Control and Stabilization of Discrete Event Systems with Limited Look-Ahead Policies

A Thesis

Presented in Partial Fulfillment of the Requirements for

the Degree Master of Science in the

Graduate School of The Ohio State University

By

Veysel Gazi, B.S.E.E.

* * * * *

The Ohio State University

2001

Master's Examination Committee:

Professor Kevin M. Passino, Adviser

Professor Hitay Özbay

Approved by

Adviser

Electrical Engineering

ABSTRACT

In this thesis, stability of a class of discrete event systems (DES) is studied using the Lyapunov stability theoretic approach. The DES is modeled by an automata model with (possibly) an infinite number of states. It is assumed that more than one event can occur simultaneously. In fact, it is assumed that an enabled uncontrollable event can occur together with an enabled controllable event. First, stabilizability of invariant sets is analyzed and sufficient conditions for stabilizability of them are derived. Next, optimality in control of DES and optimal stabilization of invariant sets are considered and conditions for optimal stabilizability are presented. After that, the idea of limited look-ahead policies (LLP) for control of DES is employed and a result for the special case for the cost function in which disabeling events is free is presented. The general problem is not analyzed because of the complexity. Finally, two illustrative examples are presented.

Control and Stabilization of Discrete Event Systems with Limited Look-Ahead Policies

By

Veysel Gazi, M.S.

The Ohio State University, 2001

Professor Kevin M. Passino, Adviser

In this thesis, stability of a class of discrete event systems (DES) is studied using the Lyapunov stability theoretic approach. The DES is modeled by an automata model with (possibly) an infinite number of states. It is assumed that more than one event can occur simultaneously. In fact, it is assumed that an enabled uncontrollable event can occur together with an enabled controllable event. First, stabilizability of invariant sets is analyzed and sufficient conditions for stabilizability of them are derived. Next, optimality in control of DES and optimal stabilization of invariant sets are considered and conditions for optimal stabilizability are presented. After that, the idea of limited look-ahead policies (LLP) for control of DES is employed and a result for the special case for the cost function in which disabeling events is free is presented. The general problem is not analyzed because of the complexity. Finally, two illustrative examples are presented.

Çok sevdiğim annem ve babama ...

To my parents for their sacrifices and support ...

ACKNOWLEDGMENTS

I would like to thank my advisor, Dr. Kevin Passino, for the patience, insight and guidance he offered me throughout the last two years. This thesis would not be possible without his experience, control and patience especially in correcting my scientific and stylistic errors. I would also like to thank Dr. Hitay Özbay for being in my examination committee.

I'm grateful to The Scientific and Research Council of Türkiye for giving me opportunity to study in the United States by financially supporting first nine months of my study here.

My parents, Hafize and Hüseyin, deserve most of the credit for this work. I would like to thank them for their unconditional love, endless support and sacrifices at all stages of my life.

I thank my sister, Gülşen, and my brother, Yüksel, and their families, for never letting me down and being always next to me when I need them.

Finally, I want to thank Manfredi Maggiore, Murat Zeren, Raúl Ordóñez, Mehmet Akar, and also other members of the Controls Group for their valuable discussions and suggestions. I appreciate very much the brotherhood and friendship that I enjoyed in our group during the last two years.

VITA

July 29, 1973	Born - Kircali, Bulgaria
June, 1996	B.S. in Electrical and Electronics Engineering Middle East Technical University, Ankara, Türkiye
September 1996 - April 1997	Scholar of The Scientific and Research Council of Türkiye The Ohio State University, Columbus, Ohio
May 1997 - Present	Graduate Research Associate The Ohio State University, Columbus, Ohio

FIELDS OF STUDY

Major Field: Electrical Engineering

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure **Page**

CHAPTER 1

Introduction

Discrete event systems (DES) are dynamic systems which evolve by occurrence of events at possible irregular intervals of time. In past two decades such systems have received a great attention in the control theory literature. This is mainly because many large scale dynamic systems have discrete event system behavior at least at some level of description. Examples for such systems are manufacturing systems, communication networks, and expert systems.

Consider, for example, a machine that operates on some items or parts on a manufacturing line. Assume that there is an input buffer where parts are stored before processing by the machine and an output buffer for storing the processed parts. In this system the possible events are "arrival of a part to be processed by the machine," "machine begins operating on a part," "machine finishes processing a part and passes it to the output buffer," "occurring a failure in the machine," "repairment of the machine," etc.

Depending on the events that have occurred, the current state of the machine could be "idle," "busy," "broken," "in maintenance," etc. Similarly, one can describe the state of the buffers with the number of parts stored in them. In general, not all the events can occur at every state. For instance, the event "machine begins operating

on a part" is not possible if the input buffer is empty. Similarly, if the current state of the machine is "idle," the event "machine finishes processing a part and passes it to the output buffer" is not possible.

Some of the events can be controllable and some of them uncontrollable. For example, if the machine is idle and the input buffer is not empty, we can force it to begin processing a part by an external control and hence this event is controllable. On the other hand, by no means we can prevent the occurrence of a failure in the machine which is an example of an uncontrollable event. Note that some of the events can occur together. For instance, it can happen that both machine begins processing an item and an item arrives to the input buffer.

The above example is basic example which illustrates how DES operate. Now we provide a brief overview of the current literature on DES.

1.1 Literature Overview

Early work on the control of discrete event systems is given by Ramadge and Wonham [8, 9, 10]. In the framework of these researchers, the control of a discrete event system is achieved by enabling and disabling of certain events. The desired behavior of the DES is specified as a set of acceptable strings of events (a language). Therefore, this approach is called *linguistic approach* since the closed-loop behavior of the system is specified in terms of a language.

This work gained considerable attention by other researchers who extended it by applying the concepts of the control theory such as stability and stabilization. Formal definitions of stability and stabilization, which are close to resiliency and error recovery concepts in computer science, were given by Özveren et al. [4].

In the Ramadge and Wonham framework the goal is to restrict the system such that all event strings generated from given initial state are in a given set of "legal" or accepted strings whereas in the work by Özveren et al. the emphasis was put on the states. The state-space is divided to a sets of "good" and "bad" states and the objective is to operate the system within the good states. If the system switches to bad state due to occurrence of an uncontrollable event then the objective is to generate a legal behavior which guarantees that the system will return to a normal operating condition, i.e., to the good states, after execution of a finite number of events. Therefore, this approach is called a *state space approach*.

Consider the manufacturing line example described before and recall that it is always subject to a failure. In the linguistic framework, one includes in the desired languages the set of event strings with all possible failures and the successful recoveries, whereas in the state space framework the state space is divided to sets of failure (bad) states and normal operation (good) states and a control strategy for transition from failure to normal states is sought.

The first definition and analysis of stability and stabilization of invariant sets of DES in the control theoretic framework can be found in [11] where authors analyze both uncontrolled and controlled DES and also provide polynomial time algorithms for verifying the types of stability or attraction defined.

The definition of *stability of discrete event systems in the sense of lyapunov* was first proposed by Passino et al. [2]. The authors analyze a class of DES in the Lyapunov framework and provide sufficient conditions for stability of an invariant set of the system. One can view an invariant set as a set of good states of the DES.

In their work [7] Cho and Lim combine the stability definitions by Özveren et al. and Passino et al. by defining stability in the sense lyapunov with resiliency. A potential function, which serves as Lyapunov function, is defined and a potential to every state in the state space is assigned. They provide algorithms for verifying stability and obtaining the domain of attraction. Moreover, the issue of robust stability and stabilizability is addressed and analyzed. For robust stability analysis the plant is assumed unknown but within given set of plants and the stability of common invariant set is studied.

The notion of robust and adaptive control was studied also by Lin in [13]. In this paper he studies the stability of a system with model uncertainty. He proposes two approaches for controlling such a system. First approach is robust control approach, in which the plant is controlled without resolving uncertainties. That is, the supervisor is designed so that the closed-loop system will be stable for all possible plants. In the second approach, the adaptive scheme is considered. As time evolves and as more information is gathered, some of the plants in the set can be thrown away and the supervisor can be updated to get better performance. The necessary and sufficient conditions for existence of a robustly stabilizing supervisor are derived and the performance of the robust and adaptive supervisor are compared and the adaptive supervisor is found to be better.

Garg and Kumar analyzed DES's with infinite states in [14]. For analysis they use *assignment-based programs* and *predicates* for describing the desired behavior of the plant. First they show that, in general, the problem with infinite states is undecidable and later they show that if it is possible to describe the DES with assignment-based

programs then the control problem is decidable and that the problem is reduced to solving arithmetic equations.

In [3] the authors analyze the problem of optimal control of DES. They specify allowable event trajectories contained within the valid event trajectories and try to find controller which will lead to optimal allowable event trajectories. Their solution to this problem is based on a heuristic search to find the optimal controller. They define a "heuristic function" to focus the search and thus to overcome the computational complexity.

Other papers which analyze the optimal control and stabilization are [12, 15, 16]. In [12] the authors build on the ideas on their previous paper on stabilization [11] and find conditions for the existence of controllers leading to optimal attraction in the sense that the cost of the path is minimized. In [15] the authors solve the problem of optimal control in the Ramadge and Wonham framework using network flow techniques. Reference [16] analyzes a similar problem as in [15] but in more general sense and provides conditions for existence of an optimal supervisor or controller.

Chung and Lafortune first applied the idea of "look-ahead" to DES [5, 6]. In [5] they study the supervisory control problem in the framework of Ramadge and Wonham using limited look-ahead policies. They adapt two attitudes, called *optimistic* and *conservative* for calculating the next control action at each step. They present some convergence properties of their scheme in terms of the look-ahead window size N, a closed-form expression for the language generated by the controlled DES and lower bounds on N which guarantee that the limited look-ahead policy will perform as well as the controller designed off-line. In [6] they show how the computations

required by their on-line scheme can be done recursively which considerably reduces the amount of computation time.

In their paper Keerthi and Gilbert [17] analyze stability of general class of non-linear time-varying discrete time systems using optimal infinite-horizon and moving-horizon (limited look-ahead) feedback control laws. They provide constraints on the system and the cost function which are sufficient for asymptotic stability of the feedback system both under optimal infinite-horizon and moving-horizon feedback control laws. They also show that the moving-horizon feedback law approaches the infinite-horizon one as the horizon is extended.

1.2 Thesis Outline

In this work we basically follow the approach in [1, 2] for analysis of DES which is a state space approach in the framework of Lyapunov and is based on a metric space. We consider (possibly) infinite set of states and a finite number of controllable and uncontrollable events and try to develop a controller based on limited look-ahead that will stabilize a desired invariant set of the system. Our limited look-ahead policy approach to control of DES is different from the approach of Chung and Lafortune [5, 6] and it is closer to the approach of Keerthi and Gilbert [17] to control general nonlinear discrete time systems. The basic difference in our model is that we assume that we have complete control of the controllable events and that if required we can fire (i.e. force to occur) them whenever they are defined. The event to be fired is chosen after doing an optimization on the possible paths within the look-ahead window. For this reason, the paths generated by the controlled DES are in a sense optimal (or suboptimal). Moreover, we allow occurrence of more than one event

simultaneously. In fact, we assume that the uncontrollable events can occur together with the controllable ones whenever they are defined in the current state of the system and enabled by the disturbance entering the system.

In Chapter 2 we establish a framework for the analysis of DES. First, we present the model of the system that we use. Next, we provide some mathematical definitions, and stability definitions and theorems. After that, we describe the closed-loop system and introduce the control problem. Chapter 3 presents stability, stabilizability and optimal stabilizability analysis for DES in our framework. Furthermore, the performance of limited look-ahead policies for our model is investigated. Chapter 4 presents two examples which illustrate the theory developed in Chapter 3 and finally in Chapter 5 we present our conclusions.

CHAPTER 2

A Framework for Analysis of DES

2.1 Finite Automata Model for DES

In this work we first consider DES [1] modeled by

$$G = (\mathcal{X}, \mathcal{E}, f_e, \delta_e, g, \mathbf{E}_v) \tag{2.1}$$

where

- $\mathcal{X}$ is the set of plant states and $x_k \in \mathcal{X}$ denotes the state at "logical time" $k \in \mathcal{N}$, where $\mathcal{N}$ is the set of natural numbers.
- $\mathcal{E} = \mathcal{E}_u \cup \mathcal{E}_d \cup \mathcal{E}_o$ is the set of events where
  - $\mathcal{E}_u$ is the (possibly infinite) set of command input or controllable events denoted with e_{u_k},
  - $\mathcal{E}_d$ is the (possibly infinite) set of disturbance input or uncontrollable events denoted with e_{d_k},
  - $\mathcal{E}_o$ is the set of output events (these are the events that we can observe).

  We assume that the sets $\mathcal{E}_u$ and $\mathcal{E}_d$ are disjoint, i.e., $\mathcal{E}_u \cap \mathcal{E}_d = \emptyset$, where $\emptyset$ is the empty set.

- $f_e : \mathcal{X} \times \mathcal{E}_u \times \mathcal{E}_d \to \mathcal{X}$ is the state transition function such that

$$x_{k+1} = f_{e_k}(x_k) \tag{2.2}$$

where occurrence of events $e_k \in \mathcal{E}_u \times \mathcal{E}_d$ at time k forces the transition of the system state form x_k to x_{k+1}.

- $\delta_e : \mathcal{X} \times \mathcal{E}_u \times \mathcal{E}_d \to \mathcal{E}_o$ is the output function such that

$$y_k = \delta_{e_k}(x_k) \tag{2.3}$$

where $e_k \in \mathcal{E}_u \times \mathcal{E}_d$ and $y_k \in \mathcal{E}_o$.

- $g : \mathcal{X} \to P(\mathcal{E}_u \times \mathcal{E}_d) - \emptyset$, where $P(\mathcal{E}_u \times \mathcal{E}_d)$ denotes the power set of $\mathcal{E}_u \times \mathcal{E}_d$, is the enable function. An event can occur at state $x \in \mathcal{X}$ only if it is in the set $g(x)$.

- $\mathbf{E}_v$ is the set of all valid (i.e. physically possible) infinite length event trajectories.

We define a *state trajectory* as any sequence $\{x_k\} \in \mathcal{X}^{\mathcal{N}}$ (the set of state sequences) such that $x_{k+1} = f_{e_k}(x_k)$ for some $e_k \in g(x_k)$ and for all $k \in \mathcal{N}$. An *event trajectory* is a sequence $\{e_k\} \in \mathcal{E}^{\mathcal{N}}$ such that there exist a corresponding state trajectory $\{x_k\}$ where for every $k \in \mathcal{N}$, $e_k \in g(x_k)$.

The set of *all* event trajectories is denoted by $\mathbf{E} \subset \mathcal{E}^{\mathcal{N}}$. As we stated above, we denote with $\mathbf{E}_v \subset \mathbf{E}$ the set of *valid* or physically possible event trajectories which are specified as a part of the modeling process. $\mathbf{E}_v(x_0)$ is used to denote the set of valid event trajectories starting from initial state x_0 and $\mathbf{E}_v(\mathcal{X}_z)$ these starting from the set of states $\mathcal{X}_z$, i.e., $\mathbf{E}_v(\mathcal{X}_z) = \bigcup_{x \in \mathcal{X}_z} \mathbf{E}_v(x)$. The set of allowable event trajectories are denoted with $\mathbf{E}_a \subset \mathbf{E}_v$ and $\mathbf{E}_a(x_0)$ is used for denoting those starting at x_0.

Concatenation of two event sequences E_k and E is denoted by $E_kE \in \mathbf{E}_v(x_0)$, where E_k represents the sequence of events $e_0, e_1, e_2, ..., e_k$, which occurred up to and including time k and E is an event sequence of infinite length.

For a given initial state x_0, occurrence of event trajectory E_k such that $E_kE \in \mathbf{E}_v(x_0)$ leads the system to a new state x_{k+1}. This is called a *motion* of the system and we denote it with $f_{E_k}(x_0)$. Here, we extended the state transition function such that $f_{E_k}(x_0) = f_{e_k}(f_{E_{k-1}}(x_0))$ where $E_k = E_{k-1}e_k$.

We denote the set of *all* state trajectories or motions of the system with $\mathbf{X} \subset \mathcal{X}^{\mathcal{N}}$. Moreover, in a way similar to event trajectories $\mathbf{X}_v \subset \mathbf{X}$ denotes the set of valid state trajectories, $\mathbf{X}_v(x_0)$ these starting at initial state x_0 and $\mathbf{X}_v(\mathcal{X}_z)$ those starting at a set $\mathcal{X}_z$.

A state trajectory $x_0, x_1, ..., x_k$ is called a *cycle* if $x_0 = x_k$ and there exist no $i, j \in 0, ..., k-1$ and $i \neq j$ such that $x_i = x_j$. In other words, no other state appears twice within the cycle.

The set of all states reachable from a state x is called *reach* of x and is denoted with $R(x)$. That is $\bar{x} \in R(x)$ is equivalent to saying that there exists $E \in \mathbf{E}_v(x)$ such that $\bar{x} = f_E(x)$. We denote with $R_m(x)$, $m \in \mathcal{N}$ the set of states that can be reached from the state x within m transitions. Therefore, $R_1(x)$ denotes all possible $f_e(x)$ for all $e \in g(x)$ and $R_2(x) = R_1(x) \cup \{R_1(\bar{x}) \text{ for all } \bar{x} \in R_1(x)\}$. Note that, $R_1(x)$ is a set of states because in general there are more than one events defined at a given state.

A DES is said to be in a *deadlock* or *blocked* if it reaches a state $x \in \mathcal{X}$ for which $|g(x)| = 0$ where $|\cdot|$ gives the cardinality or the number of elements of a set. In other words, the system is in deadlock if there is no valid transition to any other state. This

notion can be extended to liveness; a state is called *alive* if it has defined transitions to other states and all the states in its *reach* are alive.

2.2 The Closed-Loop System

In this section we introduce the controller to the system and close the loop. Figure 2.1 shows the closed-loop system with the DES plant G and the controller C. The output of the controller, denoted with $u_k \subset \mathcal{E}_u$ is a set of command input events that are enabled by the controller at time k. Therefore, the controllable events which are not in u_k cannot occur. In other words, at time k and state x_k a controllable event e_{u_k} can occur if and only if $e_{u_k} \in (u_k \cap g(x_k))$.

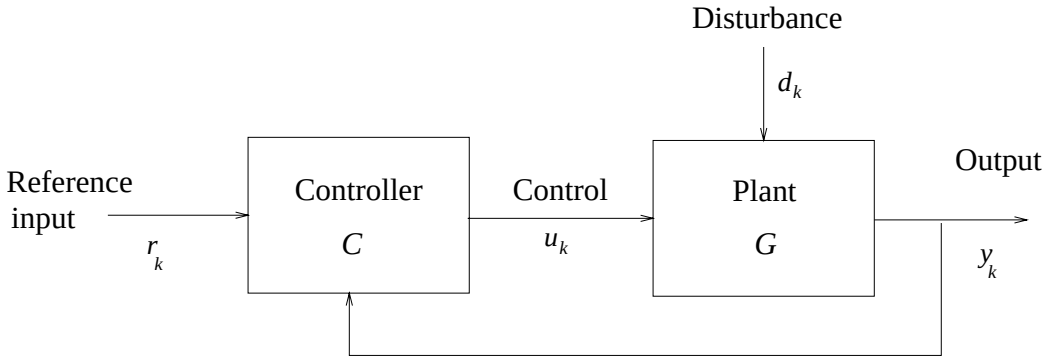

Figure 2.1: The closed-loop system.

Similarly, we have disturbance input $d_k \subset \mathcal{E}_d$ to the system which denotes the set of disturbance events that can occur. Therefore, analogously to the control input case, at time k and state x_k an uncontrollable event e_{d_k} can occur if and only if $e_{d_k} \in (d_k \cap g(x_k))$.

The reference input to the controller is denoted with r_k and no assumptions are made for the type of the input. In other words it can be a set of target states for the DES G or a particular event pattern or event trajectories (called *language* by some researchers) that we want the DES G to generate or follow.

We denote with y_k the output of the DES G and assume that it is used by the controller to generate the next control input u_{k+1}. It can be the case that y_k represents a set of events or perhaps states depending on the system at hand. We will specify explicitly r_k and y_k in the subsequent sections.

At any time k and state x_k there can be three different types of input events that can occur in the system. Let $e_{u_k} \in (u_k \cap g(x_k))$ and $e_{d_k} \in (d_k \cap g(x_k))$. Then the possible types of events are

- $e_k^1 = \{e_{u_k}, e_{d_k}\}$, i.e., one controllable and one uncontrollable event occur simultaneously,
- $e_k^2 = \{e_{u_k}\}$, or only the command input event occurred,
- $e_k^3 = \{e_{d_k}\}$, or only the disturbance input event occurred.

We assume that no two (or more) command (or disturbance) input events can occur simultaneously.

Let $g(x) = g_u(x) \cup g_d(x)$ where $g_u(x) \cap g_d(x) = \emptyset$ and $g_u(x)$ denotes the set of command input events and $g_d(x)$ denotes the set of disturbance input events that can occur at state x. Then the control input event e_{u_k} can occur if $e_{u_k} \in (u_k \cap g_u(x_k))$ and the disturbance input event e_{d_k} can occur if $e_{d_k} \in (d_k \cap g_d(x_k))$. Note that, by choosing an appropriate control u_k we can vary the set of e_{u_k} that can occur from $\emptyset$ to $g_u(x_k)$ but we do not have any control over the set of e_{d_k}. In other words, the

set of possible e_{d_k} can vary from $\emptyset$ to $g_d(x_k)$ at each state x_k depending on the input disturbance d_k which is not under our control.

Define $n_u(x) = |g_u(x)|$ and $n_d(x) = |g_d(x)|$ as the number of the controllable and uncontrollable events, respectively, in the enable function at state x. Then the maximum physically possible number of input events that can occur at state x is $n_u(x)n_d(x) + n_u(x) + n_d(x)$ (see the three different types of events defined above). By choosing an appropriate u_k, the maximum number of possible events under control can be varied between $n_d(x)$ and $n_u(x)n_d(x) + n_u(x) + n_d(x)$. Note that this is the maximum number and the actual number varies depending on d_k.

Now assume that the controller is a policy $\pi = \{u_0, u_1, ..., u_{k-1}\}$. Because of the presence of the controller, the possible event trajectories decrease in number. We denote with $\mathbf{E}_v(x_0, \pi) \subset \mathbf{E}_v(x_0)$ and $\mathbf{X}_v(x_0, \pi) \subset \mathbf{X}_v(x_0)$ the sets of valid event trajectories and state trajectories, respectively, under control policy π starting at state x_0.

Similarly, under the control policy π the reach of each state in the system is in general narrowed. We denote with $R(x_0, \pi) \subset R(x_0)$ the reach of state x_0 under control policy π and with $R_N(x_0, \pi)$ the set of the states that can be reached in N steps.

2.3 Mathematical Preliminaries

Definition 1 *We say that $\{\mathcal{X}; \rho\}$ is a metric space if there is a function, called the metric, defined as*

$$\rho : \mathcal{X} \times \mathcal{X} \to R^+, \tag{2.4}$$

(where R^+ denotes nonnegative reals) which satisfies the following conditions

1. $\rho(x, y) = 0$ *if* $x = y$

2. $\rho(x, y) = \rho(y, x)$ *for all* $x, y \in \mathcal{X}$ *and*

3. $\rho(x, z) \leq \rho(x, y) + \rho(y, z)$ *for all* $x, y, z \in \mathcal{X}$.

In the light of this definition, we define distance between a point $x \in \mathcal{X}$ and a set $\mathcal{X}_z \subset \mathcal{X}$ to be

$$\rho(x, \mathcal{X}_z) = \inf\{\rho(x, y) : y \in \mathcal{X}_z\} \tag{2.5}$$

Let $B(\mathcal{X}_z; r)$ denote the r-neighborhood (i.e. a ball with radius r) of the set $\mathcal{X}_z$ and be defined as

$$B(\mathcal{X}_z; r) = \{x \in \mathcal{X} : 0 < \rho(x, \mathcal{X}_z) < r\}. \tag{2.6}$$

Definition 2 *A function*

$$\rho : \mathcal{X} \times \mathcal{X} \to R^+, \tag{2.7}$$

is called a pseudometric if it satisfies only second and third condition in Definition 1 above.

From now on we will denote pseudometrics with $\rho_{ps}(x, y)$.

Kampke or "class K" functions are very useful in explaining Lyapunov stability theory. We introduce them next.

Definition 3 *A continuous function* $\psi : [0, r] \to R^+$ *is said to belong to class K (i.e.* $\psi \in K$*) if*

1. $\psi(0) = 0$

2. *it is strictly increasing on* $[0, r]$

Moreover, if $\psi : R^+ \to R^+$ *and* $\lim_{r\to\infty} = \infty$ *then* ψ *is said to belong to class* K_∞ *(i.e.,* $\psi \in K_\infty$*).*

In control theory generally the analysis of a system is done with respect to equilibria. The stability of an equilibrium point or more generally an invariant set, is of paramount importance. The definition of an invariant set is as follows.

Definition 4 *A set* $\mathcal{X}_m \subset \mathcal{X}$ *is called invariant set with respect to G in (2.1) if for all* $x_0 \in \mathcal{X}_m$*, all* E_k *such that* $E_k E \in \mathbf{E}_v(x_0)$ *and* $k \geq 0$ *we have* $f_{E_k}(x_0) \in \mathcal{X}_m$*.*

This definition simply says that if a set is invariant then all the motions originating within it remain in it.

For some systems, such as DES, some sets of states can be made invariant sets of the system under proper control. We call such states *potential invariant sets* and define them as follows.

Definition 5 *A set* $\mathcal{X}_m \subset \mathcal{X}$ *is called a potential invariant set with respect to G in (2.1) if there exists a control policy* π*, such that for all* $x \in \mathcal{X}_m$*, all* E_k *such that* $E_k E \in \mathbf{E}_v(x_0, \pi)$ *and* $k \geq 0$ *we have* $f_{E_k}(x_0) \in \mathcal{X}_m$*.*

Note that for every $x \in \mathcal{X}_m$, $R(x) \subset \mathcal{X}_m$ and this is possible if all the paths within the set $\mathcal{X}_m$ are cycles or there are states which are blocked, i.e. $|g(x)| = 0$.

2.4 Stability Definitions and Theorems

In this section we present some stability definitions and theorems, most of which are taken from [1]. First, we define the notion of (asymptotically) stable (potential) invariant sets.

Definition 6 *A closed (potential) invariant set $\mathcal{X}_m \subset \mathcal{X}$ of G is called stable in the sense of Lyapunov with respect to the set of event trajectories $\mathbf{E}_a \subset \mathbf{E}_v$ if for any $\epsilon > 0$ there exists some $\delta > 0$ such that whenever $\rho(x_0, \mathcal{X}_m) < \delta$, we have $\rho(f_{E_k}(x_0), \mathcal{X}_m) < \epsilon$ for all E_k such that $E_k E \in \mathbf{E}_a(x_0)$ and $k \geq 0$. Moreover, if $\rho(f_{E_k}(x_0), \mathcal{X}_m) \to 0$ as $k \to \infty$, then the closed (potential) invariant set $\mathcal{X}_m$ of G is called asymptotically stable with respect to $\mathbf{E}_a$.*

In the analyses in this work we will mainly deal with the asymptotically stable invariant sets. Note that the above definition is a local property, therefore it is sufficient that the required properties hold in a neighborhood of the invariant set. For this reason we may have states $x \in \mathcal{X}$ for which the properties in the definition does not hold. This leads to the following definition which classifies the set of states with respect to (w.r.t.) which the invariant set is asymptotically stable. In other words, all trajectories starting from this set converge to the invariant set.

Definition 7 *If the closed (potential) invariant set $\mathcal{X}_m \subset \mathcal{X}$ of G is asymptotically stable with respect to $\mathbf{E}_a$, then the set $\mathcal{X}_a \subset \mathcal{X}$ having the property that for all $x_0 \in \mathcal{X}_a$, $\rho(f_{E_k}(x_0), \mathcal{X}_m) \to 0$ for all E_k such that $E_k E \in \mathbf{E}_a(x_0)$ as $k \to \infty$ is called the region of asymptotic stability of $\mathcal{X}_m$ with respect to $\mathbf{E}_a$. If $\mathcal{X}_a = \mathcal{X}$, then the closed (potential) invariant set $\mathcal{X}_m$ of G is called asymptotically stable in the large w.r.t. $\mathbf{E}_a$ or globally asymptotically stable.*

In the following we will sometimes use the term x_0-stable for an invariant set $\mathcal{X}_m$ if $x_0 \in \mathcal{X}_a$ holds.

Now we provide two theorems which give sufficient conditions for stability in the sense of Lyapunov and asymptotic stability of invariant sets. The proofs are omitted here, however, since one can find them in [1].

Theorem 1 *In order for the (potential) invariant set $\mathcal{X}_m$ to be stable in the sense of Lyapunov with respect to $\mathbf{E}_a$ it is sufficient that in a neighborhood $B(\mathcal{X}_m; r)$ there exists a specified functional V and functions $\psi_1, \psi_2 \in K$ such that:*

- $\psi_1(\rho(x, \mathcal{X}_m)) \leq V(x) \leq \psi_2(\rho(x, \mathcal{X}_m))$, *and*
- $V(f_{E_k}(x_0))$ *is a nonincreasing function for all $x_0 \in B(\mathcal{X}_m; r)$, for all E_k such that $E_k E \in \mathbf{E}_a(x_0)$ and all $k \geq 0$.*

Theorem 2 *In order for the (potential) invariant set $\mathcal{X}_m$ to be asymptotically stable with respect to $\mathbf{E}_a$ it is sufficient that in a neighborhood $B(\mathcal{X}_m; r)$ there exists a specified functional V and functions $\psi_1, \psi_2 \in K$ that satisfies the properties of Theorem 1 and there exists a function $\psi_3 \in K$ such that $V(f_{E_{k+1}}(x_0)) - V(f_{E_k}(x_0)) \leq -\psi_3(\rho(f_{E_k}(x_0), \mathcal{X}_m))$ for all $x_0 \in B(\mathcal{X}_m; r)$ and for all E_k such that $E_{k+1} = E_k e$ ($e \in \mathcal{E}$) and $E_{k+1} E \in \mathbf{E}_a(x_0)$ and all $k \geq 0$.*

In this work we are mainly dealing with the control and stabilization of DES. In other words, we are trying to find a controller which will guarantee the asymptotic stability of an (potential) invariant set. For some DES it can be the case that there does not exists such a controller. For this reason we introduce the notion of stabilizability and asymptotic stabilizable region of a potential invariant set $\mathcal{X}_m$.

Definition 8 *A closed (potential) invariant set $\mathcal{X}_m \subset \mathcal{X}$ of G is called asymptotically stabilizable in the sense of Lyapunov if there exists a controller such that $\mathcal{X}_m$*

is asymptotically stable with respect to the set of valid event trajectories of the controlled system. Moreover, the set of states $\mathcal{X}_{a/c} \subset \mathcal{X}$ *such that for every* $x_0 \in \mathcal{X}_{a/c}$ $\rho(f_{E_k}(x_0), \mathcal{X}_m) \to 0$ *under control is called asymptotically stabilizable region of* $\mathcal{X}_m$*. If* $\mathcal{X}_{a/c} = \mathcal{X}$*, then* $\mathcal{X}_m$ *of* G *is called globally asymptotically stabilizable.*

Note that the region of asymptotic stability of a potential invariant set $\mathcal{X}_m$ is a subset of its asymptotically stabilizable region, i.e., $\mathcal{X}_a \subset \mathcal{X}_{a/c}$.

Consider a state $x_0 \in \mathcal{X}_{a/c}$ of $\mathcal{X}_m$. Then there exists a policy π such that $\mathcal{X}_m$ is asymptotically stable w.r.t. $\mathbf{E}_v(x_0, \pi)$. In that case, we will say that $\mathcal{X}_m$ is x_0-stabilizable. Therefore, $\mathcal{X}_m$ is x_0-stabilizable for all $x_0 \in \mathcal{X}_{a/c}$. Note also that, $R(x_0, \pi) \subset \mathcal{X}_{a/c}$ for any stabilizing π.

In this and preceding sections of this chapter we established the required background for the analysis of DES. In the subsequent chapter we state the control problem which is the main motivation for this work and try to solve it using planning strategies.

CHAPTER 3

Stabilizability, Optimality and Planning in DES

In the preceding chapter we established some tools and definitions for analysis of DES. In this chapter we first discuss some stabilizability and optimal stabilizability of invariant sets in DES. Then we introduce the idea of planning [5, 6] and try to verify the need for it in DES with large or infinite number of states or in time varying DES where it is difficult, if not impossible, to design a controller off-line.

3.1 Stabilizability of DES

We have defined an invariant set of a DES as asymptotically stabilizable if there exists a controller which makes it an asymptotically stable invariant set. In this section we try to investigate the conditions under which such a controller exists. Before proceeding, recall that we had three types of events associated with a pair of command and disturbance input events. In other words, for $e_{u_k} \in u_k \cap g_u(x_k)$ and $e_{d_k} \in d_k \cap g_d(x_k)$ we can have $e_k^1 = \{e_{u_k}, e_{d_k}\}$, $e_k^2 = \{e_{u_k}\}$ and $e_k^3 = \{e_{d_k}\}$. Now we provide the following definition which will be useful in the subsequent development.

Definition 9 *A subset* $\mathbf{E}_r(x_0) \subset \mathbf{E}_v(x_0)$ *is called controllable if for all* k, E_k, *and* x_{k+1} *such that* $E_k E \in \mathbf{E}_r(x_0)$ *and* $x_{k+1} = f_{E_k}(x_0)$ *we have* $E_k e_{k+1}^3 E \in \mathbf{E}_r(x_0)$ *for all*

$e_{d_{k+1}} \in g_d(x_{k+1})$ *and if* $E_k e^2_{k+1} E \in \mathbf{E}_r(x_0)$ *for some* $e_{u_{k+1}} \in g_u(x_{k+1})$ *or* $E_k e^1_{k+1} E \in \mathbf{E}_r(x_0)$ *for some* $e_{u_{k+1}} \in g_u(x_{k+1})$ *and* $e_{d_{k+1}} \in g_d(x_{k+1})$ *then* $E_k e^1_{k+1} E \in \mathbf{E}_r(x_0)$ *for this* $e_{u_{k+1}}$ *and all* $e_{d_{k+1}} \in g_d(x_{k+1})$.

This definition simply says that no occurrence of disturbance event alone leads out of this set of paths and if there is a path continuing with occurrence of a command input event in the set (which means that the command input event is enabled) then all the couples of events of this controllable event with the uncontrollable events should be within the set. (In other words since we cannot disable the disturbance events we should include them in the set of possible paths.) This notion is very similar to the notion of *controllable language* [8, 9, 10], however it is more general because we allow for simultaneous occurrence of controllable and uncontrollable events. A controllable language is a special case of our definition when simultaneous occurrence of events is not allowed, i.e. all events of type e^1_k are illegal and therefore excluded from above definition.

Now we state the following result as a lemma as it follows immediately from the above definition.

Lemma 1 *If* $\mathbf{E}_r(x_0) \subset \mathbf{E}_v(x_0)$ *is controllable then there exists a control policy* π *such that* $\mathbf{E}_v(x_0, \pi) \subseteq \mathbf{E}_r(x_0)$.

This result is very simple and intuitive. It states that if the controller enables only command input events which lead to the paths within the required set of paths then the paths generated by the controlled system will be a subset of the controllable set of event trajectories. Note also that, the controllability condition ensures that no uncontrollable event trajectory leading out of the set is left.

It is worth nothing also that, in $\mathbf{E}_v(x_0, \pi) \subseteq \mathbf{E}_r(x_0)$ the equality condition holds whenever $d_k = g_d(x_k)$ for all x_k.

Now we are ready to state the first result on the stabilizability of an invariant set $\mathcal{X}_m \subset \mathcal{X}$.

Proposition 1 *An invariant set $\mathcal{X}_m \subset \mathcal{X}$ is asymptotically stabilizable in the sense of Lyapunov w.r.t. $\mathbf{E}_a$ if for all x_0 in a sufficiently small neighborhood $B(\mathcal{X}_m; r)$ we have a controllable $\mathbf{E}_r(x_0) \subseteq \mathbf{E}_v(x_0)$ such that $\mathcal{X}_m$ is asymptotically stable w.r.t. $\mathbf{E}_r(x_0)$.*

Proof: Define $\mathbf{E}_a = \bigcup_{x_0} \mathbf{E}_r(x_0)$ for all $x_0 \in B(\mathcal{X}_m; r)$. Individual controllability of each $\mathbf{E}_r(x_0)$ implies the controllability of their union, i.e., of $\mathbf{E}_a$. Moreover, since $\mathcal{X}_m$ is stable w.r.t. each $\mathbf{E}_r(x_0)$ then it is stable w.r.t. $\mathbf{E}_a$ also. By Lemma 1 there exists a control policy π such that $\mathbf{E}_v(B(\mathcal{X}_m; r), \pi) \subseteq \mathbf{E}_a$ and this completes the proof. ■

According to this proposition, from every $x \in B(\mathcal{X}_m; r)$ we should have a controllable set of trajectories, all of which lead to the set $\mathcal{X}_m$. Since the system trajectories can be constrained by an appropriate control to follow the set of controllable trajectories, this is sufficient for stabilizability.

Proposition 1 provides sufficient conditions for stabilizability of an invariant set $\mathcal{X}_m$ of G. Therefore, one can try to find a stabilizing controllable set of event trajectories to verify the stabilizability. However, this may be difficult and time consuming. For this reason, it would be better to have some necessary conditions which are easier to check before trying to find stabilizing $\mathbf{E}_r(x_0)$. We provide such conditions in Lemma 2 but before that we define the uncontrollable subsystem of G as

$$G^d = (\mathcal{X}, \mathcal{E}, f_{ed}, \delta_{ed}, g_d, \mathbf{E}_{vd}) \tag{3.1}$$

which is obtained from G by disabling all controllable events. Note that $\mathbf{E}_{vd} \subset \mathbf{E}_r$ for every controllable $\mathbf{E}_r \subset \mathbf{E}_v$. Let $\mathbf{X}_{vd} \subset \mathbf{X}_v$ be the set of state paths that can be generated by G^d. We will call every $X \in \mathbf{X}_{vd}$ an *uncontrollable path* and if this path is a cycle we will call it an *uncontrollable cycle*. Now, we state the lemma.

Lemma 2 *For an invariant set $\mathcal{X}_m$ of G to be asymptotically stabilizable in the sense of Lyapunov it is necessary that there exists an r-neighborhood $B(\mathcal{X}_m; r)$, in which the uncontrollable subsystem of G, G^d, has no cycles.*

Proof: Assume for a sake of contradiction that G^d has cycles in every r-neighborhood $B(\mathcal{X}_m; r)$ and also that $\mathcal{X}_m$ of G is asymptotically stabilizable. Let $X = \{x_0, x_1, x_2, ..., x_l\}$ be a cycle in G^d such that $x_l = x_0$ and $E_{l-1}E \in \mathbf{E}_{vd}$ be the corresponding event trajectory, i.e., $x_l = f_{E_{l-1}}(x_0)$. Since all events in this trajectory are uncontrollable $E_{l-1}E \in \mathbf{E}_r \subset \mathbf{E}_v$ for any controllable $\mathbf{E}_r$. Therefore, no control can guarantee x-stability of $\mathcal{X}_m$ for all $x \in X$ contradicting the assumption of stabilizability of $\mathcal{X}_m$. ■

This lemma shows that the behavior of the uncontrollable system G^d is important and one can predict non-stabilizability by looking at its behavior. To check this condition can be easy or difficult depending on the DES G at hand. However, since the total number of transitions is less in G^d than in G it is easier to look at the behavior of uncontrollable system than looking at the behavior of G. Note that asymptotic stability of $\mathcal{X}_m$ of G^d implies asymptotic stabilizability of $\mathcal{X}_m$ of G. We state this as a corollary.

Corollary 1 *If a potential invariant set $\mathcal{X}_m$ of G is asymptotically stable w.r.t. $\mathbf{E}_{vd}$ then it is asymptotically stabilizable.*

Proof: Choose a control policy $\pi = \{u_k\}$ such that $u_k = \emptyset$ for all k. ■

Note that asymptotic stability of $\mathcal{X}_m$ w.r.t. $\mathbf{E}_{vd}$ is a sufficient condition but not necessary. This is because there can be states $\bar{x}$ which are blocked in G^d, i.e., $g_d(\bar{x}) = \emptyset$, but still $\mathcal{X}_m$ be $\bar{x}$-stabilizable.

It is worthwhile to mention here that not only the states $\bar{x}$ lying on cycles in G^d satisfy $\bar{x} \notin \mathcal{X}_{a/c}$ of $\mathcal{X}_m$ but also these $\bar{x}$ which lie on paths in G^d leading to cycles.

It is interesting to note here the similarities between G^d and the notion of "zero-dynamics" in conventional control theory. For output stabilizability, in the conventional feedback linearizing control theory we convert the system to the "normal form." For example, the affine system

$$\dot{x} = \alpha(x) + \beta(x)u$$

in normal form becomes

$$\begin{aligned} \dot{x}_1 &= f_1(x_1, x_2) \\ \dot{x}_2 &= Ax_2 + B\beta^{-1}(x)[u - \alpha(x)] \end{aligned}$$

where A and B are matrices with appropriate dimensions. We need the zero dynamics of this system, i.e.

$$\dot{x}_1 = f_1(x_1, 0),$$

to be stable in order the system to be output stabilizable. Discussion above shows that similar results arise here also in the sense that we need disturbance dynamics to be acyclic and stable as a necessary condition for stabilizability.

3.2 Optimality and Optimal Stabilization of DES

In this section we discuss optimality issues and optimal stabilization in DES. First, we develop the notion of optimal control of DES.

3.2.1 Optimality in DES

Optimal control is a well developed field [18, 19, 20]. The main idea is that we have some performance index or cost function subject to some predefined constraints which we try to minimize by choosing an appropriate control. The idea in DES is similar as will be seen below.

To begin with, we assume that disabling any controllable event requires some use of system resources and therefore has some cost which we will denote with $h_u(e_u)$. Similarly, there is a cost associated with being in a given state which we denote with $h_x(x)$.

Mathematically the control and state cost functions are

$$h_u : \mathcal{E}_u \rightarrow R^+ \tag{3.2}$$

and

$$h_x : \mathcal{X} \rightarrow R^+ \tag{3.3}$$

where R^+ denotes the nonnegative reals. The state cost function is defined such that

$$h_x(x) = \begin{cases} 0 & \text{if} \quad x \in \mathcal{X}_m \\ \text{finite} & \text{if} \quad x \in \mathcal{X}_{a/c} \text{ of } \mathcal{X}_m \end{cases} \tag{3.4}$$

and the control cost function is such that $0 \leq h_u(e_{u_k}) < B_u < \infty$ for some constant B_u and for all $e_{u_k} \in \mathcal{E}_u$ or disabling any controllable event has a finite cost. This cost function makes sense since we want to reach the set $\mathcal{X}_m$ the cost of whose states is zero. To prevent instability the policies leading to states lying on cycles or blocked states are discarded.

Two important questions here are "How to define $h_x(x)$?" and "Is it possible to define it using an analytic closed-form expression?" We will address those issues in the subsequent sections.

Using the above cost functions the cost of using control u_k at a state x_k is given by

$$h(x_k, u_k) = h_x(x_k) + \sum_{e_{u_k} \in (g_u(x_k) - u_k)} h_u(e_{u_k}) \tag{3.5}$$

where e_{u_k} is a disabled controllable event. This equation states that being at a state x_k costs $h_x(x_k)$ and disabling a controllable event e_{u_k} at that state adds a cost given by $h_u(e_{u_k})$ to the total cost. Note that increasing the number of disabled events increases the total control cost, however, it generally decreases the number of possible future states.

Now, consider a control policy $\pi = \{u_0, u_1, ..., u_k, ...\}$ and a state trajectory $X = \{x_0, x_1, ..., x_k, ...\}$ generated due to π, i.e., $X \in \mathbf{X}(x_0, \pi)$. The total cost of this state trajectory is given by

$$J(X, \pi) = \sum_{k=0}^{\infty} h(x_k, u_k). \tag{3.6}$$

Then, the cost of the control policy π is defined as the maximum cost among all possible state trajectories. In other words,

$$J(\pi) = \max_{X \in \mathbf{X}(x_0, \pi)} \{J(X, \pi)\}. \tag{3.7}$$

It is clear that different control policies may have different costs. A control policy with the minimum cost will be called optimal and will be denoted with π^*. In other words,

$$J^* = J(\pi^*) = \min_{\pi} \{J(\pi)\}. \tag{3.8}$$

Note that the optimal control policy may be nonunique because we may have policies with the same cost.

This development is very similar to the ideas in classical optimal control theory. Here, again we have the tradeoff between the cost of control and the cost of state which brings the need for optimization.

The above problem is called a *min-max* problem since we first have a maximization w.r.t. the possible state trajectories and then a minimization w.r.t. the control policies. *Dynamic programming* [18, 19] is one of the procedures which can be used to solve this optimization problem.

Note that when the invariant set $\mathcal{X}_m$ is reached, one can set $u = g_u(x)$ for all $x \in \mathcal{X}_m$ and then the cost is $h(x, g_u(x)) = 0$. This is important observation which implies that an optimal policy shall eventually drive the system within the invariant set $\mathcal{X}_m$ since otherwise its cost would be infinite and therefore the policy wouldn't be optimal.

Note also that, we do not have negative cost. Negative cost can be viewed as some gain or advantage obtained by occurrence of an event or reaching a particular state. This is a topic of further research.

3.2.2 Optimal Stabilization

In the previous subsection we established the basic ideas of optimal control of DES. In this subsection, we first ask the question "Is it possible to stabilize the system and minimize the cost or performance index simultaneously?" and then try to answer it.

Now, assume that the initial state x_0 is known and recall that in order for $\mathcal{X}_m$ to be x_0-stabilizable, it is sufficient to have controllable $\mathbf{E}_r(x_0) \subseteq \mathbf{E}_v(x_0)$ such that $\mathcal{X}_m$ is asymptotically stabilizable w.r.t. $\mathbf{E}_r(x_0)$. Let π be a policy such that $\mathbf{E}_v(x_0, \pi) \subseteq \mathbf{E}_r(x_0)$. Then $\mathcal{X}_m$ is x-stabilizable for all $x \in R(x_0, \pi)$ since otherwise x_0 would not

be asymptotically stabilizable. Recall also that we have infinite cost for the states which are not in $\mathcal{X}_{a/c}$ of $\mathcal{X}_m$ and for a policy π to be optimal we need $J(\pi) < \infty$. This leads to the following lemma.

Lemma 3 *The cost of every stabilizing policy π is $J(\pi) < \infty$.*

Proof: This is proved by a contradiction. Assume that there is a stabilizing control policy π such that $J(\pi) = \infty$. Note from Equation 3.4 that for $x \in \mathcal{X}_m$ we have $h_x(x) = 0$ and recall that once $\mathcal{X}_m$ is reached one can set the control $u = g_u(x)$ so that $h(x, g_u(x)) = 0$ for all $x \in \mathcal{X}_m$. Since the cost is infinite there is an infinite path $X \in \mathbf{X}_v(x_0, \pi)$ which does not reach $\mathcal{X}_m$ contradicting the assumption that π is stabilizing.

■

Now we establish the following proposition.

Proposition 2 *The potential invariant set $\mathcal{X}_m$ is optimally x_0-stabilizable iff it is x_0-stabilizable.*

Proof: *(if)* Assume that $\mathcal{X}_m$ is x_0-stabilizable. Then there exists π such that $J(\pi) < \infty$. Let Π_s be the set of all stabilizing π, then $\pi^* = \arg\min_{\pi \in \Pi_s}\{J(\pi)\}$ is the optimally stabilizing policy.

(only if) Assume optimally stabilizable, then it is stabilizable.

■

This proposition is important because it states that it is enough to perform minimization only among the stabilizing control policies. Therefore, the job of finding an optimal control policy is divided to two parts, i.e., to find all stabilizing policies and then the policy with the lowest cost among the stabilizing policies.

These ideas are similar to the ideas of robust and optimal control theory, where you first find the parameterization of all stabilizing controllers and then the one which minimizes the system norm.

In this section we investigated the issues of optimality and optimal stabilization in DES. In the next section the notion of suboptimality and limited look-ahead policies (LLP) will be discussed.

3.3 Limited Look-ahead Policies for Control of DES

In the preceding sections we investigated stabilization, optimality, and optimal stabilization of the class of DES defined. In this section we introduce the idea of planning implemented via limited look-ahead policy for optimal (or suboptimal) stabilization.

As in the previous section, an invariant set $\mathcal{X}_m$ of G is to be optimally asymptotically stabilized. One possibility is to design the controller off-line, i.e., to find a closed-form expression for the optimal controller and then apply it. The other possibility is to find the optimal control on-line. The first approach has a drawback such that for highly nonlinear systems such as DES it can be difficult if not impossible to find the closed-form expression for the optimal controller. Moreover, if the state space is infinite then the calculations required may be undesirably long and again it can be impossible to find the optimal solution. Furthermore, if the system is time varying or there is an uncertainty in the system model then the policy found may no longer be optimal or even stabilizing. For these reasons, we use the second approach with LLP controller for optimal stabilization of the invariant set.

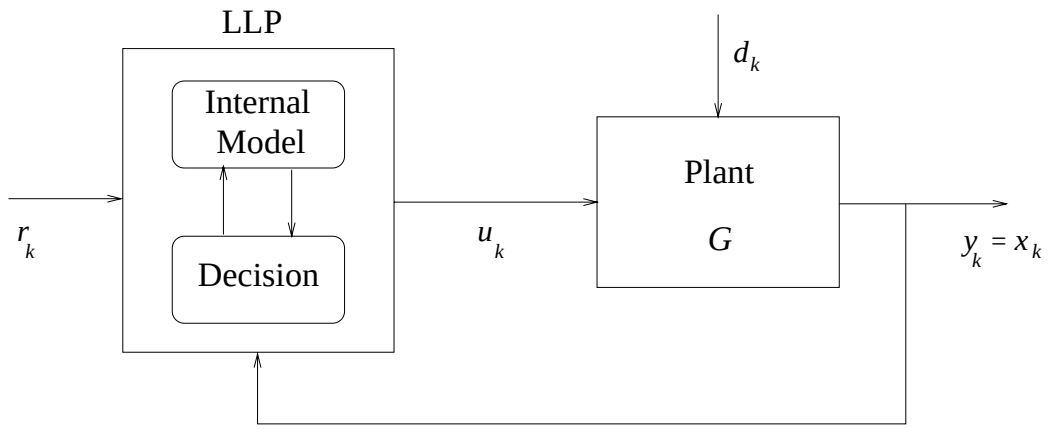

Figure 3.1: The closed-loop system with LLP.

Figure 3.1 shows the closed-loop system with this type of controller. The input to the controller is the invariant set which is to be stabilized, i.e., $r_k = \mathcal{X}_m$. For now, it is assumed that the output of the plant is the state of the system, $y_k = x_k$, and so full state feedback is assumed (hence we do not have to concern ourselves with the issue of observability). Here we assume that the state-space $\mathcal{X}$ is a finite dimensional since otherwise $y_k = x_k$ would be very restrictive assumption. This, however, does not contradict the assumption that we can have infinite number of states. The controller has a model of the DES G (denoted by "Internal Model") and at each state x_k it generates the possible future states of the system N steps ahead as an N stage "state tree." In other words, it generates $R_N(x_k)$, or the first N steps of all $X \in \mathbf{X}_v(x_k)$ as shown in Figure 3.2. The box labeled as "Decision" within the controller evaluates the paths in the tree, finds the controllable subtrees and the corresponding policies, the optimal policy, and executes the first control action in the optimal policy. After the occurrence of an event and transition to a new state x_{k+1} the procedure is repeated.

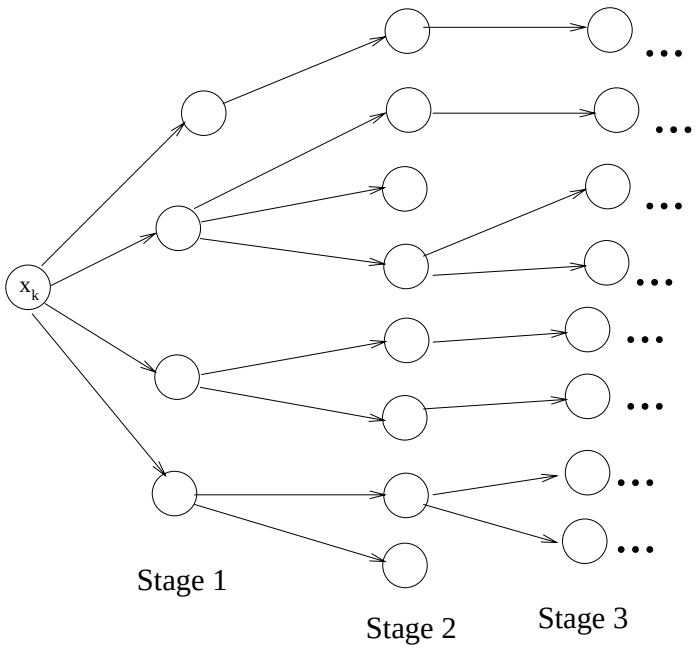

Figure 3.2: A state tree.

Depending on the size of the look-ahead, N, the amount of computations will be tremendously reduced by this algorithm. Moreover, since the control action is calculated on-line based on the current state of the system this algorithm has the potential to perform much better in time varying or uncertain systems. However, since N is finite there may be cases when the control policy generated by the algorithm is no longer an optimal one or it may even lead to instability of the invariant set. Therefore, we are looking for a lower bounds on N, say N_{l_1} and N_{l_2}, such that for all $N > N_{l_1}$ we will guarantee asymptotic stability and for all $N > N_{l_2}$ we will ensure that the policy generated by the LLP will be close to the optimal policy π^*. In other words, the infinite horizon cost of the trajectories generated by the LLP will be equal or almost equal to the cost of the trajectories due to the optimal policy.

Now, we mathematically formulate the job of the "Decision" box in the controller. Let $X_N X \in \mathbf{X}_v(x_k, \pi_N)$ be a valid state trajectory due to the control policy π_N and starting at state x_k, where $X_N = x_k, x_{k+1}, ..., x_{k+N}$ represents first $N+1$ states within the trajectory including x_k. The limited look-ahead cost of this state trajectory is defined as

$$J_N(X_N, \pi_N) = \sum_{i=k}^{k+N-1} h(x_i, u_i) + h_x(x_{k+N}) \tag{3.9}$$

where $h(x, u)$ is the function defined in the previous sections. Analogous to the previous section, the cost of the policy is then defined as

$$J_N(\pi_N) = \max_{X_N X \in \mathbf{X}(x_k, \pi_N)} \{J_N(X_N, \pi_N)\}. \tag{3.10}$$

Once again the objective is to find π_N^* such that

$$J_N^* = J_N(\pi_N^*) = \min_{\pi_N} \{J_N(\pi_N)\} \tag{3.11}$$

and to implement the first entry of this policy.

Note here that, since the maximization is performed over a finite horizon, the only way a policy to have an infinite cost is to lead to a state $\bar{x} \notin \mathcal{X}_{a/c}$. Now the question is; "How long should the look-ahead window be so that we avoid states $\bar{x} \notin \mathcal{X}_{a/c}$?"

Consider again the uncontrollable subsystem of G, G^d, and the set of valid event trajectories of it $\mathbf{E}_{vd}$. Let l_c^d denote the length of the maximum uncontrollable event trajectory $E \in \mathbf{E}_{vd}$ leading to an uncontrollable cycle. Note that if a transition to any of the states on such paths occurs then no control can guarantee convergence to the invariant set $\mathcal{X}_m$. Keeping this observation in mind, note that in G there may be other cycles which can be broken by an appropriate control. We will call those

cycles *controllable cycles* and will denote with l_c^u the length of the one with maximum length. Let l_b^d be the length of the maximum uncontrollable event trajectory leading to a blocked state in G. Consider a state at which there are no disturbance events defined but all the controllable paths lead to a blocked state, to an uncontrollable cycle or to an uncontrollable path to an uncontrollable cycle. Then to avoid transition to one of these all the events should be disabled which on the other hand leads to blocking. We call such trajectories *unavoidable paths* and denote with l_d^u the length of such event trajectory with maximum length. Note that all $\bar{x}$ lying on such trajectories satisfy $\bar{x} \notin \mathcal{X}_{a/c}$. We will denote with $l_c^d(x_0)$, $l_c^u(x_0)$, $l_b^d(x_0)$ and $l_d^u(x_0)$ the maximum length of the above described trajectories in $R(x_0)$.

Using the above definitions now we return to our LLP scheme.

Algorithm 1 *Let the initial state of the DES G be $x_0 \in \mathcal{X}_{a/c}$. Choose*

$$N(x_0) \geq \max\{l_c^d(x_0) + l_d^u(x_0) + 1, l_b^d(x_0) + l_d^u(x_0) + 1, l_c^u(x_0)\}. \tag{3.12}$$

On every iteration of the LLP do

1. *Generate the $N(x_k)$ level state tree.*

2. *Discard from consideration the control policies leading to blocked states, states in cycles and states on uncontrollable or unavoidable paths to cycles or blocked states. For the rest of the states assign $h_x(x) = \psi(\rho(x, \mathcal{X}_m))$ where $\psi \in K$ is a predefined function.*

3. *Perform the optimization among the policies left and apply the first entry of the optimal policy.*

One can see immediately that the $h_x(\cdot)$ function satisfies its definition on the previous section. Now, what remains is to show that this algorithm will lead to convergence to $\mathcal{X}_m$.

Lemma 4 *Let the initial state of the DES G be $x_0 \in \mathcal{X}_{a/c}$. Under the above proposed LLP Algorithm the system will never reach a state $\bar{x}$ such that $\bar{x} \notin \mathcal{X}_{a/c}$. Moreover, all the controllable cycles will be avoided.*

Proof: Obvious since the look-ahead window size N is big enough to detect them and therefore to avoid them.

■

This lemma states that with the algorithm defined, the system state will never reach an unstabilizable state, however, still we do not guarantee convergence to the required invariant set. It can be the case that under this algorithm the state of the system diverges from $\mathcal{X}_m$ following stabilizable states or cycles trough the stabilizable states.

Unfortunately, providing convergence for the general case is not an easy task. However, we will prove convergence for a special case where $h_u(e_u) = 0$ for all $e_u \in \mathcal{E}_u$.

Proposition 3 *Let the initial state of the DES G be $x_0 \in \mathcal{X}_{a/c}$. Assume that $h_u(e_u) = 0$ for all $e_u \in \mathcal{E}_u$. Then there exists $\psi \in K$ for the cost function such that the LLP algorithm described above will guarantee x_0-stability of $\mathcal{X}_m$.*

Proof: Since $x_0 \in \mathcal{X}_{a/c}$ there exists a policy π such that $\mathcal{X}_m$ is asymptotically stable w.r.t. $\mathbf{E}_v(x_0, \pi)$. This implies that there exist $\psi_1, \psi_2, \psi_3 \in K$ and a function $V(x)$ satisfying conditions of Theorem 1 and Theorem 2. Choose the cost function

$\psi(x) = \psi_1(x)$. Let at each step k, π_N be the truncated sub-policy of π. Then the cost of the trajectory $X_N \in \mathbf{X}_v(x_k, \pi_N)$ is $J_N(X_N, \pi_N) = \sum_{i=k}^{k+N} \psi_1(\rho(x_i, \mathcal{X}_m))$. Note that for all $X_N \in \mathbf{X}_v(x_k, \pi_N)$, $\psi(\rho(x_i, \mathcal{X}_m))$ is decreasing because of the conditions of Theorem 1 and Theorem 2 and the fact that π is stabilizing. In particular, since $V(x_{k+1}) - V(x_{k+1}) \leq -\psi_3(\rho(x_k, \mathcal{X}_m))$ and $\psi_1(\rho(x_k, \mathcal{X}_m)) \leq V(x_{k+1})$, the cost of each succeeding state in the trajectory is less then the cost of the preceding state. For the non-stabilizing policies, however, the above two conditions do not necessarily hold. Let Π_s be the set of stabilizing policies and Π_{ns} be the set of non-stabilizing ones. Then, we have $J(\pi_N^s) < J(\pi_N^{ns})$ for all $\pi_N^s \in \Pi_s$ and all $\pi_N^{ns} \in \Pi_{ns}$. By Lemma 4 all the cycles and unstabilizable states are avoided. Therefore, the policy chosen will always be the stabilizing policy with the minimum cost and the invariant set will be asymptotically stable.

■

Note that the fact that the limited look-ahead policy of the above algorithm is stabilizing does not imply that it is the optimal one since the look-ahead window is finite. To guarantee optimality we have to impose further constraints on the system and the cost function and these are topics for further research. Moreover, the proof for the case with a general cost function is a topic for future work.

In this chapter we presented some stabilizability, optimality and limited look-ahead policies for stabilization of DES. In the next chapter we will present two illustrative examples.

CHAPTER 4

Examples

In this chapter we present two examples to illustrate the theory developed in the preceding chapters. Both of the problems are taken from [1], however, here we use an LLP for solving the control problem. The first example is a level control of a surge tank in which we have controllable valves that fill the tank and an uncontrollable empty valve. This example illustrates the importance of look-ahead in systems with disturbance events. In particular, in this problem a set of states is made an asymptotically stable invariant set despite the disturbance present in the system. The second example is a load balancing problem in a flexible manufacturing systems (FMS) in which we have 6 machines which operate on parts and a robot which moves parts from one machine to other in order to obtain a balanced load. The importance of this example is that it shows that even in systems which there are no disturbances there can be cycles outside the invariant set and these cycles can be foreseen by look-ahead and therefore broken so that stability can be attained.

4.1 LLP for a Surge Tank

Consider the surge tank example in [1]. The surge tank has two fill valves labeled as "A" and "B" as shown in Figure 4.1. An empty valve labeled "C" is located at

the bottom of the tank. It is assumed that we have complete control of the opening of both the valves A and B. The liquid level in the tank is denoted by a nonnegative integer and when valve A is opened it automatically closes itself after pouring enough liquid to increase the level in the tank by one unit. Similarly, when opened, valve B shuts itself down after increasing the level of the liquid by three levels. It is required that only one fill valve is opened at once due to resource limitations. The opening of valve C is random and unpredictable; however, it is assumed that for all times there is another time that valve C will open (i.e., C will be persistently opened). Once opened, valve C empties enough liquid so that the level in the tank decreases by 2 units (if the level was greater than or equal to 2) or until empty (if the level was less than 2), and then automatically closes itself. The liquid level is measured by a sensor in an asynchronous fashion (i.e. not according to a fixed clock) and the data is provided to the LLP controller. The control objective is to control the level of the liquid in the tank between 2 (the minimum safety level since we always want some reserves for the down-stream process) and 5 for any given initial level.

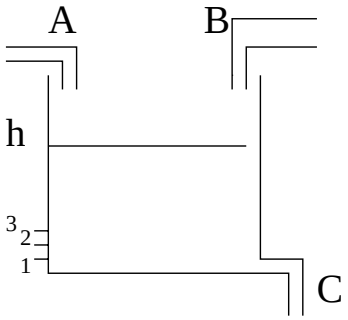

Figure 4.1: Surge tank.

The plant characteristics can be modeled by Equation 2.1 with

- The set of plant states given by $\mathcal{X} = \{0, 1, 2, 3, ...\}$.
- The set of controllable or command input events events is $\mathcal{E}_u = \{e_u^a, e_u^b\}$ where e_u^a represents the event that valve A is open and e_u^b represents the event that valve B is open.
- The set of disturbance events is $\mathcal{E}_c = \{e_d^c\}$ where e_d^c is the event of valve C is open.
- The set of output events is $\mathcal{E}_o = \mathcal{X}$, i.e., we have full state feedback.
- The output function is given by $\delta_e(x) = x$ for any valid input event e.
- The enable functions are defined as $g_u(x) = \{e_u^a, e_u^b\}$ and $g_d(x) = \{e_d^c\}$ for all $x \in \mathcal{X}$.

Using these one can see that at each time instant k and state x_k we can have the following input events to the the system.

- $e^{11} = \{e_u^a, e_d^c\}$, i.e., valve A and C are open simultaneously,
- $e^{12} = \{e_u^b, e_d^c\}$, i.e., valve B and C are open simultaneously,
- $e^{21} = \{e_u^a\}$, i.e., only valve A is open,
- $e^{22} = \{e_u^b\}$, i.e., only valve B is open,
- $e^{31} = \{e_d^c\}$, i.e., only valve C is open.

The state transition functions for these events in particular states are given as

- $f_{e^{11}}(x) = \begin{cases} x - 1 & x \geq 1 \\ 0 & x < 1 \end{cases}$

- $f_{e^{12}}(x) = x + 1$
- $f_{e^{21}}(x) = x + 1$
- $f_{e^{22}}(x) = x + 3$
- $f_{e^{31}}(x) = \begin{cases} x - 2 & x \geq 2 \\ 0 & x < 2 \end{cases}$

We define a metric $\rho : \mathcal{X} \times \mathcal{X} \rightarrow R^+$ on $\mathcal{X}$ as

$$\rho(x_1, x_2) = \mid x_1 - x_2 \mid \tag{4.1}$$

where $\mid \cdot \mid$ denotes the absolute value.

Since we want the level of the liquid in the tank to satisfy $2 \leq x \leq 5$ we choose $\mathcal{X}_m = \{2, 3, 4, 5\}$. Note that $\mathcal{X}_m$ is not a potential invariant set of the system, because for some $x \in \mathcal{X}_m$ occurrence of the disturbance event e_d^c will move the state of the system out of $\mathcal{X}_m$. Moreover, $\mathcal{X}_m$ is not asymptotically stabilizable because for some states $\bar{x} \in \mathcal{X}$ it is not possible to find a set of controllable event trajectories $\mathbf{E}_r(\bar{x}) \subset \mathbf{E}_v(\bar{x})$ w.r.t. which $\mathcal{X}_m$ is asymptotically stable. Therefore, we assume that when there is an enabled controllable event at state $\bar{x}$, the uncontrollable event e_d^c cannot occur alone. In other words, the event e^{31} can occur at state $\bar{x}$ if and only if the control output at that state is an empty set, i.e., $u(\bar{x}) = \emptyset$. This assumption fits the physical dynamics of the system and significantly simplifies the problem. Moreover, it models so called *forced event* DES. Under this assumption the set $\mathcal{X}_m$ is a potential invariant set of G since for every $x \in \mathcal{X}_m$ there exists a $e_u \in g_u(x)$ such that all corresponding valid transitions $f_e(x) \in \mathcal{X}_m$. For example, for a state $x = 2$, enabling the controllable event e_u^b will lead to state $x = 3 \in \mathcal{X}_m$ if e_d^c occurs and to the state $x = 5 \in \mathcal{X}_m$ if no disturbance event occurs. One can check similarly for the

other states in $\mathcal{X}_m$. Furthermore, it is easy to see that under this assumption $\mathcal{X}_m$ is globally asymptotically stabilizable since for every $x \in \mathcal{X}$ there exists a controllable $\mathbf{E}_r(x) \subset \mathbf{E}_v(x)$ such that $\mathcal{X}_m$ is asymptotically stable w.r.t. $\mathbf{E}_r(x)$. For example, for $x = 6$ choose $u(6) = \emptyset$ then $\mathbf{E}_r(6) = \{e_d^c E\}$ where E is an event trajectory of infinite length. Note that we do not have to wait an infinite amount of time for e_d^c to happen since we assume that it will be persistently opened.

We use a limited look-ahead of 1 step for this example, since it is enough for stabilization of $\mathcal{X}_m$, and we define the state cost function as $h_x(x) = \alpha\rho(x, \mathcal{X}_m)$ where α is a constant. The control cost function is defined as $h_u(e_u^a) = 1$ and $h_u(e_u^b) = 3$. In other words, the cost of being in a particular state is proportional to the distance of this state form the invariant set $\mathcal{X}_m$, and the cost of disabling a particular event is equal to the amount of liquid this event pours in the tank.

Choose $\alpha = 4$ so that the cost being even 1 unit away from the invariant set $\mathcal{X}_m$ is greater then the cost of disabling any controllable event. This choice is reasonable because the primary aim is stabilization of $\mathcal{X}_m$.

Now, assume that the current state of the system is $x_k = 0$. Possible control policies are $\pi^0 = \emptyset$, $\pi^1 = \{e_u^a\}$, $\pi^2 = \{e_u^b\}$ and $\pi^3 = \{e_u^a, e_u^b\}$. The policy π^0 leads to $x_{k+1} = 0$, π^1 leads to $x_{k+1} \in \{0, 1\}$, π^2 ends up in $x_{k+1} \in \{1, 3\}$ and the next possible states for π^3 are $x_{k+1} \in \{0, 1, 3\}$.

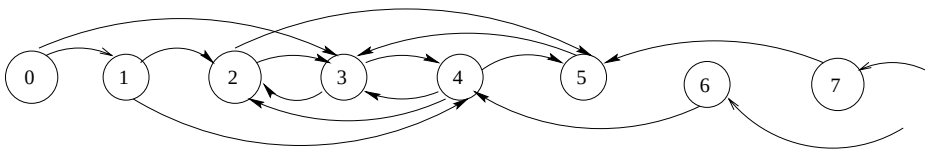

Figure 4.2: State transition diagram.

The cost of policy π^0 can be calculated as $J(\pi^0) = h_x(0)+h_u(e_u^a)+h_u(e_u^b)+h_x(0) = 8+1+3+8 = 20$. Similarly, one can calculate the cost of π^1 as $J(\pi^1) = \max\{h_x(0)+ h_u(e_u^b) + h_x(0), h_x(0) + h_u(e_u^b) + h_x(1)\} = 21$ and for π^2 and π^3 as $J(\pi^2) = 13$ and $J(\pi^3) = 16$. The policy π^2 has the minimum cost hence it is the optimal policy. For this reason at state $x_k = 0$ the control input to be applied by the LLP to the DES G is $u_k = \{e_u^b\}$. By a similar analysis for the other states one can easily obtain the state transition diagram of the system under LLP as shown in Figure 4.2. Note that there is no path leading out of $\mathcal{X}_m$ and all other paths are towards the invariant set chosen. This clearly illustrates that the controller performs well and that the system would be asymptotically stable.

To state it mathematically we can choose Lyapunov function as $V(x) = \rho(x, \mathcal{X}_m)$ which automatically satisfies the conditions of Theorem 1. From Figure 4.2 we can see that along all possible state trajectories of the system under one step LLP $V(x_{k+1}) < V(x_k)$. These results hold not for only a neighborhood of $\mathcal{X}_m$ but for entire $\mathcal{X}$. Therefore, the invariant set $\mathcal{X}_m$ is *asymptotically stable in the large*.

In the above example we have perfect state information, i.e., our system is completely observable. Moreover, the model of the system perfectly reflects the system behavior. Examples where these characteristics do not hold can become much more challenging.

4.2 LLP for a Flexible Manufacturing System

Consider the FMS given in Figure 4.3. It is composed of 6 machines which are connected with a robotic transporter. Each machine has a queue for holding the parts that it will operate on. The job of the robotic transporter is to balance the load of all

the machines. We assume that it is possible for the robot transporter to move parts along the arrows shown in Figure 4.3. It is assumed that the robot is able to sense the levels in all the buffers of all the machines. The objective is to design an LLP for movement of parts by the robot that will obtain a balanced load.

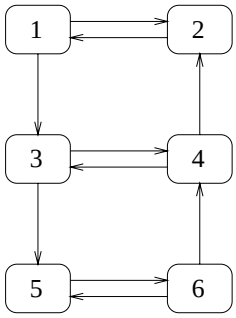

Figure 4.3: Six machine FMS.

This system can be represented with equation 2.1 where the set of states $\mathcal{X} = R^6$ since we have 6 machines. The set of command input events is

$$\mathcal{E}_u = \{e_u^{12}, e_u^{13}, e_u^{21}, e_u^{34}, e_u^{35}, e_u^{42}, e_u^{43}, e_u^{56}, e_u^{64}, e_u^{65}\}$$

where the event e_u^{ij} represents moving a part from the machine i to the machine j. For this system we do not have any disturbance events, hence $\mathcal{E}_d = \emptyset$. Under these conditions all of the paths of the system are controllable. Moreover, since we do not have blocked states, any set $\mathcal{X}_m \subset \mathcal{X}$ of the system is globally asymptotically stabilizable.

Let $\mathcal{X}_b \subset \mathcal{X}$ denotes the set of balanced states for a given initial state. (Note that $\mathcal{X}_b$ depends on the initial state of the system.) From the above discussion it follows that it is globally asymptotically stabilizable. However, there are also paths which

may increase the unbalance in the state of the system, and moreover, some paths form cycles. For this reason, LLP is very suitable for control of this system since by looking ahead we can see the paths leading to unbalance or cycles and by choosing an appropriate control avoid them.

The state of the system at time k is denoted with $x(k)$ and the state of a machine i at time k is represented with $x_i(k)$. Assume that

$$\sum_{i=1}^{6} x_i(0) = 6N_d + N_r \tag{4.2}$$

for some $N_d, N_r \in \mathcal{N}$ such that $0 \leq N_r \leq 5$. Then the set of balanced states is given by

$$\mathcal{X}_b = \{x_i = N_d + 1 \text{ for } N_r \text{ machines and } x_j = N_d \text{ for the rest of the machines}\}. \tag{4.3}$$

Let $x^*(k) = \max_{1 \leq i \leq 6} x_i(k)$ and $x_*(k) = \min_{1 \leq i \leq 6} x_i(k)$ denote the maximum and minimum number of parts among the machines at time k.

We use the metric defined by

$$\rho(x, y) = \sum_{i=1}^{6} |x_i - y_i| \tag{4.4}$$

for analysis of this system.

To begin with, first examine the paths in the system. Consider moving a part from a machine to all the other machines, for example, from machine 1 to all other machines. By examining Figure 4.3 one can see that to move a part from 1 to 2 or 3 takes one move, to move a part from 1 to 4 or 5 takes 2 moves, and to get a part to 6 from 1 takes 3 moves at minimum. Examining the other paths in the system one finds that the longest path between two different machines in the system, which does

not pass trough any machine twice, is 4 movements and it corresponds to moving a part either from 2 to 6 or from 5 to 1. Therefore, we can move a part from a machine to any other machine by firing of at most 4 transitions.

This observation is important because it suggests that the look ahead window size should be $N \geq 4$ so that the controller considers all the paths from each machine to another machine.

Now assume that the state of the system is $x(k)$ and that $g_u(x(k)) = \mathcal{E}_u$. Therefore, we have 10 different paths in the first step and for two steps we have 100 different paths. More generally, we can have maximum of 10^N different paths in N steps ahead. This shows that for large N the LLP algorithm can be computationally very complex. However, most of the paths are not stabilizing for $\mathcal{X}_b$ and some even diverge from it. For this reason, we can modify the LLP described before and prevent it from considering all the paths but only those leading to $\mathcal{X}_b$. This can be done by using the fact that to obtain a balanced load it is required to move parts from the machines with excess load to the machines which have parts less than required. For instance, by moving a part from the machine with the maximum number of parts to the machine with the minimum number of parts will for sure get the state closer to a balanced state.

The above discussion leads to the following LLP algorithm which moves a part from x^* to x_* by firing all the events in the path from x^* to x_* and guarantees that after each iteration of the algorithm the state of the system will be closer to the set of balanced states $\mathcal{X}_b$. Later in this chapter we will show another algorithm which is similar to this one but executes only one transition after each iteration.

Algorithm 2

1. *Check whether $x(k) \in \mathcal{X}_b$ or not. If yes stop, otherwise continue.*

2. *Find machines i and j such that $x_i(k) = x^*(k)$ and $x_j(k) = x_*(k)$.*

3. *Find a transition sequence which will move a part from the machine i to the machine j. In the search for a path visit every machine only once.*

4. *Execute all the events in that transition sequence.*

5. *Increment k and go to step 1.*

Note that the size of the look-ahead varies and can be either 1,2,3 or 4 depending on the relative positions of the x^* and x_*. This is due to the fact that on step 3 the algorithm stops expanding the possible future states as soon as a path, i.e., a transition sequence, from x^* to x_* is found. Moreover, since it visits every machine at most once it avoids cycles and the path found is the shortest path between the corresponding machines.

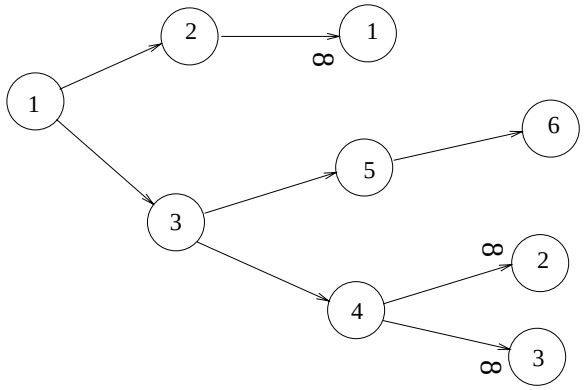

Figure 4.4: Tree generated at state $x(k) = [3, 2, 2, 2, 2, 1]^T$.

This algorithm fits the framework described in the preceding chapters, however, it is computationally more efficient. This is because instead of generating all paths in $\mathbf{X}_4(x(k))$, it generates only those which begin with moving a part from $x_i(k) = x^*(k)$. Moreover, if there are transitions to only machines j and l from the machine i, on the next step it generates possible transitions which begin by moving a part from j and l to the other machines. If a previously visited machine (or state) is encountered, which means a cycle is encountered, an infinite cost is assigned to this path and the continuations of it are not considered any more. When a path to the minimum is found, a zero cost is assigned to it, tree expansion is stopped and all the events in the path from the maximum to the minimum are executed. For example, assume that $x(k) = [3, 2, 2, 2, 2, 1]^T$. then the tree generated is as shown in Figure 4.4. After each iteration of this algorithm the load is closer to balanced state. We state this more formally next.

Proposition 4 *The balanced set $\mathcal{X}_b$ of the FMS shown in Figure 4.3 is asymptotically stable under the LLP controller of Algorithm 2.*

Proof: Assume that at time k, $x(k) \notin \mathcal{X}_b$. It is clear that either $x^*(k) > \max_{1 \leq i \leq 6} y_i$ for all $y \in \mathcal{X}_b$ or $x_*(k) < \min_{1 \leq i \leq 6} y_i$ for all $y \in \mathcal{X}_b$, or both hold together. Therefore, increasing of x_* or decreasing of x^* will lead to a decrease in the distance of the given state, $x(k)$, to the set of balanced states, $\mathcal{X}_b$. To show this assume that at time k, $x_i(k) = x^*(k)$ and $x_j(k) = x_*(k)$. After moving a part from x_i to x_j (no matter in how many movements is this performed, therefore assume 4 movements since this is the maximum number of movements possible) we have $x_i(k+4) = x^*(k) - 1$ and $x_j(k+4) = x_*(k) + 1$. It is evident that $\rho(x(k), \mathcal{X}_b) > \rho(x(k+4), \mathcal{X}_b)$. Moreover,

$\rho(x(k), \mathcal{X}_b) \geq \rho(x(k+l), \mathcal{X}_b), \forall l = 1, 2, 3$. Therefore, the metric, or distance, between current state and the set of balanced states is non-increasing. Furthermore, it is decreasing at the end of each iteration of the algorithm.

Choose Lyapunov function $V(x) = \rho(x, \mathcal{X}_b)$ and let $\psi_1(x) = \psi_2(x) = x$. Therefore, the condition $\psi_1(\rho(x, \mathcal{X}_b)) \leq V(x) \leq \psi_2(\rho(x, \mathcal{X}_b))$ is satisfied by definition. Moreover, $\rho(x, \mathcal{X}_b)$ and therefore $V(x)$ is non-increasing function as was shown above. Furthermore, because of the arguments above we have $V(x(k+4)) < V(x(k)), \forall k$. Therefore, since $V(x) \geq 0$ and strictly decreasing on every 4 steps $V(x(k)) \to 0$ as $k \to \infty$, therefore by Theorems 1 and 2 $x(k) \to \mathcal{X}_b$ as $k \to \infty$.

■

The crucial idea in the above algorithm is to move a part from the machine with the maximum number of parts to the machine with the minimum number of parts. It helps us to design an LLP which is a computationally very simple and it guarantees a constant decrease in the "distance" to the set of balanced states, $\mathcal{X}_b$. However, it is different from the LLP described before, since it uses an information about the system and also executes all the events in the "best" path found. Throughout the analysis above we saw that moving a part from the x^* to any other machine ends up in a nondecreasing distance to $\mathcal{X}_b$. This suggests an algorithm which does not need to execute all the events in the path from x^* to x_* as in the original LLP and still guarantees convergence to a balanced state.

Algorithm 3 *In the Algorithm 2 above change step 4 to: Execute the first event in the path leading to moving a part from $x^*(k)$ to $x_*(k)$. The other steps in Algorithm 2 remain unchanged.*

Before proceeding with the analysis of this algorithm we should note that in an unbalanced state $x(k)$ one of the following holds

1. $x_*(k) < N_d$
2. $N_r = 0$ and $x^*(k) > N_d$
3. $N_r \neq 0$ and $x^*(k) > N_d + 1$
4. Condition 1 hold together with condition 2 or condition 3.

Now, consider the case when a part is moved from machine the i to the machine j at time k. Then, in general, one of the following cases holds

1. If $x_i(k) > N_d$ and $x_j(k) > N_d$ then $\rho(x(k+1), \mathcal{X}_b) = \rho(x(k), \mathcal{X}_b)$
2. If $x_i(k) > N_d$ and $x_j(k) < N_d$ then $\rho(x(k+1), \mathcal{X}_b) < \rho(x(k), \mathcal{X}_b)$
3. If $x_i(k) < N_d$ and $x_j(k) > N_d$ then $\rho(x(k+1), \mathcal{X}_b) > \rho(x(k), \mathcal{X}_b)$
4. If $x_i(k) < N_d$ and $x_j(k) < N_d$ then $\rho(x(k+1), \mathcal{X}_b) = \rho(x(k), \mathcal{X}_b)$

Now we present the following proposition which states that the set of balanced states $\mathcal{X}_b$ will be asymptotically stable under the described algorithm.

Proposition 5 *The balanced set $\mathcal{X}_b$ of the 6 machine FMS shown in Figure 4.3 is asymptotically stable under the LLP controller of Algorithm 3.*

Proof: For the LLP described in Algorithm 3 the above cases 3 and 4 cannot hold since the algorithm chooses $x^*(k)$ as the source for the part to be moved and $x_i(k) < N_d$ implies that $x_i(k) \neq x^*(k)$. For this reason, after each time step $\rho(x(k+1), \mathcal{X}_b) \leq$

$\rho(x(k), \mathcal{X}_b)$ holds. Now we have to show that once in a finite number of iterations this inequality holds strictly.

Before showing this, it is worthwhile to discuss whether infinite cycles are possible under the supervision of the algorithm or not. Possible cycles of the uncontrolled system are passing a part back and forth between machines 1 & 2, 3 & 4, 5 & 6 or cycles among 1-3-4-2, or 3-5-6-4. However, under the proposed algorithm these cycles are not possible and we address this next.

Now consider machines 1 and 2. We will show that once a part is passed from 1 to 2 it is not possible to pass it back from 2 to 1 and vice versa. Assume $x_1(k) = x^*(k)$. There are two possibilities for moving a part; either to machine 2 or to machine 3. Since the path to machine 2 is a dead end, in order for it to lie on the path from $x_1(k) = x^*(k)$ to $x_*(k)$, $x_2(k) = x_*(k)$ should hold, otherwise the algorithm will move a part to machine 3. Therefore, if a part from machine 1 is moved to machine 2, $\rho(x(k+1), \mathcal{X}_b) < \rho(x(k), \mathcal{X}_b)$. Since $x_2(k+1)$ cannot be the maximum on the next step, it is not possible to pick a part from it for movement. For this reason, when a part is moved once from 1 to 2 the algorithm does not allow a part to be moved from 2 back to 1.

If $x_2(k) = x^*(k)$, then a part from machine 2 is moved to machine 1. In order to pass this part back to machine 2 on the next step we need $x_1(k+1) = x^*(k+1)$ and $x_2(k+1) = x_*(k+1)$ which is not possible. Therefore, under this algorithm once a part is moved from 2 to 1 it is not possible to pass it back to 2. The analysis for machines 3 and 4, and 5 and 6 is similar.

Now let us consider the cycle among the machines 1-3-4-2. (The analysis for the other cycle 3-5-6-4 is similar.) Assume again that machine 1 holds the maximum,

i.e., $x_1(k) = x^*(k)$. In order for the robot to move a part from 1 to 2, $x_2(k) = x_*(k)$ should hold, otherwise it will move the part to 3. After a part is moved to 3, in order to pick a part from 3 to move to the other machines $x_3(k+1) = x^*(k+1)$ is required. Moreover, to pass this part to 4, either $x_2(k+1) = x_*(k+1)$ or $x_4(k+1) = x_*(k+1)$ should hold, because if $x_5(k+1) = x_*(k+1)$ or $x_6(k+1) = x_*(k+1)$ then the part will be passed to 5, not to 4. Since 2 cannot be the minimum, it should be the case that either 4 is the minimum and the part is passed to it, or 5 or 6 is the minimum and the part is moved to 5 breaking the loop. Note that by moving a part we obtain either decrease in the distance to the set of balanced states, i.e., $\rho(x(k+1), \mathcal{X}_b) < \rho(x(k), \mathcal{X}_b)$, or breaking the cycle. Actually, having a minimum among the machines of the hypothesized cycle means that this cycle is not possible because the algorithm never will pick a part from the machine with the minimum to move it out, which also disallows the cycle. In fact, for the loop 1-3-4-2 to happen, one of these machines should hold the x^*, moreover all others in the loop should hold $x^* - 1$ so that the x^* alternates among them. However, this means that either 5 or 6 hold the x_*, which on the other hand will result in breaking the loop at machine 3 by passing a part to machine 5 as was discussed above.

Proceeding with the analysis, we note that if after a transition of a part is moved to $x_j(k) = x_*(k)$ then $\rho(x(k+1), \mathcal{X}_b) < \rho(x(k), \mathcal{X}_b)$ or otherwise $x_j(k+1) = x_*(k+1) = x_j(k) = x_*(k)$, i.e., the same machine still holds the minimum. For this reason, at time $k+1$ the algorithm will still seek a path to the machine j, and this will continue until a part is passed to it. Since we have finite number of machines and parts, and there are no cycles possible, then after time k there exists finite time step l such that $\rho(x(k+1), \mathcal{X}_b) < \rho(x(k), \mathcal{X}_b)$. Having showed this we can define Lyapunov function

as $V(x) = \rho(x, \mathcal{X}_b)$ and the rest of the proof follows in a similar way as that of Proposition 4.

■

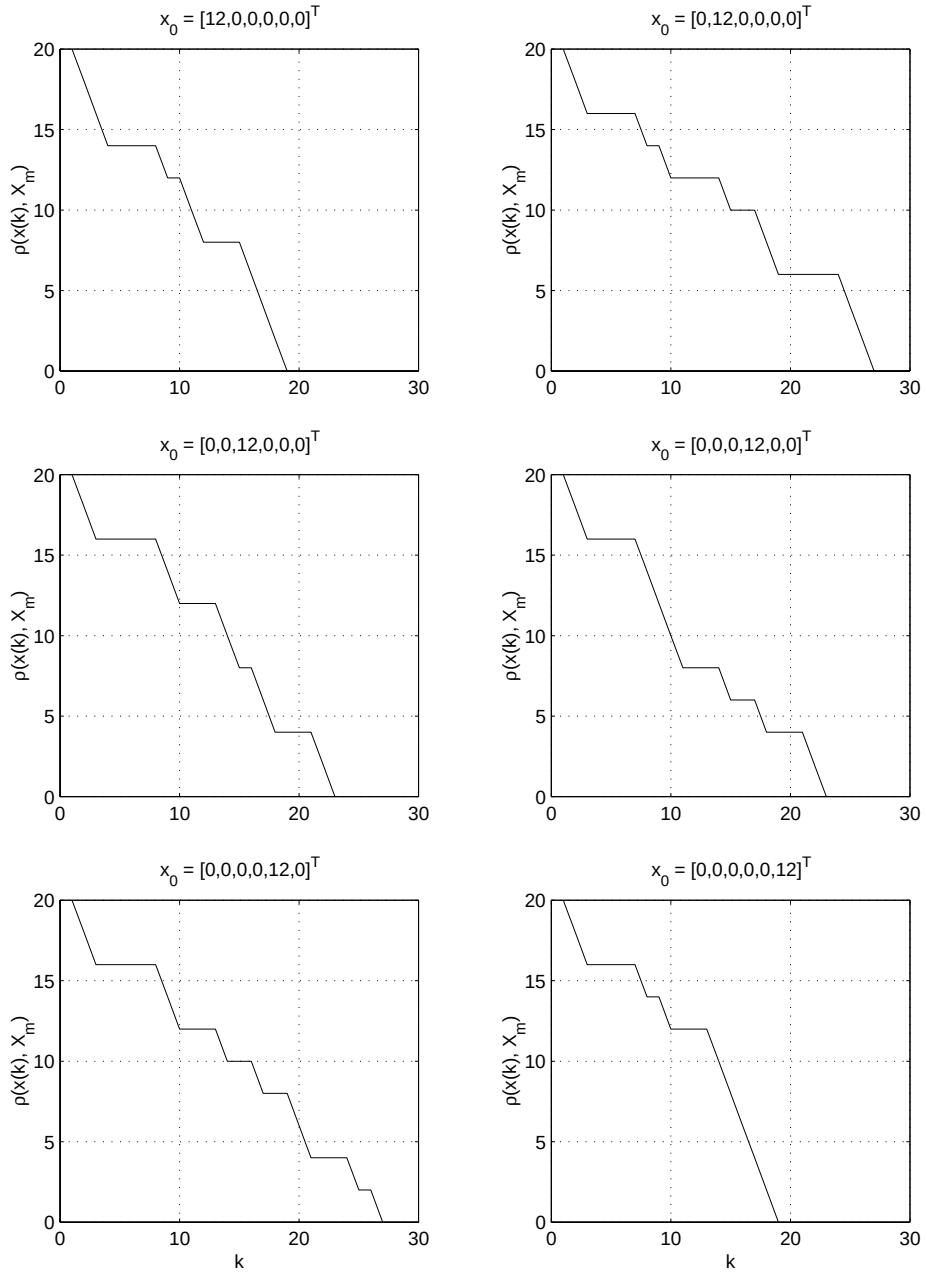

Figure 4.5: Plot of $\rho(x(k), \mathcal{X}_b)$ for different initial conditions.

In order to enhance the study of Algorithm 3 above we also performed a simulation analysis of the system. The results are shown in Figure 4.5 where we plotted $\rho(x(k), \mathcal{X}_b)$ for various different initial conditions. One can immediately see that the simulation results completely support the theoretical results obtained.

Finally, we want to discuss the computational complexity of the algorithm. We claim that the algorithm is not computationally complex. It expands a state tree while searching for a path from x^* to x_* whose size depends on the *dimension* of the state space not on the number of the states. In particular for the example given, in the worst case it expands a 4 level tree and visits all the machines, i.e., all 6 machines at most once, which does not require a lot of computational time. If one still needs to decrease the computational time then it is worthwhile to note that there is only one path out of some machines (e.g. 2 and 5). Which implies that if one of these machines holds the maximum there is no choice but to use the only path available. Therefore, there is no need to look ahead and expand a tree of future states and one can move a part without it. This can be implemented by a simple if statement which does not add overhead to the program.

CHAPTER 5

Conclusions

In this chapter we will summarize the work done in this thesis and also point some potential future research directions.

5.1 Summary and Contributions

In this work we have analyzed DES in the framework of [1, 2]. In Chapter 2 we introduced the DES model, provided some mathematical framework, presented the closed-loop system and introduced the control problem. The main difference between the model that we use and the models in the literature is that we allow more then one event to occur together. In our framework a controllable and an uncontrollable event can occur simultaneously and the control objective is to stabilize the invariant set despite the uncontrollable events. In Chapter 3 we first analyzed stabilizability in DES. We defined a notion of controllability which is more general then the ones existing in the literature [8, 9, 10] and stated it as a sufficient condition for stabilizability. Then, we addressed the optimality and optimal stabilization concepts for the class of DES we are concerned with. The ideas of optimal control are similar to these existing in the literature [3, 12, 15], however, the performance index to be minimized is different since the system itself and the control objectives are different

in our framework. We showed that the invariant set is optimally stabilizable if and only if it is stabilizable. After that, we used the idea of using limited look-ahead policies for the control of DES introduced in [5, 6]. The formulation in our framework is different since we are mostly concerned with the state of the system whereas in [5, 6] they are dealing with event sequences or the language generated by the system. In applying the idea of look-ahead we used also the ideas from optimal stabilization hence the limited look-ahead controller is not only stabilizing but also optimal (or suboptimal) in a sense that on every step it chooses the best move as long as the finite horizon is concerned.

In Chapter 4 we presented two illustrative examples. The first example is a level control of a surge tank and the second one is a load balancing in a flexible manufacturing system. We showed that under the proposed limited look-ahead policy scheme the level in the tank will converge and remain within a predefined set of thats and that the load of the flexible manufacturing system will become balanced after finite number of transitions. The first example was important because it showed that an appropriate set of states can be made an invariant set and can be asymptotically stabilized despite the uncontrollable events and the second example showed the importance of cycles, and how they can be avoided or broken for stabilization.

5.2 Future Research Directions

There are several possible future directions which can be extensions of this work. First of all, recall that here we assumed that we know the state of the system perfectly. More challenging and important work would be to derive stability, stabilizability and optimality conditions for DES in our framework in which the state is not perfectly

known. Moreover, deriving necessary and sufficient conditions under which the limited look-ahead controller will perform as well as in the full state feedback case is an important future direction.

Other basic assumption this work is based on is that the model of the system is perfect or it exactly reflects the system behavior. Derivation of conditions under which the proposed algorithm will still perform well despite model uncertainties is certainly a challenging research area.

Finally, inspired from the idea of look-ahead one may try to perform a stability analysis of a general planning system with model uncertainties, and uncontrollable and unobservable events. Proving stability of such system will for sure be a state of the art contribution to the theory of intelligent control.

BIBLIOGRAPHY

[1] Kevin M. Passino and Kevin L. Burgess, *Stability Analysis of Discrete Event Systems*, John Wiley and Sons, NY 1998

[2] K. M. Passino, A. N. Michel and P. J. Antsaklis, *Lyapunov Stability of a class of discrete event systems*, IEEE Transactions on Automatic Control, Vol. 39, No. 2, Feb. 1994, pp. 269-279

[3] K. M. Passino and P. J. Antsaklis, *On the Optimal Control of Discrete Event Systems*, Proceedings of the 28th IEEE Conference on Decision and Control, Tampa, Florida, USA, Dec. 13-15 1989, Vol. 3, pp. 2713-2718

[4] C. M. Özveren and A. S. Willsky and P. J. Antsaklis, *Stability and stabilizability of Discrete Event Dynamic Systems*, Journal of the Association for Computing Machinery, Vol. 38. No. 3, July 1991, pp. 730-752

[5] S.-L. Chung S. Lafortune and F. Lin, *Limited Lookahead Policies in Supervisory Control of Discrete Event Systems*, IEEE Transactions on Automatic Control, Vol. 37, No. 12, Dec. 1992, pp. 1921-1935

[6] S.-L. Chung and S. Lafortune, and F. Lin, *Recursive computation of Limited Lookahead Supervisory Controls for Discrete Event Systems*, Discrete Event Dynamic Systems: Theory and Applications, Vol. 3, 1993, pp. 71-100

[7] K.-H. Cho and J.-T. Lim, *Stability and robustness of discrete event dynamic systems*, International Journal of Systems Science, Vol. 28. No. 7, 1997, pp. 691-703

[8] P. J. Ramadge and W. M. Wonham, *Supervisory Control of a Class of Discrete Event Processes*, SIAM Journal of Control and Optimization, Vol. 25, No. 1, Jan. 1987, pp. 206-230

[9] P. J. Ramadge and W. M. Wonham, *The Control of Discrete Event systems*, IEEE Proceedings, Vol. 77. No. 1, Jan. 1989, pp. 81-98

[10] W. M. Wonham and P. J. Ramadge, *On the Supremal Controllable Sublanguage of a Given Language*, SIAM Journal of Control and Optimization, Vol. 25. No. 3, May 1987, pp. 637-659

[11] Y. Brave and M. Heymann, *On Stabilization of Discrete-Event Processes*, Proceedings of the 28th IEEE Conference on Decision and Control, Tampa, Florida, USA, Dec. 13-15 1989, Vol. 3, pp. 2737-2742

[12] Y. Brave and M. Heymann, *On Optimal Attraction in Discrete-Event Processes*, Information Sciences, Vol. 67, 1993, pp. 245-276

[13] F. Lin, *Robust and Adaptive Supervisory Control of Discrete Event Systems*, IEEE Transactions on Automatic Control, Vol. 38, No. 12, Dec. 1993, pp. 1848-1852

[14] V. K. Garg and R. Kumar, *A State-Variable Approach for Controlling Discrete Event Systems with Infinite States*, Proceedings of 1992 American Control Conference, Vol. 4, pp. 2809-2813

[15] R. Kumar and V. K. Garg, *Optimal Supervisory Control of Discrete Event Dynamic Systems*, SIAM Journal of Control and Optimization, Vol. 33. No. 2, March 1995, pp. 419-439

[16] R. Sengupta and S. Lafortune, *An Optimal Control Theory for Discrete Event Systems*, SIAM Journal of Control and Optimization, Vol. 36. No. 2, March 1998, pp. 488-541

[17] S. S. Keerthi and E. G. Gilbert *Optimal Infinite-Horizon Feedback Laws for a General Class of Constrained Discrete-Time Systems: Stability and Moving-Horizon Approximations*, Journal of Optimization Theory and Applications, Vol. 57, No. 2, May 1988, pp. 265-293

[18] D. P. Bertsekas *Dynamic Programming and Optimal Control: Volume 1*, Athena Scientific, Belmont, Massachusetts 1995

[19] D. P. Bertsekas *Dynamic Programming and Optimal Control: Volume 2*, Athena Scientific, Belmont, Massachusetts 1995

[20] Donald E. Kirk, *Optimal control theory; an introduction*, Prentice-Hall, N.J. 1970