf{x}) \in \chi^S(h_0)$ satisfies $d_H((q^i, x^i(t)), L) < \xi$ for all $i \in \langle\tau\rangle$ and $t \in [\tau^i, \tau^{i+1})$, and there also exists a $\Delta > 0$ such that for all $h_0 \in Reach_{\mathbf{H}}$ with $d_H(h_0, L) < \Delta$ every infinite execution

$(\tau, \mathbf{q}, \mathbf{x}) \in \chi_H^\infty(q_0, x_0)$ satisfies $\lim_{t\to|\tau|} d_H((q^i, x^i(t)), L) = 0$. Then by construction of $\tilde{H}$, and the definition of $d_H(h, h')$ there exists a $\tilde{\delta} = \lfloor\delta\rfloor+1$ for all $\tilde{\xi} = \lfloor\xi\rfloor+1$ ($+1$ is added due to the addition of $\tilde{q}_f$ in Def. 6.12) such that for all $\tilde{h}_0 \in Reach_{\tilde{\mathbf{H}}}$ with $d_D(\tilde{q}^0, \tilde{q}_f) < \tilde{\delta}$, every execution $(\tilde{\tau}, \tilde{\mathbf{q}}, \tilde{\mathbf{x}}) \in \chi_{\tilde{\mathbf{H}}}(\tilde{h}_0)$ satisfies $d_D(\tilde{q}^i, \tilde{q}_f) < \tilde{\xi}$ for all $i \in \langle\tau\rangle$ and $t \in [\tau^i, \tau^{i+1})$ and there also exists a $\tilde{\Delta} = \lfloor\Delta\rfloor$ such that for all $\tilde{h}_0 \in Reach_{\tilde{\mathbf{H}}}$ with $d_D(\tilde{q}^0, \tilde{q}_f) < \tilde{\Delta}$ every infinite execution $(\tilde{\tau}, \tilde{\mathbf{q}}, \tilde{\mathbf{x}}) \in \chi_{\tilde{\mathbf{H}}}^\infty(\tilde{h}_0)$ satisfies $\lim_{t\to|\tilde{\tau}|} d_D(\tilde{q}^i, \tilde{q}_f) = 0$, thus making $\tilde{q}_f$ the a.s. discrete invariant set of $\tilde{H}$. Since the continuous part of the AS invariant set of $\tilde{H}$ is the whole domain of q_f, the theorem is proved. ■

Let $(q(T), x(T)) \in Reach_{\mathbf{H}}$ denote the state of the hybrid system $\mathbf{H}$ at time T. The next theorem states that the (finite time) reachability properties of $\mathbf{H}$ are preserved by $\tilde{\mathbf{H}}$:

Theorem 6.2 *(Reachability of* $\mathbf{H}$*) If $(q(T), x(T)) \in Reach_{\mathbf{H}}$ there exists a $k \in \langle\tau\rangle$ such that after some execution $\chi(h_0)$, $q(T) = q^k$, $x(T) = x^k(T)$ with $T \in [\tau^k, \tau^{k+1})$. Moreover T will be upper-bounded by $\gamma^k(\tau^k)$ (the second component of the continuous state of* $\tilde{\mathbf{H}}$ *at the end of mode k), i.e. $T \leq \gamma^k(\tau^k)$.*

Proof: Let the hybrid state $(q, x) \in Reach_{\mathbf{H}}$ then the abstract hybrid state $\tilde{q}, \tilde{x}$ is in $Reach_{\tilde{\mathbf{H}}}$ by Definition 6.12. Assume that the hybrid automaton starts at the initial conditions h_0. Then there exists a hybrid execution $\chi(h_0) = (\tau, \mathbf{q}, \mathbf{x})$ that maps the initial condition h_0 to an state (q, x) such that q is equal to the discrete state at a $k \in \langle\tau\rangle$ and the corresponding x is equal to the continuous state at a $k \in \langle\tau\rangle$ and at $T \in [\tau^k, \tau^{k+1})$, i.e. $(q, x) = (q^k, x^k(T))$ such that $k \in \langle\tau\rangle$ and $T \in [\tau^k, \tau^{k+1})$. This state (q, x) is the state of the hybrid system at a time T: $(q, x) = (q(T), x(T))$ along the execution $\chi(h_0) = (\tau, \mathbf{q}, \mathbf{x})$. If a timed automaton $\tilde{\mathbf{H}}$ is constructed as in definition 6.12 $(\lambda, \gamma)^T \in \tilde{X}$ correspond to the local and global clocks of $\tilde{\mathbf{H}}$. So by the definition of $\tilde{D}$ and $\tilde{G}$,

$$\lambda^k(\tau^{k+1}) = \Theta_{q^k} \geq \tau^{k+1} - \tau^k \tag{6.2}$$

By the construction of $\tilde{R}$ in Definition 6.12 $\gamma^k(\tau^{k+1}) = \sum_{i=1}^k \lambda^i(\tau^{i+1})$. Then using (6.2) and noting that $T \in [\tau^k, \tau^{k+1}]$, and that $\gamma^k(\tau^{k+1}) \geq \sum_{i=1}^k(\tau^{i+1} - \tau^i) \geq T$, we obtain that

$T \leq \gamma^k(\tau^{k+1})$. ■

6.7 On the Application of Abstractions to the Resource Allocation Problem

In this section we briefly discuss how the abstraction ideas could be beneficial to the resource allocation problem. The basic approach is to take advantage of the stable continuous dynamics on the agents and nodes of the system, such that the system's description can be simplified without loosing relevant information for the final objective.

Note in Chapter 3 that the main design objective is to have the agents converge to a configuration that coincides with the optimal allocation of the network resources among all the agents located in it. This design objective is not concerned with the steps that the agents and nodes take to converge to such configuration. Also note that the continuous dynamics of the agents and nodes converge to a single asymptotically stable equilibrium point for each possible configuration of the network. This implies that the continuous dynamics may be decoupled from the discrete dynamics if every time the agents and nodes execute a discrete transition, the continuous dynamics have enough time to converge to their equilibria, just as it was done in the simulations of Section 5.5.

Therefore, following the ideas explained above, if the continuous dynamics on the agents and nodes are substituted by the information about the (optimal) equilibrium they reach in their current configuration, a discrete state network system may be formed by mapping the nodes in the network to one type of nodes in a bipartite graph, and the agents in the network to a second type of nodes in a bipartite graph as shown in Figure 6.3. To keep the information about the connectivity in the network and the location of agents among nodes, two types of links may be superimposed on graph. *Location links* would provide information about the location of agents in the network. Only one location link

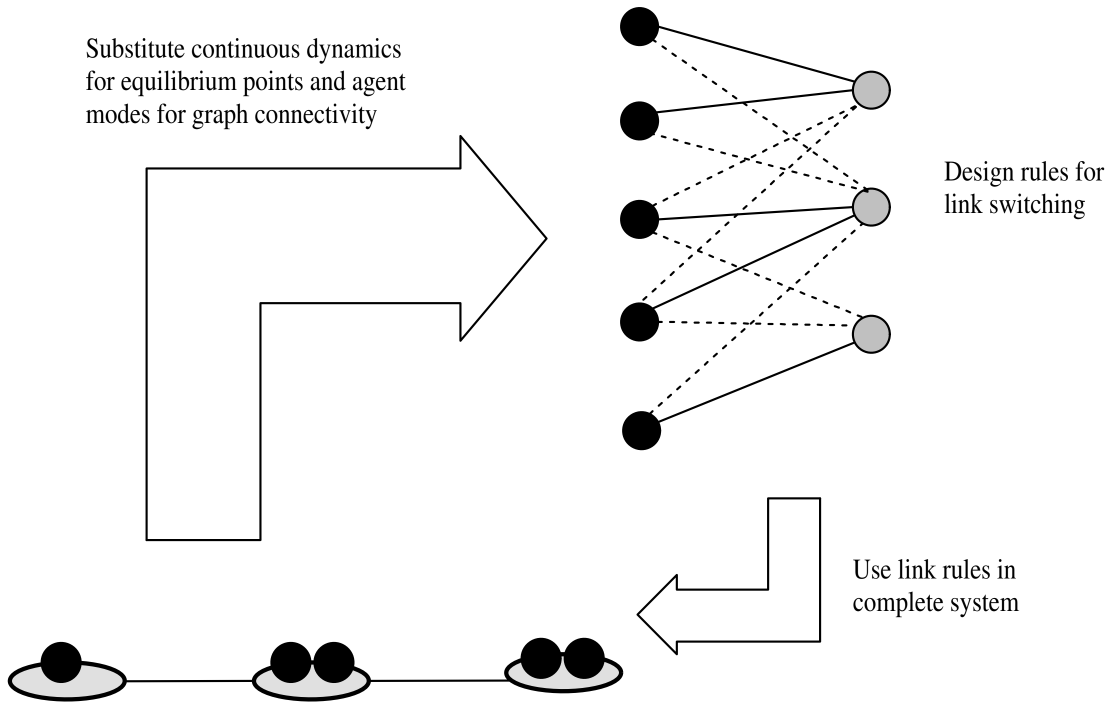
Substitute continuous dynamics for equilibrium points and agent modes for graph connectivity
Design rules for link switching
Use link rules in complete system

Figure 6.3: Pictorial representation of a possible abstraction of the continuous dynamics in the resource allocation problem. The continuous dynamics may me substituted by the optimal solutions of the optimization problem, keeping only the discrete dynamics. Then the discrete dynamics may be designed and the obtained controllers should be applicable on the original system.

is allowed for each agent. *Information links* would substitute the connectivity information of the original network by links between an agent and the nodes that are connected to the node this agent occupies. In this form the agents could obtain information from all the nodes that are connected to the node they occupy, and would also be able to migrate to those nodes if desired. Other information to be kept on the abstraction procedure include the resources available in the nodes, and the benefit function of each agent.

The evolution of this new system description results as follows. Agents and nodes start at an initial configuration. The nonlinear optimization problem is solved within each node. Then the agents may decide to execute transitions among nodes. After these transitions, the optimization problem is solved again, and the system keeps on repeating interweaved steps of transitions and optimization solutions, until it converges to the optimal configuration. Note that this system does not involve continuous dynamics, but would still be capable of converging to an optimal configuration. The benefit of using this description is that the effort of numerical analysis or design tasks like the one performed in Section 5.5 would be greatly reduced by a simplified description of the system's model.

6.8 Conclusions

We introduce the notion of Finite Time Mode Abstraction for a special class of (convergent) hybrid systems. According to this concept, most of the continuous dynamics of the hybrid system is abstracted away, leaving only information about the time that takes a continuous state to reach a transition guard within each particular discrete mode. This information is then used to construct a timed automaton which is shown to preserve the stability and reachability properties of the original hybrid system. Our current analysis applies to the class of hybrid automata with one guard per mode and only one asymptotically stable equilibrium set, but we suggest a procedure for generalization more general classes of hybrid systems, through refinement of their discrete modes. We consider this work as the first step in a path that will allow us to map continuous and hybrid dynamics into (almost completely) discrete ones. We also define a new distance for hybrid dynamical systems, composed by two completely identifiable parts: a discrete part that is the number of transitions separating two discrete modes, and a continuous part that is a function of a standard distance (induced by a norm) between their corresponding continuous states. We finally discuss the applicability of the ideas in this chapter to the resource allocation problem the motivates this dissertation.

Chapter 7

Conclusion

We have explored the modeling and control of multiagent systems with hybrid interacting dynamics. These systems are essentially composed of multiple individual hybrid systems whose dynamics interact at both the continuous and discrete levels. These types of systems can be seen in multivehicle application where the individual vehicles are capable of operating at different discrete operating conditions, in networks of sensors, actuators and embedded systems, and in communication networks like that in the motivation for this dissertation.

We have studied these interacting hybrid systems from various different perspectives: Motivated by a control problem in future communication networks, we formulated a hybrid framework to address the problem of allocating continuous resources among agents moving in a discrete environment. We then studied basic dynamical properties for general multiagent systems with hybrid interacting dynamics, which we called Interconnected Hybrid Systems. We also explored the application of randomized optimization techniques for the optimal control of individual and multiagent hybrid systems, and finally formulated a new abstraction technique for the model simplification of individual hybrid systems with a brief discussion about its possible application to multiagent systems with hybrid dynamics.

The hybrid framework for resource allocation among agents moving in discrete locations is designed in three major steps: The design objective is expressed as a mixed integer nonlinear optimization problem. This problem is then decomposed into two hierarchical optimization problems, a high level integer optimization, and a lower level nonlinear programming problem. And this decomposition is then used to propose hybrid models for both agents and nodes, and design the continuous dynamics using resource allocation theory [62]. The system obtained at this point has completely designed continuous dynamics that optimize the resource allocation problem within each node, and partially designed discrete dynamics that represent the movement of agents across the network and the changes in the network.

The dynamical properties studied for general multiagent systems with hybrid interacting dynamics include the existence and uniqueness of the system's executions, which are then applied in the design of the discrete dynamics of the agents in the resource allocation problem. To study the dynamical properties we introduce a new type of system called Interconnected Hybrid System (IHS), which provides a general model for studying multiagent systems with hybrid interacting dynamics. We recast several hybrid concepts into the IHS framework. We then provide conditions for the existence and uniqueness of an interconnected hybrid execution in term of the dynamics of the individual agents. The merit of this result is that it is possible to guarantee that the whole IHS has a unique interconnected hybrid execution just by verifying some conditions in the design of the individual agents. These conditions are then applied to the design of the general structure of the discrete dynamics of the agents in the resource allocation problem, such that the network is guaranteed to be "well behaved" in terms of the existence and uniqueness of the system's executions.

We also explore the use of randomized optimization techniques for the optimal control of individual and interacting hybrid systems. To do this we transform a general hybrid optimal control problem into a randomized hybrid optimal control one, and then outline how to apply a simple randomized algorithm in the optimization of hybrid systems' performance. In particular we study the optimization of two important types of hybrid systems in detail: a time driven switching system, and a state driven hybrid system. The numerical results we obtain are comparable to those using existing gradient techniques for hybrid systems. The advantage of the randomized approach is that the analytical complexity is simplified, at the cost of not being able to guarantee that the obtain solution is *the optimal*. We then apply the same randomized algorithm to the optimization of the agents discrete dynamics in the resource allocation problem. The numerical results we obtain indicate that the randomized technique yields a system whose the agents converge to a point that is close to the optimal configuration of agents in the network as specified in the design objective.

Finally, we explore the use of abstractions techniques for the model simplification of individual hybrid systems, and discuss its applicability to the resource allocation problem. The abstraction technique we propose substitutes the continuous dynamics of a hybrid system by clocks, generating a timed automaton that keep track of the time that takes the original hybrid system to converge to an asymptotically stable invariant set. The timed automaton is shown to retain the stability and reachability properties of the original hybrid systems, but contains almost purely discrete dynamics. Additionally we propose a new hybrid metric capable of providing information about the distance between the continuous components of two hybrid states, and the number of jumps required to go between the discrete states of those two hybrid states.

The work that is summarized in this dissertation has opened up several interesting paths for future research, which include the use of randomized algorithms for optimal control of uncertain hybrid system, the study of consensus and cooperative control problems for multiagent systems with hybrid interacting dynamics, the formulation of asymptotic abstractions for more general hybrid systems, the complete control of agents moving across a network with time varying topology, and the actual application of abstraction techniques to multiagent hybrid systems as discussed in Chapter 6.

The resource allocation problem we studied in this dissertation got constrained to fixed network topology due to the limitation of some of the theoretical results presented here. The inclusion of a time varying network topology would require the addition of event driven dynamics to the IHS framework in Chapter 4, and the extension of the randomized algorithm in Chapter 5 to one that can deal with event uncertainties.

The extension of the randomized optimal control algorithm in Chapter 5 to uncertain dynamics could be approached using statistical learning theory. Other interesting alternatives may include stochastic optimization algorithms and reinforcement learning, which have been scarcely explored in the hybrid systems domain. Other important directions include the formal study of different sampling procedures for the hybrid case, and the control of more general dynamics (with or without uncertainty).

The discrete abstractions presented in Chapter 6 are currently applicable to a particular type of hybrid system with one outgoing transition for each discrete state, and limit set contained in the transition guards. In order to make this procedure more widely applicable, these restrictions must be removed. We expect that by removing these restrictions the abstract system will become non-deterministic, or alternatively a stack of timed automata indexed by groups of initial conditions that lead to different transition guards.

The IHS framework has introduced a general model to study multiagent systems with hybrid interacting dynamics, and we have addressed the problem of existence and uniqueness of its executions. It would be desirable that properties like stability or consensus were also pursued for this important class of multiagent systems.

We now outline the published and submitted papers related to this dissertation.

Publications directly related to this dissertation

Journal papers

1. “A hybrid framework for resource allocation among multiple agents moving on discrete environments”, J.L. Piovesan, C.T. Abdallah, and H.G. Tanner: In the Special issue on Collective Behavior and Control of Multi-agent Systems, Asian Journal of Control, 10(2), March 2008.

Conference papers

1. “Preliminary Results on Interconnected Hybrid Systems”, J.L. Piovesan, C.T. Abdallah, and H.G. Tanner, In the Proc. of the 16th Mediterranean Conference on Control and Automation, pp. 101 - 106, 2008.

2. “Statistical Learning for Optimal Control of Hybrid Systems”, J.L. Piovesan, C.T. Abdallah, M. Egerstedt, H.G. Tanner, and Y. Wardi; In Proc. of the 2007 American Control Conference, pp. 2775 2780, July 2007.

3. “Discrete Asymptotic Abstractions of Hybrid Systems”, J.L. Piovesan, H.G. Tanner, and C.T. Abdallah: In Proc. of the 45th IEEE Conference on Decision and Control, pp. 917 922, Dec. 2006.

4. “Modeling Multiple Interacting Hybrid Systems”, J.L. Piovesan, C.T. Abdallah, and H.G. Tanner,: Submitted, American Control Conference, 2009.

Publications that extended results from this dissertation

1. “Statistical Learning controller for the energy management in a Fuel Cell Electric Vehicle”, M. Cavalletti, J. Piovesan, C. Abdallah, S. Longhi, P. Dorato, G. Ippoliti, To appear in the 47th IEEE Conference on Decision and Control, 2008.

2. “Statistical Learning applied to the energy management in a Fuel Cell Electric Vehicle”, M. Cavalletti, J. Piovesan, C. Abdallah, S. Longhi, P. Dorato, G. Ippoliti, In the Proc. of the 17th IFAC World Congress, pp. 4659 - 4664, 2008

References

[1] T. Alpcan and T. Başar. A utility-based congestion control scheme for Internet-style networks with delay. In *Proc. of the IEEE Infocom*, volume 3, pages 2039–2048, San Francisco, CA, USA, April 2003.

[2] R. Alur, T. Henzinger, G. Lafferriere, and G. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88(7):971–984, 2000.

[3] H. Axelsson, Y. Wardi, M. Egerstedt, and E. Verriest. A provably convergent algorithm for transition-time optimization in switched systems. In *Proceedings of the IEEE Conference on Decision and Control*, pages 1387–1402, Seville, Spain, Dec. 2005.

[4] H. Axelsson, Y. Wardi, M. Egerstedt, and E. Verriest. A gradient descent approach to optimal mode scheduling in hybrid dynamical systems. *Journal of Optimization Theory and Applications*, 136(2):167–186, Feb. 2008.

[5] C. Belta and V. Kumar. Abstraction and control for groups of robots. *IEEE Trans. on Robotics*, 20(5):865–875, 2004.

[6] A. Bemporad, F. Borrelli, and M. Morari. Piecewise linear optimal controllers for hybrid systems. In *Proceedings of the American Control Conference*, pages 1190–1195, Chicago, IL, USA, June 2000.

[7] A. Bemporad, A. Giua, and C. Seatzu. Synthesis of state-feedback optimal controllers for continuous-time switched linear systems. In *Proceedings of the IEEE Conference on Decision and Control*, pages 3182–3187, Las Vegas, NV, USA, Dec. 2002.

[8] S. Bengea and R. DeCarlo. Optimal control of switching systems. *Automatica, Elsevier Science Ltd.*, 41(1):11–27, Jan. 2005.

[9] A. Bhatia and E. Frazzoli. Incremental search methods for reachability analysis of continuous and hybrid systems. In R. Alur and G. Pappas, editors, *Proc. of 7th Inter. Workshop, Hybrid Systems: Computation and Control, LNCS 2993*, pages 142–156, Berlin, Germany, 2004. Springer-Verlag.

[10] M. Boccadoro, Y. Wardi, M. Egerstedt, and E. Verriest. Optimal control of switching surfaces in hybrid dynamical systems. *Journal of Discrete Event Dynamic Systems*, 15(4):433–448, Dec. 2005.

[11] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, Cambridge, United Kingdom, 2004.

[12] M. Branicky, V. Borkar, and S. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):31–45, Jan. 1998.

[13] L. Chaimowicz, N. Michael, and V. Kumar. Controlling swarms of robots using interpolated implicit functions. In *Proceedings of the International Conference on Advanced Robotics*, pages 2498–2503, Barcelona, Spain, April 2005.

[14] J. Clarck and R. Fierro. Cooperative hybrid control of robotic sensors for perimeter detection and tracking. In *Proceedings of the American Control Conference*, pages 3500–3505, Portland, OR, USA, June 2005.

[15] C. Collaro, C. Abdalah, A. Tornambè, and U. Dole. Optimal control of hybrid systems using statistical learning. In *Proceedings of the IEEE Conference on Decision and Control*, pages 1415–1420, Seville, Spain, December 2005.

[16] W. Cook, W. Cunningham, W. Pulleyblank, and A. Schrijver. *Combinatorial Optimization*. John Wiley and Sons, 1997.

[17] J. Cortés and F. Bullo. Coordination and geometric optimization via distributed dynamical systems. *SIAM Journal on Control and Optimization*, 44(5):1543–1574, 2005.

[18] M. Egerstedt, Y. Wardi, and H. Axelsson. Transition-time optimization for switched-mode dynamical systems. *IEEE Transactions on Automatic Control*, 51(1):110–115, Jan. 2006.

[19] J. Finke, K. Passino, and A. Sparks. Stable task load balancing strategies for cooperative control of networked autonomoues vehicles. *IEEE Transactions on Control Systems Technology*, 14(5):789–803, September 2006.

[20] V. Gazi and M. Passino. Stability analysis of swarms. *IEEE Transactions on Automatic Control*, 48(4):692–697, Apr. 2003.

[21] B. Ghosh, A. Polpitiya, and W. Wang. Bio-inspired networks of visual sensors, neurons and oscillators. *Proceedings of the IEEE*, 95(1):188–214, January 2007.

[22] A. Girard, A. Julius, and G. Pappas. Approximate simulation relations for hybrid systems. *Discrete Event Dynamic Systems*, 2007. DOI 10.1007/s10626-007-0029-9.

[23] A. Girard and G. Pappas. Approximate bisimulation relations for constrained linear systems. *Automatica*, 43(8):1307–1317, 2007.

[24] C. Godsil and G. Royle. *Algebraic Graph Theory*. Springer-Verlag, New York, NY, USA, 2001.

[25] I. Grossmann. Review of nonlinear mixed-integer and disjunctive programming techniques. *Journal on Optimization and Engineering*, 3(3):227–252, Sept. 2002.

[26] S. Hedlund and A. Rantzer. Convex dynamic programming for hybrid systems. *IEEE Transactions on Automatic Control*, 47(9):1536–1540, Sept. 2002.

[27] A. Jadbabaie, J. Lin, and S. Morse. Coordination of groups of mobile autonomous agents using nearest neighbor rules. *IEEE Transactions on Automatic Control*, 48(6):988–1001, June 2003.

[28] H. Jerez, C. Abdallah, and J. Khoury. A mobile transient network architecture. Preprint available at `http://hdl.handle.net/2118/hj_tran_06`, 2006.

[29] F. Kelly. Charging and rate control for elastic traffic. *European Transactions on Telecommunications*, 8:33–37, 1997.

[30] H. Khalil. *Nonlinear Systems*. Prentice Hall, third edition, 2002.

[31] M. Khatir and E. Davison. Decentralized control of a large platoon of vehicles operating on a plane with steering dynamics. In *Proceedings of the American Control Conference*, pages 2159–2165, Portland, OR, USA, June 2005.

[32] E. Klavins. Programmable self-assembly. *Control Systems Magazine*, 24(4):43–56, Aug. 2007.

[33] M. Kloetzer and C. Belta. Temporal logic planning and control of robotic swarms by hierarchical abstractions. *IEEE Trans. on Robotics*, 23(2):320–331, 2007.

[34] V. Koltchinskii, C. Abdallah, M. Ariola, P. Dorato, and D. Panchenko. Improved sample complexity estimates for statistical learning control of uncertain systems. *IEEE Transactions on Automatic Control*, 45(12):2383–2388, December 2000.

[35] S. Liu, T. Basar, and R. Srikant. Controlling the Internet: A survey and some new results. In *Proceedings of the IEEE Conference on Decision and Control*, pages 3048–3057, Maui Hawaii, USA, December 2003.

[36] J. Lygeros. Lecture notes on hybrid systems. Notes for an ENSIETA workshop, February–June 2004.

[37] J. Lygeros, K. Johansson, S. Simić, J. Zhang, and S. Sastry. Dynamical properties of hybrid automata. *IEEE Transactions on Automatic Control*, 48(1):2–16, Jan. 2003.

[38] J. Marshall, M. Broucke, and B. Francis. Formations of vehicles in cyclic pursuit. *IEEE Transactions on Automatic Control*, 49(11):1963–1974, Nov. 2004.

[39] J. McNew, E. Klavins, and M. Egerstedt. Solving coverage problems with embedded graph grammars. In A. Bemporad, A. Bicchi, and G. Buttazzo, editors, *Hybrid Systems: Computation and Control*, pages 413–427, London, 2007. Springer-Verlag.

[40] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[41] L. Moreau. Stability of multiagent systems with time-dependent communication links. *IEEE Transactions on Automatic Control*, 50(2):169–182, Feb. 2005.

[42] P. Ogren, M. Egerstedt, and X. Hu. A control Lyapunov function approach to multiagent coordination. *IEEE Transactions on Robotics and Automation*, 18(5):847–851, Oct. 2002.

[43] S. Oh, L. Schenato, P. Chen, and S. Sastry. Tracking and coordination of multiple agents using sensor networks: system design, algorithms and experiments. *Proceedings of the IEEE*, 95(1):234–254, January 2007.

[44] R. Olfati-Saber. Consensus problems in networks of agents with switching topology and time delay systems. *IEEE Transactions on Automatic Control*, 49(9):1520–1533, Sept. 2004.

[45] R. Olfati-Saber, J. Fax, and R. Murray. Consensus and cooperation in networked multi-agent systems. *Proceedings of the IEEE*, 95(1):215–233, January 2007.

[46] A. Pant, P. Seiler, and K. Hedrick. Mesh stability of look-ahead interconnected systems. *IEEE Transactions on Automatic Control*, 47(2):403–407, Feb. 2002.

[47] G. J. Pappas. Bisimilar linear systems. *Automatica*, 39(12):2035–2047, December 2003.

[48] G. J. Pappas, G. Lafferriere, and S. Sastry. Hierarchically consistent control systems. *IEEE Transactions on Automatic Control*, 45(6):1144–1160, June 2000.

[49] G. J. Pappas and S. Simic. Consistent hierarchies of nonlinear abstractions. In *Proceedings of the 39th IEEE Conference in Decision and Control*, pages 4379–4384, Sydney, Australia, Dec. 2000.

[50] G. J. Pappas and S. Simic. Consistent abstractions of affine control systems. *IEEE Transactions on Automatic Control*, 47(5):745–756, May 2002.

[51] K. Passino and K. Burgess. *Stability Analysis of Discrete Event Systems*. John Wiley and Sons, New York, NY, USA, 1998.

[52] D. Pepyne and C. Cassandras. Optimal control of hybrid systems in manufacturing. *Proceedings of the IEEE*, 88(7):1108–1123, 2000.

[53] J. Piovesan, C. Abdallah, and H. Tanner. A hybrid framework for resource allocation among multiple agents moving on discrete environments. *Asian Journal of Control, Special Issue on Collective Behavior and Control of Multi-Agent Systems*, 10(2):171–186, March 2008.

[54] J. Piovesan, C. Abdallah, and H. Tanner. Interconnected hybrid systems: A framework for multi-agent systems with hybrid interacting dynamics. UNM Technical Report EECE-TR-08-001, Dept. Electrical and Computer Engineering, Univeristy of New Mexico, Albuquerque, NM, 87131, March 2008. `http://hdl.handle.net/1928/3655`.

[55] J. Piovesan, C. Abdallah, H. Tanner, H. Jerez, and J. Khoury. Resource allocation for multi-agent problems in the design of future communication networks. UNM Technical Report EECE-TR-07-001, Dept. Electrical and Computer Engineering, Univeristy of New Mexico, Albuquerque, NM, 87131, April 2007. `http://hdl.handle.net/1928/2973`.

[56] S. Rao. *Engineering Optimization: Theory and Practice*. John Wiley and Sons, Inc., third edition, 1996.

[57] W. Ren, R. Beard, and E. Atkins. Information consensus in multi-vehicle cooperative control. *IEEE Control Systems Magazine*, 27(2):71–82, April 2007.

[58] R. Sandoval-Rodriguez, C. Abdallah, P. Hokayem, E. Schamiloglu, and R. Byrne. Robust mobile robotic formation control using internet-like protocols. In *Proceedings of the IEEE Conference on Decision and Control*, pages 5109–5112, Maui, Hawaii, USA, Dec 2003.

[59] A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1998.

[60] M. Shaikh. *Optimal Control of Hybrid Systems: Theory and Algorithms*. PhD thesis, Department of Electrical and Computer Engineering, McGill University, Montréal, Canada, 2004.

[61] M. Shields. *An introduction to automata theory*. Blackwell Scientific Publications, 1987.

[62] R. Srikant. *The Mathematics of Internet Congestion Control*. Birkhauser, 2004.

[63] H. Sussmann. A maximum principle for hybrid optimal control problems. In *Proceedings of the IEEE Conference on Decision and Control*, pages 425–430, Phoenix, AZ, USA, Dec. 1999.

[64] S. Swaroop and J. Hedrick. String stability of interconected systems. *IEEE Transactions on Automatic Control*, 41(3):349–356, Mar. 1996.

[65] P. Tabuada. Approximate simulation relations and finite abstractions of quantized control systems. In A. Bemporad, A. Bicchi, and G. Buttazzo, editors, *Hybrid Systems: Computation and Control*, volume 4416 of *Lecture Notes in Computer Science*. Springer, April 2007.

[66] P. Tabuada. Symbolic models for control systems. *Acta Informatica*, 43:477–500, 2007.

[67] P. Tabuada and G. J. Pappas. Bisimilar control affine systems. *Systems & Control Letters*, 51(1):49–58, May 2004.

[68] P. Tabuada and G. J. Pappas. Quotients of fully nonlinear control systems. *SIAM Journal of Control and Optimization*, 43(5):1844–1866, 2005.

[69] H. Tanner, A. Jadbabaie, and G. Pappas. Flocking in fixed and switching networks. *IEEE Transactions on Automatic Control*, 52(5):863–868, May 2007.

[70] H. Tanner, G. Pappas, and V. Kumar. Leader-to-formation stability. *IEEE Transactions on Robotics and Automation*, 20(3):443–455, June 2004.

[71] H. G. Tanner and G. J. Pappas. Simulation relations for discrete-time linear systems. In *Proceedings of the 15th IFAC World Congress*, Barchelona, Spain, July 2002. Submitted.

[72] R. Tempo, G. Calafiore, and F. Dabbene. *Randomized Algorithms for Analysis and Control of Uncertain Systems*. Communications and Control Engineering. Springer, London, UK, 2003.

[73] C. Tomlin, J. Lygeros, and S. Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, 2000.

[74] A. van der Schaft. Equivalence of dynamical systems by bisimulation. *IEEE Transactions on Automatic Control*, 49(12):2160–2172, Dec. 2004.

[75] M. Vidyasagar. *A Theory of Learning and Generalization*. Communications and Control Engineering. Springer, London, UK, 1997.

[76] M. Vidyasagar. Randomized algorithms for robust controller synthesis using statistical learning theory. *Automatica, Elsevier Science Ltd.*, 37:1515–1528, 2001.

[77] J. Wen and M. Arcak. A unifying passivity framework for network flow control. In *Proc. of the IEEE Infocom*, volume 2, pages 1156–1166, San Francisco, CA, USA, April 2003.

[78] L. Xiao, M. Johansson, and S. Boyd. Simultaneous routing and resource allocation via dual decomposition. *IEEE Trans. on Communications*, 52(7):1136–1144, July 2004.

[79] X. Xu and P. Antsaklis. Optimal control of switched systems based on parameterization of the switching instants. *IEEE Transactions on Automatic Control*, 49(1):2–16, Jan. 2004.

[80] B. Young, R. Beard, and J. Kelsey. A control scheme for improving multi-vehicle formation maneuvers. In *Proceedings of the American Control Conference*, volume 2, pages 704–709, Arlington, VA, USA, June 2001.