On Interconnection and Equivalence of Continuous and Discrete Systems

A Behavioral Perspective

Dutch Institute of Systems and Control

Compositional Analysis and Specification of Hybrid Systems

The research described in this thesis was undertaken at the Department of Applied Mathematics, in the Faculty EWI, Universiteit Twente, Enschede. The funding of the research was provided by the NWO Grant through project number 617.023.002 (Compositional Analysis and Specification of Hybrid Systems).

Printed by Wöhrmann Printing Service, Zutphen, The Netherlands.
The summary in Dutch was done by Stefan Strubbe and Jan Willem Polderman.

ISBN: 90-365-2145-9

ON INTERCONNECTION AND EQUIVALENCE OF CONTINUOUS AND DISCRETE SYSTEMS: A BEHAVIORAL PERSPECTIVE

DISSERTATION

to obtain
the doctor's degree at the University of Twente,
on the authority of the rector magnificus,
prof. dr. W.H.M. Zijm,
on account of the decision of the graduation committee,
to be publicly defended
on Friday 11 February 2005 at 13.15 hours

by

Anak Agung Julius
born on 7 July 1977
in Palembang, Indonesia

This dissertation has been approved by the promotor
Prof. dr. A. J. van der Schaft

Composition of the Graduation Committee

Chairperson:	
Prof. dr. W.H.M. Zijm	Universiteit Twente, EWI
Secretary:	
Prof. dr. W.H.M. Zijm	Universiteit Twente, EWI
Promotor:	
Prof. dr. A.J. van der Schaft	Universiteit Twente, EWI
Members:	
Prof. dr. J. C. Willems	Katholieke Universiteit Leuven, Belgium
Prof. dr. J. M. Schumacher	Universiteit van Tilburg, The Netherlands
Prof. dr. E. Brinksma	Universiteit Twente, EWI
Prof. dr. G. J. Pappas	University of Pennsylvania, USA
Dr. J. W. Polderman	Universiteit Twente, EWI
Dr. R. Langerak	Universiteit Twente, EWI

Contents

Summary

In this thesis a unified systems theoretic framework for modeling and analysis of dynamical systems is presented. The approach is based on Willems' behavioral approach to systems theory. The main tenet of the behavioral approach is that systems are identified by their behavior, which is the collection of all possible trajectories of the system.

We present results on the classes of linear systems and discrete event systems, as well as hybrid systems.

Two main themes are discussed in this book, namely interconnection and external equivalence of systems.

We show how the behavioral notion of interconnection can be applied to dynamical systems in the general sense, as well as systems of the classes mentioned above. Interconnection in the behavioral perspective can be regarded as synchronization of the trajectories of the systems. In this thesis we consider full interconnection as well as partial interconnections.

In a full interconnection, complete information about the behaviors involved is shared. In a partial interconnection, the behaviors involved do not share all information. The fact that some information is hidden in the interconnection is represented by a behavioral projection. A projection is a surjective map that maps one behavior onto another, possibly of a different type. After projection, some information about the behavior may be lost, since it is possible that multiple trajectories are mapped onto the same trajectory in the domain.

A topic that is closely related to systems interconnection is that of control. We analyze control problems in a general behavioral setting, and give conditions for solvability of several types of control problems. We also treat the problem of control with minimal interaction for linear systems. This problem is about designing a controller that has as few control variables as possible.

The discussion on external equivalence is mainly centered around the so called external behavior equivalence and bisimulation. We factor the signal space of the behavior into internal and external part. The external behavior of a system is simply its behavior projected to the external part of the signal space.

Bisimulation is a concept that originates from the field of discrete event systems and concurrent processes. We cast the notion of bisimulation into the behavioral setting, and use it to analyze external equivalence of systems. It is generally known that bisimilarity is a stronger notion of systems equivalence than external behavior equivalence. However, with the framework developed in this thesis we prove that for continuous time linear systems, the two notions coincide.

Samenvatting

In dit proefschrift wordt een algemeen systeemtheoretisch kader voor dynamische systemen gepresenteerd. De benadering is gebaseerd op Willems' *behavioral approach* voor systeemtheorie. Het basisidee van de *behavioral approach* is dat systemen worden vereenzelvigd met hun gedrag (Engels: *behavior*) in plaats van met de vergelijkingen die dit gedrag beschrijven. Met het gedrag wordt de verzameling van uitkomsten bedoeld die voldoen aan de wetten van het systeem. Deze uitkomsten kunnen tijdfuncties, meestal trajecten genoemd, zijn maar ook algemener, functies van combinaties van gebeurtenissen en tijd.

We leiden resultaten voor de klasses van lineaire systemen en *discrete event* systemen, alsmede voor de klasse van hybride systemen.

De twee hoofdthema's van dit proefschrift zijn interconnectie en externe equivalentie van systemen.

We laten zien hoe de notie van interconnectie zoals bestudeerd in de *behavioral approach* kan worden toegepast op dynamische systemen in het algemeen, alsook op de specifieke bovengenoemde systeemklasses. Interconnectie kan, vanuit het perspectief van de *behavioral approach*, worden gezien als synchronisatie van de trajecten het systeem. In dit proefschrift beschouwen we zowel volledige als partiële interconnceties.

Bij volledige interconnectie wordt alle informatie van de betrokken *behaviors* gebruikt. Bij partiële interconnectie wordt slechts een gedeelte van de informatie gebruikt. Het feit dat een gedeelte van de informatie verborgen wordt voor de interconnectie, wordt gerepresenteerd door een *behavioral* projectie. Een projectie is een surjectieve afbeelding die een behavior afbeeldt op een ander *behavior*, mogelijk van een ander type. Na projectie kan een gedeelte van de informatie van het *behavior* verloren zijn gegaan, omdat het mogelijk is dat verschillende trajecten worden afgebeeld op hetzelfde traject.

Een onderwerp dat nauw verwant is met systeeminterconnectie is *besturing*. We analyseren besturingsproblemen in een algemene *behavioral setting* en geven condities voor de oplosbaarheid van diverse klassen besturingsproblemen. We behandelen ook het probleem van besturing-met-minimale-interactie voor lineaire systemen. Bij dit probleem gaat het er om een regelaar met zo min mogelijk besturingsvariabelen te ontwerpen.

De discussie betreffende externe equivalentie wordt grotendeels toegespitst op externe *behavior* equivalentie en op bisimulatie. We splitsen de signaalruimte van het systeem (*behavior*) op in een intern en een extern gedeelte. Het externe *behavior*

van een systeem is simpelweg de projectie van het systeemgedrag op het externe deel van de signaalruimte.

Het begrip bisimulatie is afkomstig uit de theorie van *discrete event* systemen en concurrente processen (*concurrent processes*). We definiëren bisimulatie binnen de *behavioral setting* en gebruiken dat om de externe equivalentie van systemen te analyseren. Het is algemeen bekend dat bisimulatie een sterkere equivalentienotie is dan externe *behavior* equivalentie. Echter, met behulp van het kader ontwikkeld in dit proefschrift, bewijzen we dat voor continue tijd lineaire systemen beide noties op het zelfde neerkomen.

Notation

Symbol	Description	Page
$\wedge_{\tau_2}^{\tau_1}$	concatenation operator indexed by time τ_1 and τ_2	7
$\mathbb{R}, \mathbb{R}_+$	real numbers, nonnegative real numbers	8
$\mathbb{Z}, \mathbb{Z}_+$	integers, nonnegative integers	8
$\mathbb{R}^{\mathrm{g}\times\mathrm{q}}[\xi]$	polynomial matrices with g rows and q columns, with indeterminate ξ	9
$\mathfrak{L}_{\mathrm{c}}^{\mathrm{q}}$	continuous time LTI behaviors with q variables, consisting of strong solutions to a kernel representation	9
σ	unit time-shift operator	9
$\mathfrak{L}_{\mathrm{d}}^{\mathrm{q}}$	discrete time LTI behaviors with q variables	9
$\mathfrak{C}^{\infty}(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$	infinitely differentiable functions mapping $\mathbb{R}$ to $\mathbb{R}^{\mathrm{q}}$	10
$\mathfrak{D}(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$	test functions mapping $\mathbb{R}$ to $\mathbb{R}^{\mathrm{q}}$	10
$\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$	locally integrable functions mapping $\mathbb{R}$ to $\mathbb{R}^{\mathrm{q}}$	10
$\bar{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$	continuous time LTI behaviors with q variables, consisting of weak solutions to a kernel representation	11
$\overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$	continuous time LTI behaviors with q variables, consisting of left-continuous weak solutions to a kernel representation	12
$L(A)$	the language generated by the finite state automaton A.	16
$L_{\mathrm{m}}(A)$	the marked language of the finite state automaton A.	16
$L(E)$	the language associated with the regular expression E.	18
T_ζ	the hybrid time axis of the hybrid trajectory ζ	22
$N_\zeta(t)$	the event multiplicity of T_ζ at time $t \in \mathbb{R}_+$	22

Symbol	Description	Page
$\|$	interconnection operator for behaviors	27
$\preceq, \eqcirc$	partial ordering and equivalence of behavioral projections	37
$\preccurlyeq, \approx$	partial ordering and equivalence of dynamic maps	56
$\mathbb{T}^+$	the positive elements of the underlying group of $\mathbb{T}$	59
$\mathfrak{B}_\omega$	the suffix behavior corresponding to $\mathfrak{B}$	84
$\mathtt{n}()$	the McMillan degree of a linear system	88
$\mathtt{p}()$	the number of outputs of a linear system	92
π_v	projection to the external signal space	123
$\approx_{\mathrm{ext}}$	external behavior equivalence	124
π_d	projection to the internal signal space	123
$\approx_{\mathrm{bis}}$	bisimilarity	126
$\approx_{\mathrm{sim}}$	mutual simulation	129

1 Introduction

"Begin at the beginning and go on till you come to the end; then stop." - Lewis Carroll

Herein we provide an introduction for the research presented in this book. The first part of this chapter provides a summary and background information about the results. In the second part we present the structure of this book, and highlight the contributions made in it.

In this chapter the reader might find a few terminologies that have not been explained or some ideas that appear vague. They will be explained later in the following chapters. They are included in this chapter for the ease and completeness of presentation.

1.1 Background

In this thesis we develop a unified systems theoretic framework for modeling and analysis of dynamical systems. In doing so, we follow the behavioral approach to systems theory. The main tenet of the behavioral approach is that systems are identified by their *trajectories*. The collection of trajectories of a system is called its *behavior*.

The behavioral approach to systems theory is pioneered by Jan Willems in a series of his papers [Wil86a, Wil86b, Wil87]. Subsequent works on behaviors of *linear time invariant systems* are documented in, for example, [Wil91, Wil97, PW98] and many more that we shall mention later in this book. Although most applications of the behavioral approach are for linear time invariant systems, there have also been applications in other fields, such as nonlinear systems [Sch03a, PP04], discrete event systems [Sme87, Sme89], and hybrid systems [MR99, JSS03, BKU04].

We aim to develop a unified framework based on the behavioral approach to facilitate modeling and analysis of general dynamical systems. We shall show how this framework can be applied to, in particular, linear systems, discrete event systems and hybrid systems.

The emergence of new branches of systems and control theory, such as hybrid systems [ACHH93, Alu95, Hen96, BBM98, SS00], has spurred a cross fertilization of ideas among various existing branches. Hybrid systems, for example, have attracted application of theories from continuous time dynamical systems as well as discrete event systems. Our motivation is to provide a framework, within which ideas from different branches of systems theory can be brought and applied to other branches.

As indicated by the title of the book, there are two main themes that are discussed in this book, namely *interconnection* and *external equivalence* of systems.

Interconnection of systems in the behavioral perspective is understood as synchronization of the trajectories of the systems. Roughly speaking, the trajectories of two systems are synchronized if they are the same function of time. The trajectories can be either fully synchronized or partially synchronized. By partial we mean that only a certain aspect or dimension of the trajectories is synchronized. These kinds of synchronizations are related to the so called full interconnection, as well as partial interconnection. Interconnection of system is discussed in more detail in Chapter 3.

One obvious reason why it is desirable to study systems interconnection is that it enables modular modeling. That is, it enables us to model complex systems by interconnecting simpler subsystems. Another reason for it, which we shall discuss in more detail in Chapter 4, is that interconnection is related to control problems. A control problem in the behavioral perspective can be viewed as follows. Given a system to be controlled (called the *plant*), the problem is to find another system (called the *controller*) that when interconnected with the plant yields a desired specification [Wil97, Tre99, Sch03a, Bel03]. For linear time invariant systems, control in the behavioral perspective can be regarded as a generalization of that in the classical input-output perspective [Wil97, Bel03]. Some results on control problems in a general behavioral setting has also been presented in [SJ02, Sch03a, JS04a]. In addition to that, there has also been research on behavioral control problems in discrete event systems [Sme87, Sme89] and hybrid systems [MR99, JSS03].

The second theme of the thesis, external equivalence of systems, is covered in Chapter 5. There are various existing notions of systems equivalence in systems theory. For linear systems, there are notions such as transfer function equivalence and state space isomorphism. For discrete event systems, there is a long list of equivalence notions [HS96], among which are the trace equivalence and bisimulation [Mil89].

In particular, we are interested in *bisimulation*. The concept of bisimulation originates in the field of discrete event systems / concurrent processes [Par81, Mil89] and has been applied to other classes of systems such as, hybrid systems [LPS98, AHLP00, PvD04], linear systems [Pap03, Sch04a, Sch04b], and nonlinear systems [Tab04, Sch04b]. We can specify a few reasons why bisimulation is a favorable notion of equivalence for discrete event systems. First, bisimulation is computationally cheaper to check than, for example, trace equivalence [HS96]. Second, bisimulation can be used for reduction of the state space of systems [LPS98]. Third, bisimulation conserves the temporal logic properties of, for example, LTL and CTL [AHLP00].

Our interest is to put bisimulation as a notion of systems equivalence in the systems theoretic framework that we develop. As the framework is considerably general, it is interesting to learn what existing and new results we can derive from it.

1.2 Layout and contributions

The material presented in this book is based on the following papers [SJ02, JS03, JSS03, WBJT03, JS04a, JS04b, JWBT04, JPS05]. The book is divided into six chapters. The content of the remaining chapters is briefly summarized as follows.

Chapter 2. In this chapter we lay the mathematical foundation for the discussion in the subsequent chapters. We also explain and review how the behavioral perspective can be used for linear systems, discrete event systems and hybrid systems. This chapter is mainly based on the following papers [JSS03, JS04a], as well as other references mentioned in the chapter.

Chapter 3. In this chapter we discuss the notion of interconnection of systems in the behavioral perspective. The discussion begins with *full interconnection*, followed with *behavioral projection* and *partial interconnection*. We explain how interconnection in the behavioral perspective fits the notion of systems interconnection of the classes covered in the previous chapter. In addition, we also define and discuss the notion of *dynamic maps*, which is much used in the later chapters. Results in this chapter are mainly based on the papers [JS03, JS04a].

Chapter 4. This chapter contains a discussion on the control problem in the behavioral setting. We begin with discussing the control problem in a general setting and introduce the so called *canonical controllers*. We also introduce some constraints that can possibly arise in the control problem, particularly for linear systems. We study the notion of *achievability* of the specification given in the control problem. Necessary and sufficient conditions for the existence of a solution for some types of control problems, which correspond to the achievability of the specification, are also derived. The conditions for achievability can be thought of as defining the limit of performance of the plant, as they tell exactly which specifications can be achieved. The chapter is concluded by a treatment of the problem called *control with minimal interaction* The material in this chapter is mostly based on the following papers [SJ02, JS03, WBJT03, JS04a, JPS05].

Chapter 5. In this chapter we discuss some notions of external equivalence of systems. We begin by introducing the notion of external behavior equivalence, and then review the notion of bisimulation and other issues related to it. We cast bisimulation in the framework that has been built so far, and derive some results from it. This chapter is primarily based on the paper [JS04b].

Chapter 6. This chapter contains conclusions that can be drawn from the discussion so far and highlights the contributions made in this thesis. At the end of the chapter we also present a few recommendations on possible future research directions.

2

Behaviors and systems

"The behavior is all that is the case." - Shiva Shankar paraphrasing Ludwig Wittgenstein.

2.1 Introduction

The behavioral approach to systems theory was pioneered by Jan Willems, in a series of his works beginning in the 1980s [Wil86a, Wil86b, Wil87]. Most of the material presented in this chapter is adopted from subsequent works by him and his coauthors, particularly [Wil91, Wil97, PW98].

In short, the behavioral approach to systems theory identifies *systems* with the collection of all *trajectories* consistent with the mathematical laws of the systems.

Definition 2.1. A **dynamical system** Σ is defined as a triple $(\mathbb{T}, \mathbb{W}, \mathfrak{B})$, where $\mathbb{T}$ is called the *time axis*, $\mathbb{W}$ is called the *signal space*, and $\mathfrak{B} \subset \mathbb{W}^{\mathbb{T}}$ is called the *behavior* of the system. The pair $(\mathbb{T}, \mathbb{W})$ is defines the *type* of the system and the behavior.

A behavior is a collection of trajectories, which are functions mapping the time axis to the signal space. We do not require the trajectories to be total functions, as they can be partial functions as well. The behavior of a dynamical system is the collection of all possible trajectories of the system. A trajectory is possible if it is consistent with the laws describing the system. Thus,

$$\mathfrak{B} := \{w : \mathbb{T} \to \mathbb{W} \mid w \text{ is compatible with the laws of } \Sigma\}. \tag{2.1}$$

The type of a behavior indicates the type of the trajectories it contains. That is, the time axis on which they are defined and the space in which they assume their value.

Example 2.2. Newton's law of motions stipulates that the force $(\mathbf{F})$ needed to accelerate a physical body is proportional to the mass $(\mathbf{m})$ and the acceleration $(\mathbf{a})$. This is epitomized in the well-known relation

$$F = m \cdot a. \tag{2.2}$$

A dynamical system that describes this theory can be written as a triple $(\mathbb{T}, \mathbb{W}, \mathfrak{B})$. The trajectories of this system are trajectories of the acceleration and the force variables. Each variable is vector valued, so we take $\mathbb{W}$ as, for example, $\mathbb{R}^3 \times \mathbb{R}^3$. The evolution of the variables takes place continuously, so we take $\mathbb{T}$ as, for example, $\mathbb{R}$. The behavior $\mathfrak{B}$ is then described by

$$\mathfrak{B} := \left\{(F, a) \in \mathbb{R} \to \mathbb{R}^3 \times \mathbb{R}^3 \mid \forall t \in \mathbb{R},\ F(t) = m \cdot a(t)\right\}. \tag{2.3}$$

If we denote the position of the center of gravity of the body as $\mathbf{x} \in \mathbb{R}^3$, then the behavior of the dynamical system that describes the relation between the force and the position of the body is

$$\mathfrak{B} := \left\{(F, x) \in \mathbb{R} \to \mathbb{R}^3 \times \mathbb{R}^3 \mid \forall t \in \mathbb{R},\ F(t) = m \cdot \frac{d^2 x(t)}{dt^2}\right\}. \tag{2.4}$$

Example 2.3. The amount of money (*balance*) in one's bank account at the end of every quarter, assuming that there is no cash flow, can be expressed as follows.

$$x_{k+1} = \lfloor (1 + r_k) \cdot x_k \rfloor, \tag{2.5}$$

where x_k is the balance at the end of the k-th quarter and r_k is the effective interest rate of that quarter. The operation $\lfloor \cdot \rfloor$ rounds down its argument to the nearest smaller integer. This dynamical system can also be written as a triple $(\mathbb{Z}_+, \mathbb{R} \times \mathbb{Z}, \mathfrak{B})$, where

$$\mathfrak{B} := \{(r, x) \in \mathbb{Z}_+ \to \mathbb{R} \times \mathbb{Z} \mid \forall k \in \mathbb{Z}_+,\ x_{k+1} = \lfloor (1 + r_k) \cdot x_k \rfloor\}. \tag{2.6}$$

Notice that the behavior in Example 2.2 is actually a linear space. That is, if $w_1, w_2 \in \mathfrak{B}$ then for any $\alpha_1, \alpha_2 \in \mathbb{R}$,

$$\alpha_1 w_1 + \alpha_2 w_2 \in \mathfrak{B}. \tag{2.7}$$

This property is not possessed by the behavior in Example 2.3.

Example 2.4. A discrete event system modelled as a finite state automaton (see more about it in Section 2.4) can be associated with the collection of strings that it executes. Consider as an example, a printer machine that operates in the following way. It can receive a printing job, this event is annotated by `job`. It must print out the job that it has received and feed out the output before it can receive another job. Suppose that the printing is associated to an event annotated by `print`, and the feeding out the printout is associated to `feed`. The behavior of this system can be defined as the collection of all strings such that `job.print.feed` always occur in this order. Any string that has, for example, `print.job.feed`, as a substring is not an element of the behavior as it corresponds to the acceptance of a new job before the printout is fed out. The behavior has $\mathbb{Z}_+$ as its time axis, and $A = \{\mathtt{print}, \mathtt{job}, \mathtt{feed}\}$ as its signal space.

Example 2.5. A hybrid system, loosely speaking, is a dynamical system that has continuous as well as discrete aspect in its dynamics (see more about it in Section 2.5). Later in this chapter we shall see the dynamics of the temperature inside an air-conditioned room modelled as a hybrid system. The evolution of the temperature itself is a continuous process, but it interacts with the state of the air conditioner, which is discrete. The behavior of this system consists of all trajectories describing not only the evolution of the system but also the events related to the changes in the discrete state. The time axis and the signal space of the behavior associated to a hybrid system are described in Section 2.5.

2.2 The structure of the time axis

We have seen some instances of the time axis in the previous section. In this book, we assume that the time axis $\mathbb{T}$ has a certain structure that we shall define in this section.

Definition 2.6. A **totally ordered commutative group** $\mathcal{G}$ is a triple $(G, +, <)$, where $(G, +)$ is a commutative group and $<$ a total ordering on G such that for any $a, b, c \in G$,

$$(a < b) \Rightarrow (a + c < b + c). \tag{2.8}$$

Throughout the book, when $\mathbb{T}$ is not specified, we shall assume that there is an underlying totally ordered group $\mathcal{G} = (G, +, <)$ such that one of the following is true.
(i) $\mathbb{T} = \mathcal{G}$.
(ii) There is a t_0 such that $\mathbb{T} = \{t \in G \mid t > t_0\}$.
(iii) There is a t_0 such that $\mathbb{T} = \{t \in G \mid t \geq t_0\}$.

Assuming that the time axis $\mathbb{T}$ has a structure as described above, we have a formal base for defining time-shifting of trajectories and concatenation. Concatenation of trajectories is defined as follows.

Definition 2.7. Take any behavior $\mathfrak{B}$ and let $(\mathbb{T}, \mathbb{W})$ be its type. For any two time instants $t_1, t_2 \in \mathbb{T}$, the **concatenation** operation $\wedge_{t_2}^{t_1}$ is a binary operation. It is defined such that for any two trajectories $w_1, w_2 \in \mathfrak{B}$,

$$w_3 := w_1 \wedge_{t_2}^{t_1} w_2 \in \mathbb{W}^{\mathbb{T}}, \tag{2.9}$$

$$w_3(t) = \left\{ \begin{array}{cc} w_1(t) & t \leq t_1 \\ w_2(t - t_1 + t_2) & t > t_1 \end{array} \right. . \tag{2.10}$$

Notice that in the definition of concatenation, on the second line of the right hand side of (2.10), we use time-shifting to connect the future of w_2 with the past of w_1. For clarity, please refer to Figure 2.1 for an illustration of the concatenation.

The following table gives examples of time axes that have the above mentioned structure.

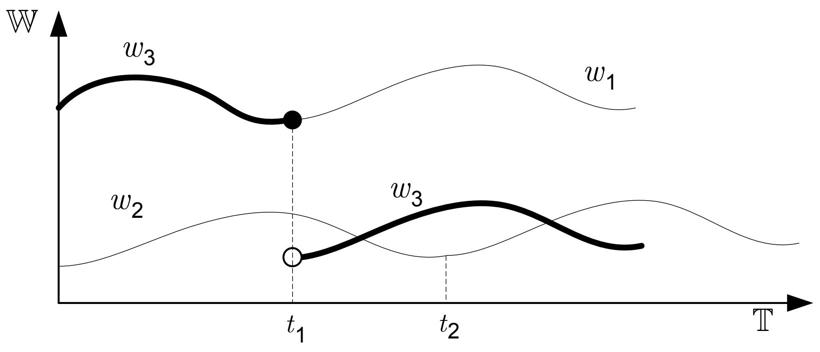


Figure 2.1: An illustration of concatenation of trajectories. The trajectory w_3 (thick line) is defined as $w_3 = w_1 \wedge_{t_2}^{t_1} w_2$.

$\mathbb{T}$	G	Ordering relation
$\mathbb{R}, \mathbb{R}_+$	$\mathbb{R}$	standard ordering of real numbers
$[t_0, \infty), t_0 \in \mathbb{R}$	$\mathbb{R}$	standard ordering of real numbers
$\mathbb{Z}, \mathbb{Z}_+$	$\mathbb{Z}$	standard ordering of integers
$(\mathbb{R}_+ \times \mathbb{Z}_+)$	$(\mathbb{R} \times \mathbb{Z})$	lexicographic ordering (Definition 2.32)

An important property of the time axis, that we shall use later in this book is given in the following lemma.

Lemma 2.8. Let $\mathcal{G} = (G, +, <)$ be a totally ordered commutative group. Let t_1 and t_2 be elements of G. The sets $\mathbb{T}_1$ and $\mathbb{T}_2$ defined as

$$\mathbb{T}_i := \{t \in G \mid t > t_i\}, i = 1, 2, \tag{2.11}$$

are isomorphic in the following sense. There exists a one-to-one mapping $\phi : \mathbb{T}_1 \to \mathbb{T}_2$ such that for all $\tau_1, \tau_2 \in \mathbb{T}_1$,

$$\phi(\tau_1) - \phi(\tau_2) = \tau_1 - \tau_2. \tag{2.12}$$

Proof. Define

$$\phi(t) := t - t_1 + t_2.$$

This mapping is one-to-one, because its inverse exists. Namely,

$$\phi^{-1}(t) = t - t_2 + t_1. \tag{2.13}$$

For all $\tau_1, \tau_2 \in \mathbb{T}_1$,

$$\begin{aligned} \phi(\tau_1) - \phi(\tau_2) &= \tau_1 - t_1 + t_2 - (\tau_2 - t_1 + t_2), \\ &= \tau_1 - \tau_2. \end{aligned} \tag{2.14}$$

□

In the remaining part of this chapter we shall review the classes of systems that we are dealing with in this book. These systems have been treated numerously in the literature before, nevertheless, it is essential to review how they all can fit into the behavioral systems theory framework.

2.3 Linear time invariant systems

A very important class of systems discussed in this book is the finite dimensional *linear time invariant systems*, which we shall also refer to as the *linear systems*. The behavioral approach to the theory of linear systems is a lively research field in systems theory. There are numerous literature covering this subject, which we are not going to enumerate exhaustively. Among them are the references shown in the beginning of this chapter. For the sake of completeness of the book, we shall also cover a review on the behavioral systems theory for linear systems.

The class of linear systems is divided into two subclasses, namely continuous time linear systems and discrete time linear systems. Continuous time linear systems are typically of type $(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$ or $(\mathbb{R}_+, \mathbb{R}^{\mathrm{q}})$, while discrete time linear systems are typically of type $(\mathbb{Z}, \mathbb{R}^{\mathrm{q}})$ or $(\mathbb{Z}_+, \mathbb{R}^{\mathrm{q}})$. Here the symbol q denotes the dimension of the variables involved in the system. For example, the system described in Example 2.2 has two variables, namely F and a, with a total dimension of six.

Polynomial matrices are essential to the behavioral theory of linear systems. We shall denote the set of g$\times$q polynomial matrices with real coefficients and indeterminate ξ as $\mathbb{R}^{\mathrm{g}\times\mathrm{q}}[\xi]$.

Throughout this book, we restrict the linear systems as systems, whose behavior admits a so called *kernel representation*. By this, we mean behaviors that can be expressed as the kernel of a suitable operator formed by a polynomial matrix $R(\frac{d}{dt})$, for continuous time systems, or $R(\sigma)$ for discrete time systems. The symbols $\frac{d}{dt}$ corresponds to the time-differentiation operator, while σ stands for forward unit time-shift.

$$(\sigma w)(k) := w(k+1),\ \forall k \in \mathbb{Z}. \tag{2.15}$$

Thus, by linear systems we mean systems that admit a representation of the form

$$R\left(\frac{d}{dt}\right) w = 0 \text{ or } R(\sigma)\, w = 0. \tag{2.16}$$

The class of continuous time linear systems with q variables is denoted by $\mathfrak{L}_{\mathrm{c}}^{\mathrm{q}}$, while its discrete time counterpart is denoted by $\mathfrak{L}_{\mathrm{d}}^{\mathrm{q}}$.

Since linear behaviors in $\mathfrak{L}_{\mathrm{c}}^{\mathrm{q}}$ are defined to be solutions of a system of linear differential equations, it is essential to clarify the concept of solution used in the representation. That is, we need to clarify when a trajectory $w : \mathbb{R} \to \mathbb{R}^{\mathrm{q}}$ is said to be a solution of

$$R\left(\frac{d}{dt}\right) w = 0,\ R \in \mathbb{R}^{\mathrm{g}\times\mathrm{q}}[\xi]. \tag{2.17}$$

Different concepts of solution will result in different sets of trajectories, and thus different behaviors. There are two solution concepts that appear in the literature (see, for example, Chapter 2 in [PW98]). They are the so called *strong solution* and *weak solution*.

Definition 2.9. The **strong solutions** to the differential equations (2.17) are infinitely differentiable functions (also denoted as $\mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^\mathtt{q})$ functions) such that (2.17) is satisfied in the usual sense.

In the textbook [PW98], strong solutions are not required to be infinitely differentiable, but rather sufficiently many times differentiable, depending on the differential operator $R\left(\frac{d}{dt}\right)$. Here we use a slightly different characterization, because it is mathematically more convenient.

Before we can define the concept of weak solutions, we need to introduce the class of *locally integrable functions* and *test functions*.

Definition 2.10. (cf. Definition 2.3.4 in [PW98]) A function $w : \mathbb{R} \to \mathbb{R}^\mathtt{q}$ is said to be **locally integrable** if for all $a, b \in \mathbb{R}$,

$$\int_a^b \|w(t)\| \, dt < \infty. \tag{2.18}$$

The symbol $\|\cdot\|$ denotes Euclidian norm on $\mathbb{R}^\mathtt{q}$. The class of locally integrable functions $w : \mathbb{R} \to \mathbb{R}^\mathtt{q}$ is denoted as $\mathfrak{L}_1^{\text{loc}}(\mathbb{R}, \mathbb{R}^\mathtt{q})$.

Definition 2.11. A function $w : \mathbb{R} \to \mathbb{R}^\mathtt{q}$ is said to be a **test function** if it is infinitely differentiable and has compact support. The class of test functions $w : \mathbb{R} \to \mathbb{R}^\mathtt{q}$ is denoted as $\mathfrak{D}(\mathbb{R}, \mathbb{R}^\mathtt{q})$.

Example 2.12. An example of a test function is $\phi : \mathbb{R} \to \mathbb{R}$,

$$\phi(t) = f(t) \cdot f(1-t), \tag{2.19}$$

where

$$f(t) := e^{-\frac{1}{t^2}}. \tag{2.20}$$

It can be verified that the test function ϕ is infinitely differentiable and has a compact support, namely $[0, 1]$.

Definition 2.13. Weak solutions to the differential equations (2.17) are locally integrable functions w that satisfy (2.17) in the distributional sense, i.e. for any $v \in \mathfrak{D}(\mathbb{R}, \mathbb{R}^\mathtt{g})$ the following relation holds.

$$\int_{-\infty}^{+\infty} \left(R^*\left(\frac{d}{dt}\right) v(t) \right)^{\mathrm{T}} w(t) \, dt = 0, \tag{2.21}$$

where $R^*(\xi) := R^{\mathrm{T}}(-\xi)$.

From the definitions, we can infer that strong solutions of (2.17) are also weak solutions. The class of weak solutions is an extension of the strong solutions, so that the behavior can capture non-smooth functions that are traditionally of interest in linear systems theory, such as the step function and ramp function.

Throughout this book, we associate the symbol $\mathfrak{L}_{\mathrm{c}}^{\mathrm{q}}$ to the class of behaviors with q variables, consisting of strong solutions of a kernel representation. When weak solution is meant, we use the notation $\bar{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$.

Example 2.14. Consider the single integrator system, whose behavior $\mathfrak{B} \in \mathfrak{L}_{\mathrm{c}}^{2}$ is described as

$$\mathfrak{B} := \left\{ (u, y) \mid \frac{dy}{dt} - u = 0 \right\}. \tag{2.22}$$

The input-output pair

$$u(t) = \begin{cases} 0, & t \leq 0, \\ 1, & t > 0, \end{cases} \tag{2.23}$$

$$y(t) = \begin{cases} 0, & t \leq 0, \\ t, & t > 0, \end{cases} \tag{2.24}$$

is only included in $\mathfrak{B}$ in the weak sense, since it only satisfies the differential equation in (2.22) in the distributional sense.

There are some results on the topological property of the space of weak solutions, which we shall use later in the book. These results have appeared in [PW98], and will be presented here without proofs. We begin by defining the notion of convergence in $\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$.

Definition 2.15. (cf. Definition 2.4.2 in [PW98]) A sequence $\{w_k\}$ of functions in $\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$ is said to **converge** to $w \in \mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$ in the sense of $\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$ if for all $a, b \in \mathbb{R}$,

$$\lim_{k \to \infty} \int_a^b \|w_k(t) - w(t)\| \; dt = 0. \tag{2.25}$$

With the notion of convergence above, we can express some topological properties of the space of weak solutions.

Theorem 2.16. (cf. Section 2.4 in [PW98]) Given any $R \in \mathbb{R}^{\mathrm{g} \times \mathrm{q}}[\xi]$, and denote the behaviors corresponding to the weak solutions and strong solutions of $R(\frac{d}{dt})w = 0$ as $\mathfrak{B}_{\mathrm{w}}$ and $\mathfrak{B}_{\mathrm{s}}$ respectively.
(a) The space $\mathfrak{B}_{\mathrm{w}}$ is **closed**. This means for any converging sequence $\{w_k\}$ in $\mathfrak{B}_{\mathrm{w}}$, its limit w is also an element of $\mathfrak{B}_{\mathrm{w}}$.
(b) The space $\mathfrak{B}_{\mathrm{s}}$ is **dense** in $\mathfrak{B}_{\mathrm{w}}$. This means any element $w \in \mathfrak{B}_{\mathrm{w}}$ is approachable as limit of a converging sequence $\{w_k\}$ in $\mathfrak{B}_{\mathrm{s}}$.

Notice that by definition of the space $\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$, the immediate value of a function at a given point in $\mathbb{R}$ is not important. This is because two functions

$w_1, w_2 \in \mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathtt{q}})$ are equal (in the sense of $\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathtt{q}})$) if

$$\int_a^b \|w_1(t) - w_2(t)\| \; dt = 0, \tag{2.26}$$

for any $a, b \in \mathbb{R}$. For this reason, in some cases, we are interested in the trajectories in $\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathtt{q}})$ with some continuity property.

A function $w : \mathbb{R} \to \mathbb{R}^{\mathtt{w}}$ is *left-continuous* if the limit $\lim_{t \uparrow 0} w(t)$ exists for all $t \in \mathbb{R}$. Denote the class of functions in $\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathtt{q}})$ that are also left-continuous as $\overrightarrow{\mathfrak{L}}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathtt{q}})$. The class of behaviors with q variables, consisting of left continuous weak solutions of a kernel representation is denoted as $\overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathtt{q}}$. The class of left-continuous weak solutions then lies between the class of strong solutions and weak solutions. The left-continuous weak solutions are also dense in the set of weak solutions. However, this space of solutions is not closed.

In general, for a given linear behavior $\mathfrak{B}$ in $\mathfrak{L}_{\mathrm{c}}^{\mathtt{q}}$, $\bar{\mathfrak{L}}_{\mathrm{c}}^{\mathtt{q}}$, $\overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathtt{q}}$ or $\mathfrak{L}_{\mathrm{d}}^{\mathtt{q}}$, there are infinitely many kernel representations that describe it. Consider the following examples.

Example 2.17. Take a dynamical system $\Sigma = (\mathbb{Z}, \mathbb{R}^2, \mathfrak{B})$, with $\mathfrak{B} \in \mathfrak{L}_{\mathrm{d}}^2$, where

$$\mathfrak{B} := \left\{ (w_1, w_2) \in \mathbb{Z} \to \mathbb{R}^2 \;\middle|\; \begin{bmatrix} 1 & 1-\sigma^2 \\ 0 & 1+\sigma \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = 0 \right\}. \tag{2.27}$$

We can compute that $\mathfrak{B}$ consists of trajectories (w_1, w_2), such that for all $k \in \mathbb{Z}$,

$$\begin{aligned} w_1(k) &= 0, \\ w_2(k) &= K \cdot (-1)^k, \end{aligned}$$

for some $K \in \mathbb{R}$. Notice that the following kernel representation

$$\mathfrak{B} := \left\{ (w_1, w_2) \in \mathbb{Z} \to \mathbb{R}^2 \;\middle|\; \begin{bmatrix} 1 & 0 \\ 0 & 1+\sigma \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = 0 \right\}, \tag{2.28}$$

also represents the same behavior.

Example 2.18. Take a dynamical system $\Sigma = (\mathbb{R}, \mathbb{R}^2, \mathfrak{B})$, with $\mathfrak{B} \in \mathfrak{L}_{\mathrm{c}}^2$, where

$$\mathfrak{B} := \left\{ (w_1, w_2) \in \mathbb{R} \to \mathbb{R}^2 \;\middle|\; \begin{bmatrix} 1 & 1-\frac{d^2}{dt^2} \\ 0 & 1+\frac{d}{dt} \\ 1 & -\frac{d^2}{dt^2} - \frac{d}{dt} \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = 0 \right\}. \tag{2.29}$$

Notice that the third differential equation is nothing but the difference between the first and the second. Hence this representation is somewhat nonminimal, as one of the rows can be removed without affecting the behavior. The formal definition of minimality for kernel representation will be given later in this section.

Equivalent kernel representations of a linear behavior are related by the so called *unimodular matrices*. A square polynomial matrix is a unimodular matrix if its determinant is a nonzero real number. Equivalently, a unimodular matrix is a square polynomial matrix, whose inverse exists as a polynomial matrix (which is also unimodular). Unimodular matrices have an important property described in the following lemma.

Lemma 2.19. Given any unimodular matrix $U \in \mathbb{R}^{\mathrm{q}\times\mathrm{q}}[\xi]$, the strong, weak, and left-continuous weak behaviors corresponding to the solution of

$$U\left(\frac{d}{dt}\right)w = 0, \tag{2.30}$$

and the $\mathfrak{L}_{\mathrm{d}}^{\mathrm{q}}$ behavior corresponding to the solution of

$$U(\sigma)\,w = 0, \tag{2.31}$$

consist of only the zero trajectory.

Proof. We have to show that if w satisfies (2.30) or (2.31), then w is the zero trajectory. For the case of the solution to (2.31) and the strong solution to (2.30), this fact is evident if we premultiply both equations with U^{-1}. For the case of the weak solution to (2.30), it follows from Theorem 2.16 and the fact that the strong behavior consists of only the zero trajectory. Obviously, the left-continuous part of the weak behavior consisting of only the zero trajectory consists of only the zero trajectory as well. □

As a consequence of this lemma, we can have the following theorem.

Theorem 2.20. (cf. Theorem 2.5.4 in [PW98]) Given any $R_1, R_2 \in \mathbb{R}^{\mathrm{g}\times\mathrm{q}}[\xi]$ such that there exists a unimodular U where $R_1 = UR_2$. Denote the behaviors corresponding to the weak, left-continuous weak and strong solutions of $R_i\left(\frac{d}{dt}\right)w = 0$ as $\mathfrak{B}_{\mathrm{w},i}$, $\mathfrak{B}_{\mathrm{l},i}$ and $\mathfrak{B}_{\mathrm{s},i}$ respectively, $i = 1, 2$. Also, denote the discrete time behavior corresponding to the solutions of $R_i(\sigma)\,w = 0$ as $\mathfrak{B}_{\mathrm{d},i}$, $i = 1, 2$. The following relations hold.

$$\mathfrak{B}_{\mathrm{w},1} = \mathfrak{B}_{\mathrm{w},2}, \tag{2.32}$$
$$\mathfrak{B}_{\mathrm{l},1} = \mathfrak{B}_{\mathrm{l},2}, \tag{2.33}$$
$$\mathfrak{B}_{\mathrm{s},1} = \mathfrak{B}_{\mathrm{s},2}, \tag{2.34}$$
$$\mathfrak{B}_{\mathrm{d},1} = \mathfrak{B}_{\mathrm{d},2.} \tag{2.35}$$

The previous theorem says that we can manipulate the kernel representation of an LTI behavior by premultiplying it with a unimodular matrix, while maintaining the behavior it represents. Manipulating kernel representations can lead to the so called *minimal kernel representation*.

A polynomial matrix $R \in \mathbb{R}^{\mathrm{g}\times\mathrm{q}}[\xi]$ is said to have *full row rank* if g≤q and one can select g columns of $R(\xi)$ to form a square polynomial matrix with nonzero determinant. It can be proven that R has full row rank if and only if there exists no polynomial row vector $v \in \mathbb{R}^{1\times\mathrm{g}}[\xi]$, such that $v(\xi)R(\xi) = 0$.

A kernel representation of an LTI behavior is minimal if the polynomial matrix $R(\xi)$ has full row rank. It means that the behavior cannot be represented as a kernel of a polynomial matrix with fewer rows than R. This holds for behaviors in $\mathfrak{L}_c^q$, $\bar{\mathfrak{L}}_c^q$, $\overrightarrow{\mathfrak{L}}_c^q$ or $\mathfrak{L}_d^q$. It can be proven that for any nonminimal representation (corresponding to nonfull row rank polynomial matrix $R \in \mathbb{R}^{g\times q}[\xi]$), we can always compute a unimodular matrix $U \in \mathbb{R}^{g\times g}[\xi]$ such that

$$R'(\xi) := U(\xi)R(\xi) \tag{2.36}$$

can be partitioned as

$$R'(\xi) = \begin{bmatrix} R''(\xi) \\ 0 \end{bmatrix}, \tag{2.37}$$

where $R''(\xi)$ is a full row rank polynomial matrix that gives a minimal kernel representation of the same behavior.

Theorem 2.21. (cf. Theorem 3.6.2 in [PW98]) Two polynomial matrices $R_1(\xi)$ and $R_2(\xi)$ in $\mathbb{R}^{g\times q}[\xi]$ represent the same behavior (in $\mathfrak{L}_c^q$, $\bar{\mathfrak{L}}_c^q$, $\overrightarrow{\mathfrak{L}}_c^q$ or $\mathfrak{L}_d^q$) if and only if there exists a unimodular matrix $U \in \mathbb{R}^{g\times g}[\xi]$ such that $R_1 = UR_2$.

Remark 2.22. In equation (2.36)-(2.37) we see that using unimodular matrix U, we can manipulate a polynomial matrix R to yield an upper full row rank matrix R''. In fact, we can also choose U such that R'' is not only full row rank but also *upper triangular* or *lower triangular*.

2.4 Discrete event systems

The next class of systems we shall cover in this book is the class of discrete event systems, particularly finite state automata and their languages. As in the case of LTI systems, the theory of finite state automata and their languages is well established, and there is an abundant literature on it. The author would only like to name a few, which have been used in preparing this book. They are [Büc89, CL99, HMU01].

Let us start by defining alphabets, strings and languages.

Definition 2.23. An **alphabet** $\mathbf{A}$ is a set of symbols. A **string** s over the alphabet $\mathbf{A}$ is a sequence of symbols in $\mathbf{A}$. An empty string is denoted by ε. A **language** over an alphabet $\mathbf{A}$ is a collection of finite-length strings of elements of $\mathbf{A}$.

Example 2.24. Let the alphabet $\mathbf{A}$ be $\{a, b, c\}$, then the following set

$$L = \{\varepsilon, a, aa, abab, acabb\}$$

is an example of a language over $\mathbf{A}$.

For a string s, the expression $|s|$ denotes the length of s. We then have $|\varepsilon| = 0$.

Since strings can be seen as partial functions from $\mathbb{Z}_+$ to the alphabet $\mathbf{A}$, a language over $\mathbf{A}$ is a behavior of type $(\mathbb{Z}_+, \mathbf{A})$. The set of all possible strings (including the empty string) of an alphabet $\mathbf{A}$ is denoted as $\mathbf{A}^*$, which is also called the *Kleene closure* of $\mathbf{A}$. Before we proceed to review some representations of the languages, we shall review some material about operations on strings and languages.

Strings over an alphabet can be concatenated to form longer strings. The concatenation of two strings s_1 and s_2, both elements of $\mathbf{A}^*$ is written as $s_1 s_2$, which is a string formed by merging the tail of s_1 with the head of s_2. For example, if $s_1 = abb$ and $s_2 = ccab$, then $s_1 s_2 = abbccab$. The empty string ε acts as the identity element for concatenation.

If the strings $s, t, u, v \in \mathbf{A}^*$ are such that $s = tuv$, then we say that t is a *prefix* of s, u is a *substring* of s, and v is a *suffix* of s.

Languages are basically sets. Hence the usual set operations, such as union, intersection, and difference apply for languages. Other than that, the following three operations are considered basic for languages [CL99].

concatenation Let L_a and L_b be languages over $\mathbf{A}$. The concatenation of L_a and L_b is defined as

$$L_a L_b := \{s \in \mathbf{A}^* \mid \exists (s_a, s_b) \in L_a \times L_b \text{ s.t. } s = s_a s_b\}. \tag{2.38}$$

This means any string in $L_a L_b$ can be written as a concatenation of a string in L_a and a string in L_b.

prefix-closure Let L be a language over $\mathbf{A}$. The prefix closure of L is defined as

$$\bar{L} := \{s \in \mathbf{A}^* \mid \exists r \in \mathbf{A}^* \text{ s.t. } sr \in L\}. \tag{2.39}$$

If $L = \bar{L}$, then L is said to be *prefix closed*.

Kleene closure Let L be a language over $\mathbf{A}$. The Kleene closure of L is defined as

$$L^* := \{\varepsilon\} \cup L \cup LL \cup LLL \cup \cdots . \tag{2.40}$$

Although languages can be considered as dynamical systems themselves, in this book we study them in relation with the automata that produce them. In our point of view, the automata are representations of the behaviors given by the languages.

Definition 2.25. A **finite state automaton** (FSA) A is a five-tuple $A = (X, E, T, X_{\mathrm{m}}, x_0)$, where X is a finite set of states, E is the set of events, $T \subset X \times E \times X$ is the transition relation, $X_{\mathrm{m}} \subset X$ is the set of marked states, and $x_0 \in X$ is the initial state.

The transition relation is a subset of $X \times E \times X$ such that $(x, a, x') \in T$ means the automaton will jump from state x to state x' with event a. The marked states[1] are states that have a special property, which will be discussed later. The initial state is the state in which the execution of the automaton starts.

[1]in the literatures, e.g. [HMU01], marked states are also called *final* states or *accepting* states.

Definition 2.26. A finite state automaton $A = (X, E, T, X_{\mathrm{m}}, x_0)$ is **deterministic** if for any $x \in X$ and $a \in E$, there can be at most one $x' \in X$ such that $(x, a, x') \in T$. A finite state automaton that is not deterministic is called **nondeterministic**.

Given a finite state automaton $A = (X, E, T, X_{\mathrm{m}}, x_0)$, a finite sequence $x_0 a_0 x_1 a_1 \cdots x_n a_n x_{n+1}$ is called a *run* or *execution* of A if for any $i \in \{0, 1, \cdots, n\}$ the following hold.

$$a_i \in E, \tag{2.41a}$$

$$x_i \in X, \tag{2.41b}$$

$$(x_i, a_i, x_{i+1}) \in T. \tag{2.41c}$$

Notice that an execution always starts from the defined initial state x_0. The execution $x_0 a_0 x_1 a_1 \cdots x_n a_n x_{n+1}$ is said to *terminate* at state x_{n+1}.

We can easily observe that the collection of all executions of A forms a language over $(X \cup E)$. However, in most cases, we are interested in the language consisting of strings formed by removing the states from the executions.

Definition 2.27. Given an FSA $A = (X, E, T, X_{\mathrm{m}}, x_0)$, the **language generated by** A, denoted by $L(A)$ consists of strings $a_0 a_1 \cdots a_n$ such that there exists an execution of A in the form of $x_0 a_0 x_1 a_1 \cdots x_n a_n x_{n+1}$. In other words, we form the strings in $L(A)$ by removing the symbols related to states in the executions of A. By definition, $L(A)$ is prefix closed.

Example 2.28. Consider a rather simple production line modeled by an automaton $P = (X, E, T, \{\texttt{start}\}, \texttt{start})$. The illustration of the production line and the automaton P can be seen in Figure 2.2. In this production line, raw material arrives at the processing unit and get processed, provided that the unit is not busy. If the unit is busy, the material has to wait until it is ready to be processed. After the process, the end product is fed out of the production unit. Here we assume that the initial state is `start`, which is also the only marked state. The language generated by the automaton consists of all strings of events starting from the initial state that the automaton can execute. Examples of the strings are (`arrival wait`), (`arrival process`), (`arrival process output arrival`), and so on.

Another kind of language associated to a finite state automaton A is its marked language, which is also called the language accepted by A.

Definition 2.29. Given an FSA $A = (X, E, T, X_{\mathrm{m}}, x_0)$, the **language accepted by** A, or the **marked language of** A is defined as the subset of $L(A)$ consisting of all the strings that terminate on a marked state. The marked language of A is denoted by $L_{\mathrm{m}}(A)$.

The motivation behind the concept of marked language is the following. It is often the case that we are not interested in all possible executions of A, but only in those with a certain property corresponding to the terminal state (i.e. the state where they terminate). States with this desirable property are the marked states.

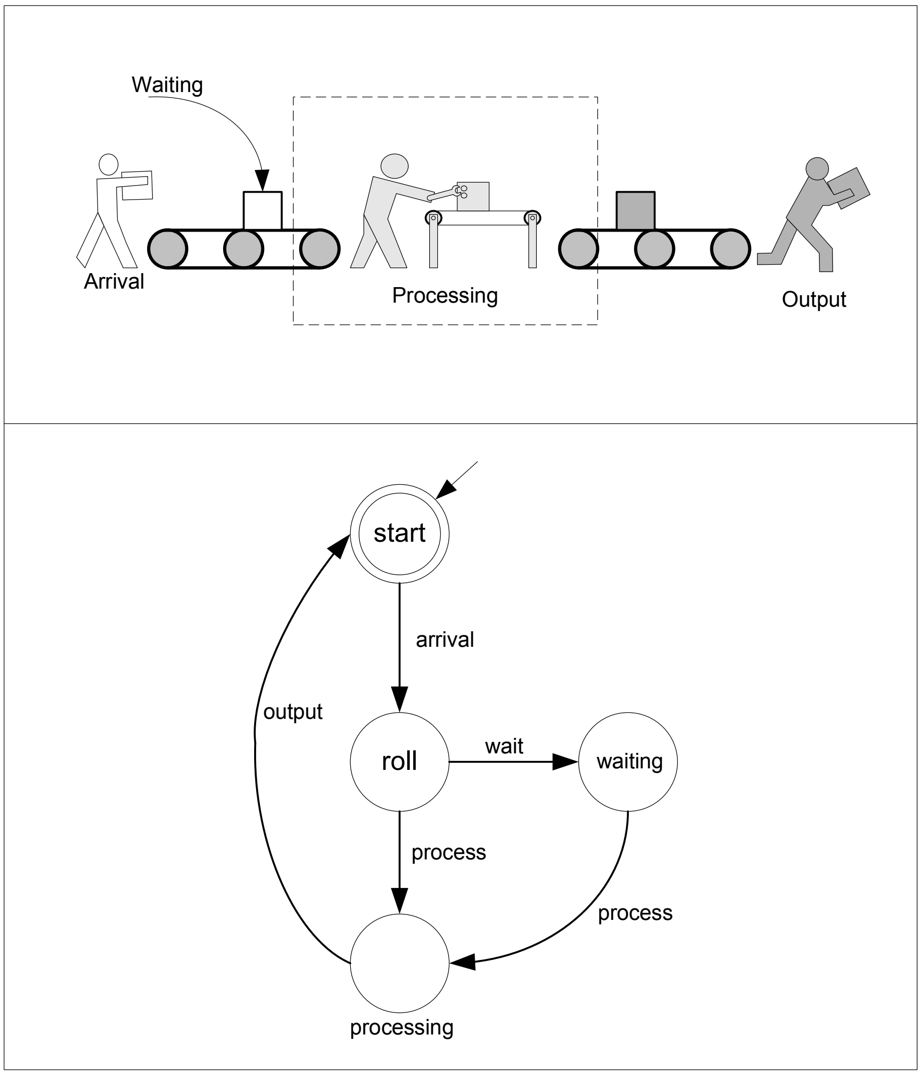


Figure 2.2: Illustration of Example 2.28. (Top) An illustration of the production line. (Bottom) An automaton that models it.

Consider again the automaton in Example 2.28. The state `start` is marked because any execution that terminates there represents a completed cycle of production. Therefore, the marked language $L_{\mathrm{m}}(P)$ of this automaton consists of strings corresponding to completed production cycle, with or without waiting.

Given a language, we cannot always find a finite state automaton that accepts it. The class of languages that can be accepted by finite state automata is called *regular languages*. Regular languages are type 3 languages in Chomsky's classification of languages. A larger class of languages is, for example, type 2 languages consisting of context-free languages, which are associated to finite state pushdown automata [HMU01]. In this book, we shall consider only regular languages.

Although we introduce the class of regular languages by mentioning its connection with finite state automata, they are also defined by the following property.

Theorem 2.30. (cf. Theorem 4.1 in [HMU01]) Given a regular language L, there exists an integer n such that every string $w \in L$, with $|w| \geq n$, can be broken into three strings $w = xyz$, where
(i) $y \neq \varepsilon$,
(ii) $|xy| \leq n$,
(iii) For all $k \geq 0$, the string $xy^k z \in L$.

Associated to regular languages, are the so called *regular expressions*. Regular expressions can be thought of as formulae that represent regular languages. To each regular expression E, we can uniquely associate a regular language $L(E)$. In fact, a language is regular if and only if it can be expressed as a regular expression. This result is known as the Kleene's theorem.

Given an alphabet $\mathbf{A}$, the set of regular expressions is recursively defined as follows.

1. For any symbol $a \in \mathbf{A}$, the regular expression a is associated with the language $\{a\}$. The expression ε is associated with $\{\varepsilon\}$, and $\emptyset$ with the empty language.

2. If E_1 and E_2 are regular expressions, then the regular expressions $E_1 + E_2$ and $E_1 \cdot E_2$ are defined as

$$L(E_1 + E_2) = L(E_1) \cup L(E_2), \tag{2.42}$$
$$L(E_1 \cdot E_2) = L(E_1)L(E_2). \tag{2.43}$$

3. If E is a regular expression, the regular expressions E^* and $\bar{E}$ are associated to the Kleene closure and prefix closure of $L(E)$, respectively.

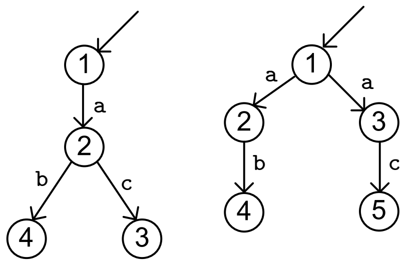
Notice that now we use the same notation $L()$ to denote the language generated by an automaton and the language associated with a regular expression. For brevity, hereafter we shall adopt the common practice of dropping the symbol $L()$ when we want to denote the language associated to the regular expression E and symbol E itself instead. Thus instead of writing $L(E)$, we simply write E. From the context, it shall be clear whether a regular expression is meant or the language associated to it.

Consider again Example 2.28. The generated language and the marked language of the automaton P can be written in terms of regular expressions as

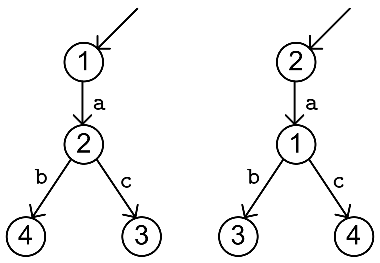
$$L(P) = \overline{(\texttt{arrival}(\texttt{wait.process} + \texttt{process})\texttt{output})^*}, \tag{2.44}$$

$$L_{\mathrm{m}}(P) = (\texttt{arrival}(\texttt{wait.process} + \texttt{process})\texttt{output})^*. \tag{2.45}$$

In our discussion about behaviors corresponding to discrete event systems so far, we have considered only generated language and marked language of finite state automata. Identifying the finite state automata with their languages, we cannot distinguish, for example, two automata that generate the same language. For nondeterministic automata, one may be interested in being able to distinguish between two automata with different branching property. For example, the following two automata generate the same language.



If we want to be able to distinguish between these two automata, then we have to enrich the information contained in the trajectories. Obviously, having trajectories that contain only the strings of events is not sufficient. One naive alternative is to include the state. This might be too strong, for example, we might not want to distinguish between the following two automata.



There are other alternatives. For example, we can consider the behavior as the collection of the *failure traces* of the automata [Hoa84, Lan92, HS96]. In the failure traces, in addition to the usual trace one has the information about the events that are not possible at the corresponding state. Another alternative is to consider the behavior as the collection of the *ready traces* of the automata [BBK87, BKO88]. In the ready traces, the additional information is about the events that are possible at the corresponding state. We are not going further into this issue, but the underlying message is that we can include more information in the trajectories if we want to be able to distinguish between systems that are otherwise identical.

2.5 Hybrid systems

In this section, we shall discuss hybrid systems from the behavioral theory point of view. Representations of hybrid systems in the literature are plentiful. Among them are hybrid automata [ACHH93, Alu95, Hen96, SS00], hybrid process algebra [CR03, BKU04, Cui04, BKU05], hybrid input-output automata [LSV01, LSV03], general hybrid dynamical systems [BBM98], hybrid behavioral automata [JSS03]. Despite of the varieties, most representations share the same features of hybrid systems, namely

1. The system has continuous time dynamics that can depend on discrete states. The discrete states are called *locations*.
2. Transitions between discrete states are labelled in a similar manner as in automata.
3. Transitions from one discrete state to another can occur when certain conditions called the *guards* are satisfied.
4. Transitions must occur when a condition, called *invariant condition* of a location is violated.
5. The transitions can not only change the continuous dynamics, but also impose a *reset map* on the continuous state.

In this book, we shall represent hybrid systems as hybrid behavioral automata.

Remark 2.31. The reader who knows about the definition of hybrid behavioral automata in [JSS03] will notice that there is a slight difference between the definition in this book and that in that in [JSS03]. The difference is that we do not distinguish between active and passive transitions. Distinguishing between active and passive transitions can arguably make system modeling more convenient, at the cost of more complicated interconnection rules. In the spirit of the behavioral approach, in this book we opt for simpler interconnection rules, and thus drop the distinction between active and passive transitions.

A **hybrid behavioral automaton** A is a 6-tuple $(L, W, \mathfrak{B}, E, T, Inv)$, where

- L is the set of locations or discrete states,
- W is the set of continuous variables taking values in $\mathbb{W}$,

- $\mathfrak{B}$ maps each location to its continuous behavior. A behavior is a subset of $\mathbb{W}^{\mathbb{R}_+}$.
- E is the set of events/labels,
- T is the set of transitions. Each transition is given as a 5-tuple $(l, \mathbf{a}, l', G, R)$. The triple $(l, \mathbf{a}, l')$ is a subset of $L \times E \times L$, where l is the origin location, $\mathbf{a}$ is the label of the transition, l' is the target location. $G := (\gamma, g)$ is the guard of the transition, where $\gamma : \mathfrak{B}(l) \times \mathbb{R}_+ \to$range$(\gamma)$ and $g \subset$range(γ), and $R : \mathfrak{B}(l) \times \mathbb{R}_+ \to 2^{\mathfrak{B}(l')}$ is the reset map of the transition.
- Inv is the invariant condition for the automata. It maps each location $l \in L$ to a pair $Inv(l) := (\nu, V)$, where $\nu : \mathfrak{B}(l) \times \mathbb{R}_+ \to$range$(\nu)$ and $V \subset$range(ν).

The guard of a transition can be regarded as a Boolean valued function that takes a continuous trajectory w and a time instant τ and returns true if it is possible to execute the transition at time τ, provided that the continuous dynamics follows the trajectory w. The reset map of a transition resets the trajectory of the continuous dynamic to another one that belongs to the behavior of the target location.

The invariant of a location can also be regarded as a Boolean valued function that takes a continuous trajectory w and a time instant τ. It returns true if w satisfies the invariant condition of the location at time τ. As soon as the invariant condition is violated, a transition must occur. If there is no enabled transition (no guard is satisfied) then the system deadlocks. This means the evolution of the trajectory cannot continue beyond the time τ. When we model a hybrid dynamical system, a deadlock is typically not desired.

The guard G and the invariant $Inv(l)$ as introduced above are instances of *dynamic predicates*. In general, a dynamic predicate is a pair $C := (\psi, \Psi)$,

$$\psi : \mathfrak{B} \times \mathbb{R}_+ \to \text{range}(\psi), \tag{2.46a}$$
$$\Psi \subset \text{range}(\psi). \tag{2.46b}$$

$\mathfrak{B}$ signifies a behavior over the time axis $\mathbb{R}_+$. A pair $(w, t) \in \mathfrak{B} \times \mathbb{R}_+$ is said to satisfy the dynamic predicate C if $\psi(w, t) \in \Psi$. We denote it by $(w, t) \models C$. The negation of this statement is denoted by $(w, t) \not\models C$.

We assume that the maps γ, R, and ν are *past-induced* and *local*. A map $x : \mathfrak{B} \times \mathbb{R}_+ \to X$ is past-induced if for any w_1 and w_2 in $\mathfrak{B}$ and $\tau \in \mathbb{R}_+$, the following implication holds.

$$\left(w_1(t)|_{t \le \tau} = w_2(t)|_{t \le \tau} \right) \implies (x(w_1, \tau) = x(w_2, \tau)) \,. \tag{2.47}$$

A map $x : \mathfrak{B} \times \mathbb{R}_+ \to X$ is local if for any w_1 and w_2 in $\mathfrak{B}$ and $\tau \in \mathbb{R}_+$, the following implication holds. If there exists an interval $B(\tau, r)$ of any length r around τ such that $\left(w_1(t)|_{t \in B(\tau,r)} = w_2(t)|_{t \in B(\tau,r)} \right)$ then $(x(w_1, \tau) = x(w_2, \tau))$. Examples of local maps are functions of $w(t)$ and their time derivatives.

We shall now describe the execution of a hybrid behavioral automaton. Our description of the execution is similar to the approach in [SS00]. To describe the execution of a hybrid system, we need to define a suitable time axis. In this case, we take $\mathbb{R}_+ \times \mathbb{Z}_+$ as the hybrid time axis. A hybrid execution is then a partial function defined on the time axis. First, we define the total ordering $>$ on the hybrid time axis.

Definition 2.32. Given two pairs (t_1, n_1) and (t_2, n_2) elements of $\mathbb{R}_+ \times \mathbb{Z}_+$. We define $(t_1, n_1) < (t_2, n_2)$ if $t_1 < t_2$, or $t_1 = t_2$ and $n_1 < n_2$. This ordering is commonly called **lexicographic** ordering.

A hybrid execution can be thought of as a trajectory of type $(\mathbb{R}_+ \times \mathbb{Z}_+, L \times \big(\mathbb{W} \cup (E \times \mathbb{W}^{\mathbb{R}_+})\big)$ in the following sense. We associate each trajectory with the subset of $(\mathbb{R}_+ \times \mathbb{Z}_+)$, on which it is defined. If ζ is a hybrid execution, we denote the subset as T_ζ. We assume that for any hybrid execution ζ, its time axis T_ζ is structured such that

- $(0, 0) \in T_\zeta$,
- For any $t \in \mathbb{R}_+$ and $n \in \mathbb{Z}_+$, if $(t, n) \in T_\zeta$ then $(t, n') \in T_\zeta$ for all nonnegative integer $n' < n$,
- For any $t \in \mathbb{R}_+$, if $(t, 0) \in T_\zeta$ then $(t', 0) \in T_\zeta$ for all nonnegative real $t' < t$.

Given a hybrid execution ζ with its time axis T_ζ, we define the *span* of the time axis as

$$|T_\zeta| := \sup\{t \in \mathbb{R}_+ \mid (t, 0) \in T_\zeta\}. \tag{2.48}$$

We then define the *event multiplicity* of the time axis at a certain time $t \in \mathbb{R}_+$, where $0 \leq t \leq |T_\zeta|$ as

$$N_\zeta(t) := \max\{n \in \mathbb{Z}_+ \mid (t, n) \in T_\zeta\}. \tag{2.49}$$

We always assume that the event density of 0 is 0. The set of *event times* of T_ζ is defined as

$$\varepsilon_\zeta := \{t \in \mathbb{R}_+ \mid t \leq |T_\zeta| \text{ and } N_\zeta(t) > 0\}. \tag{2.50}$$

By the assumption above, 0 is not an event time. Moreover, we assume that the set of event times is countable and has nonzero infimum. We want to include only the executions where there is a finite time interval between the start of the evolution at time 0 to the first event time.

For any time instant in T_ζ, the execution ζ takes its value according to the following table.

Time instant	ζ takes value in
$(t, 0) \in T_\zeta$	$L \times \mathbb{W}$
$(t, n) \in T_\zeta, n > 0$	$L \times E \times \mathbb{W}^{\mathbb{R}_+}$

Notice that the second row of the table indicates that at the event times, the trajectory takes value in a function space rather than in a the signal space. This is related to the discrete transitions, as we shall discuss below (particularly conditions (c3) and (c4)).

An execution ζ belongs to the set of executions of the hybrid behavioral automaton $A = (L, W, \mathfrak{B}, E, T, Inv)$ if the following conditions are satisfied.

(c1) The location is a piecewise constant function of time. It must constant between event times.

(c2) Denote the smallest event time in ε_ζ as τ_1. There is $l \in L$ and $w \in \mathfrak{B}(l)$ such that

$$\zeta(t,0) = (l, w(t)),\ 0 \le t \le \tau_1, \tag{2.51}$$

$$(w(t),t) \models Inv(l),\ 0 \le t \le \tau_1. \tag{2.52}$$

(c3) For any event time $\tau \in \varepsilon_\zeta$, denote $\zeta(\tau, n) =: (l_n, \mathbf{a}_n, w_n)$, then for any n such that $0 < n \le N_\zeta(\tau)$ there should exists a transition $\delta_n = (l_{n-1}, \mathbf{a}_n, l_n, G_n, R_n) \in T$ such that

$$(w_{n-1}, \tau) \models G_n, \tag{2.53}$$

$$w_n \in R_n(w_{n-1}, \tau), \tag{2.54}$$

$$(w_n, \tau) \models Inv(l_n), \tag{2.55}$$

with the convention that l_0 is the location immediately before the event time τ and w_0 is the continuous trajectory in the interval before the event time τ. Notice that since this interval can be of finite length, there could be a technical difficulty as the guards, the reset maps and the invariants need the whole continuous trajectory as its argument. However, this problem is averted by requiring that they are local.

(c4) After an event time ε_1 until the next event time (if exists) ε_2, denote the hybrid execution at the last event time $\zeta(\varepsilon_1, N_\zeta(\varepsilon_1)) := (l, \mathbf{a}, w)$. Then for any t such that $\varepsilon_1 < t \le \varepsilon_2$,

$$\zeta(t,0) = (l, w(t)), \tag{2.56}$$

$$(w(t),t) \models Inv(l). \tag{2.57}$$

In words, the executions of the hybrid behavioral automaton can be described as follows. Every execution starts in a particular location, say l, and proceed with a continuous trajectory that satisfies the invariant condition in l. Whenever there is a transition, whose guard is satisfied, a transition can occur. When a transition occur, the location changes, say to l', and also the continuous trajectory is reset to another one compatible with the reset map and the invariant condition of the new location. It can also happen that immediately after the transition, the trajectory satisfies a guard of a transition in the new location. This is what happens at event times.

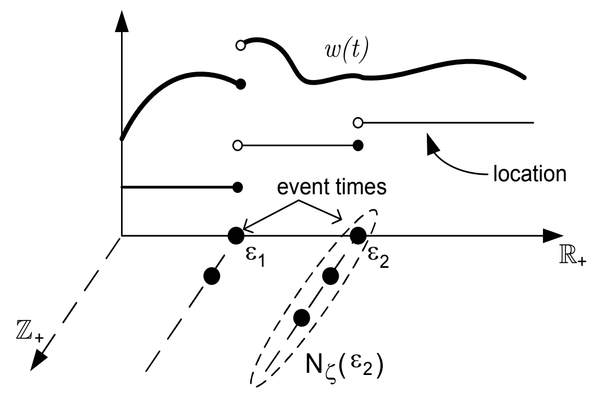


Figure 2.3: An illustration of the hybrid trajectory and hybrid time axis.

Refer to Figure 2.3 for an illustration of a hybrid execution and hybrid time axis. In this figure, the hybrid time axis ($\mathbb{R}_+ \times \mathbb{Z}_+$) is depicted as the base plane. The trajectory of the location is shown to be constant between event times. The concept of event times and event density are also shown. However, the part of of trajectory at event times is not drawn, so that the figure is not overcrowded.

The behavior of a hybrid system represented by a hybrid behavioral automaton A, denoted by $L(A)$, is the collection of *hybrid trajectories* obtained by removing the information about the location from its hybrid executions. Notice that this is analogous to what we have done with discrete-event systems. We first define executions of finite state automata, and then define the generated language as the collection of strings obtained by removing the state from the execution. The behavior of A is a behavior of type $(\mathbb{R}_+ \times \mathbb{Z}_+, \left(\mathbb{W} \cup (E \times \mathbb{W}^{\mathbb{R}_+})\right))$.

Example 2.33. Consider a room with an air conditioner. We model the temperature inside the room with a hybrid behavioral automaton $A_1 = (L_1, W_1, \mathfrak{B}_1, E_1, T_1, Inv_1)$. We use the subscript index 1 because later we are going to use this automaton again in another example. The set of locations $L_1 = \{\texttt{with}, \texttt{without}\}$. The location `with` represents the temperature dynamics inside the room when the air conditioner is on. When it is off, the temperature dynamics is represented by the location `without`. The set of continuous variable(s) $W_1 = \{\mathtt{x}\}$ taking value in $\mathbb{R}$. The variable $\mathtt{x}$ represents the temperature in the room. The behaviors in each location are given as

$$\mathfrak{B}_1(\texttt{with}) = \left\{ x \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}) \mid \frac{dx}{dt} = (10 - x) \right\}, \tag{2.58}$$

$$\mathfrak{B}_1(\texttt{without}) = \left\{ x \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}) \mid \frac{dx}{dt} = (30 - x) \right\}. \tag{2.59}$$

The set of labels/events $E_1 = \{\texttt{on}, \texttt{off}\}$. The set of transitions $T_1 = \{\delta_{11}, \delta_{12}\}$

where

$$\delta_{11} := (\texttt{with}, \texttt{off}, \texttt{without}, \textbf{true}, R_{11}), \tag{2.60}$$

$$\delta_{12} := (\texttt{without}, \texttt{on}, \texttt{with}, \textbf{true}, R_{12}). \tag{2.61}$$

Here the guards are given as the dynamic predicate **true**, which is always true for any trajectory at any time. If we want to express it in a formal way as in (2.46), the guards can be written as a pair (ψ, Ψ) where Ψ is a singleton and is the image of ψ. The reset maps R_{11} and R_{12} are defined as

$$R_{11}(x,t) = \{x' \in \mathfrak{B}_1(\texttt{without}) \mid x'(t) = x(t)\},\ \forall x \in \mathfrak{B}_1(\texttt{with}), t \in \mathbb{R}_+, \tag{2.62}$$

$$R_{12}(x,t) = \{x' \in \mathfrak{B}_1(\texttt{with}) \mid x'(t) = x(t)\},\ \forall x \in \mathfrak{B}_1(\texttt{without}), t \in \mathbb{R}_+. \tag{2.63}$$

The reset map R_{11} resets any temperature trajectory in the location `with` to a trajectory in `without`, that has the same temperature value. Thus, there is no jump in the value of the trajectories. The reset map R_{12} does the opposite action. The transitions δ_{11} and δ_{12} correspond to the events of the air conditioner being turned on and off respectively.
The invariant condition Inv_1 for both locations is satisfied if the temperature is between 10 and 30 degrees. Thus $10 \leq x(t) \leq 30$.
The executions of this hybrid behavioral automaton are those corresponding trajectories exponentially converging 10 or 30 depending on the location. At any time, the location may change due to a transition and thus changing the point of convergence of the temperature. Notice that since the guards are always enabled, it is also possible to have multiple but finitely many transitions at a particular event time.
This model describes a rather irregular air conditioning system, as the temperature can fluctuate freely between 10 and 30 degrees. Also, the air conditioner can be switched on and off arbitrarily. In the next chapter, when we discuss interconnection, we shall come back to this example and introduce some kind of control to regulate the system.

2.6 Summary

In this chapter we have set up a foundation for further discussion in this book. We begin by introducing the concept of systems as behaviors and explaining about key ingredients, such as trajectories, time axis, the type of behaviors. Subsequently, we elucidate how systems belonging to several classes, namely linear time invariant systems, discrete event systems and hybrid systems can fit into the behavioral framework. This is done by pointing out what a behavior is for these classes of systems.

In the following chapters, we shall discuss further concepts in behavioral systems theory in a general way, and link them to various classes of systems.

3

Interconnection of behaviors

"Everything is vague to a degree you do not realize till you have tried to make it precise." - Bertrand Russell.

3.1 Full interconnection

In this chapter we shall discuss the interconnection of behaviors. First, let us make it clear what we mean by interconnection. Generally speaking, when two behaviors are interconnected, the laws that define them are superimposed so that the resulting behavior is a collection of trajectories compatible with both behaviors. In what follows, we shall define interconnection in a mathematically precise way.

We start with the so called *full interconnection*. A full interconnection is an interconnection between behaviors of the same type.

Definition 3.1. Given two behaviors $\mathfrak{B}_1$ and $\mathfrak{B}_2$ of the same type, the **full interconnection** between them is denoted and defined as follows.

$$\mathfrak{B}_1 \parallel \mathfrak{B}_2 := \mathfrak{B}_1 \cap \mathfrak{B}_2. \tag{3.1}$$

So far, the reason behind the term 'full' in 'full interconnection' is not so apparent. Actually, the term is used for historical reasons, from its use in LTI systems. Recall that for LTI systems, the type of a behavior is determined by the number of variables involved in the description. Thus, when two LTI behaviors of the same type are interconnected, all the variables are involved in the interconnection. Hence the term 'full'.

Notice that by construction, the full interconnection is both commutative and associative. That is, for any behaviors of the same type $\mathfrak{B}_i$, $i \in \{1, 2, 3\}$, the following holds.

$$\mathfrak{B}_1 \parallel \mathfrak{B}_2 = \mathfrak{B}_2 \parallel \mathfrak{B}_1, \tag{3.2}$$

$$\mathfrak{B}_1 \parallel (\mathfrak{B}_2 \parallel \mathfrak{B}_3) = (\mathfrak{B}_1 \parallel \mathfrak{B}_2) \parallel \mathfrak{B}_3. \tag{3.3}$$

Having defined a notion of interconnection, we may then ask the following question. In the previous chapter we have been discussing about identifying systems with behaviors. Now we have a mathematical concept called interconnection defined for behaviors. How does behaviors interconnection relate to systems interconnection? We shall address this question by reviewing what interconnection means for the classes of systems we have covered in the previous chapter.

3.1.1 Linear time invariant systems

For LTI systems in classes $\mathfrak{L}_{\mathtt{d}}^{\mathtt{q}}$, $\mathfrak{L}_{\mathtt{c}}^{\mathtt{q}}$, $\overrightarrow{\mathfrak{L}}_{\mathtt{c}}^{\mathtt{q}}$ and $\bar{\mathfrak{L}}_{\mathtt{c}}^{\mathtt{q}}$, interconnecting two behaviors means superimposing the linear differential or difference equations governing each of the systems. Thus, if two LTI behaviors $\mathfrak{B}_1$ and $\mathfrak{B}_2$ represented by the kernel representation

$$\mathfrak{B}_1 := \left\{ w \mid R_1 \left(\frac{d}{dt} \right) w = 0 \right\} \text{ or } \mathfrak{B}_1 := \{ w \mid R_1 (\sigma) \, w = 0 \} , \tag{3.4}$$

$$\mathfrak{B}_2 := \left\{ w \mid R_2 \left(\frac{d}{dt} \right) w = 0 \right\} \text{ or } \mathfrak{B}_2 := \{ w \mid R_2 (\sigma) \, w = 0 \} , \tag{3.5}$$

are interconnected, then the resulting behavior $\mathfrak{B} := \mathfrak{B}_1 \parallel \mathfrak{B}_2$ is given by the kernel representation

$$\mathfrak{B} := \left\{ w \mid \begin{bmatrix} R_1 \\ R_2 \end{bmatrix} \left(\frac{d}{dt} \right) w = 0 \right\} \text{ or } \mathfrak{B} := \left\{ w \mid \begin{bmatrix} R_1 \\ R_2 \end{bmatrix} (\sigma) \, w = 0 \right\} . \tag{3.6}$$

Obviously, $\mathfrak{B} := \mathfrak{B}_1 \cap \mathfrak{B}_2$.

As a side remark, the kernel representation given in (3.6) is not necessarily minimal, even if (3.4) and (3.5) are minimal kernel representations.

Example 3.2. Think of MIMO feedback control interconnection between the plant, whose behavior

$$\mathcal{P} := \left\{ (u, y) \mid N \left(\frac{d}{dt} \right) u - D \left(\frac{d}{dt} \right) y = 0 \right\} . \tag{3.7}$$

Here $N(\xi) \in \mathbb{R}^{\mathtt{d} \times \mathtt{n}}[\xi]$ and $D(\xi) \in \mathbb{R}^{\mathtt{d} \times \mathtt{d}}[\xi]$, where $\mathtt{n}$ and $\mathtt{d}$ are the dimension of the input and output variables respectively. This is a kernel representation of a linear system, whose transfer function is $H(s) = D^{-1}(s) N(s)$, assuming that $D^{-1}(s)$ exists as a rational polynomial matrix (that is, the determinant of $D(\xi)$ is a nonzero polynomial). In a similar fashion, a feedback controller with static gain $u = Ky$ can be expressed as a behavior

$$\mathcal{C} := \{ (u, y) \mid u - Ky = 0 \} . \tag{3.8}$$

Notice that $\mathcal{P}$ and $\mathcal{C}$ have the same type, namely $(\mathbb{R}, \mathbb{R}^{\mathtt{n}+\mathtt{d}})$. The closed-loop system is given by the behavior interconnection $\mathcal{P} \parallel \mathcal{C}$, whose kernel representation is

$$\mathcal{P} \parallel \mathcal{C} = \left\{ (u, y) \mid \begin{bmatrix} N \left(\frac{d}{dt} \right) & -D \left(\frac{d}{dt} \right) \\ I & -K \end{bmatrix} \begin{bmatrix} u \\ y \end{bmatrix} = 0 \right\} . \tag{3.9}$$

Remark 3.3. Notice that interconnection of linear systems can be seen as sharing of variables. This point of view is quite natural if one thinks of interconnection of physical systems. Consider, for example, an electric circuit. In an electric circuit, components that are serially connected can be thought of as sharing the same electric current. Similarly, components in a parallel interconnection share the same voltage.

3.1.2 Discrete event systems

Behavior interconnection for discrete event systems corresponds to synchronization of the representing automata.

Before we proceed to define automata synchronization, recall that the type of the behavior corresponding to an automaton A is given by $(\mathbb{Z}_+, E)$, where E is the set of events defined for the automaton. Therefore, since we are discussing interconnection of behaviors with the same type, synchronization is also going to be of automata with the same set of events.

Definition 3.4. Given two automata with the same set of events $A_i = (X_i, E, T_i, X_\mathrm{m}, x_{0i})$, $i = 1, 2$. The **synchronization** of these two automata is an automaton $A = \{X, E, T, X_\mathrm{m}, x_0\}$ such that

$$X = X_1 \times X_2, \tag{3.10a}$$
$$T = \{((x_1, x_2), \mathbf{a}, (x_1', x_2')) \in X \times E \times X \mid (x_i, \mathbf{a}, x_i') \in T_i,\ i = 1, 2.\}, \tag{3.10b}$$
$$X_\mathrm{m} = X_{\mathrm{m}1} \times X_{\mathrm{m}2}, \tag{3.10c}$$
$$x_0 = (x_{01}, x_{02}). \tag{3.10d}$$

We denote the synchronization as $A = A_1 \parallel A_2$.

The interpretation behind the definition is that when two automata are synchronized, an event can take place only if both automata can execute that event simultaneously. Also, the automata must start at their respective initial states, and a marked state in the combined automaton is a combination of marked states in each automaton.

The synchronization operation that we introduce here coincides with the *product* and *parallel composition* of automata with the same set of events [CL99].

The correspondence between behavior interconnection and automata synchronization is given in the following theorem.

Theorem 3.5. Given two automata with the same set of events $A_i = (X_i, E, T_i, X_{\mathrm{m}i}, x_{0i})$, $i = 1, 2$. We synchronize these automata to form $A = A_1 \parallel A_2$. Let $L(\cdot)$ and $L_\mathrm{m}(\cdot)$ denote the generated and marked language of an automaton, then
(i) $L(A) = L(A_1) \cap L(A_2)$, and
(ii) $L_\mathrm{m}(A) = L_\mathrm{m}(A_1) \cap L_\mathrm{m}(A_2)$.

Proof. $(L(A) \subset L(A_1) \cap L(A_2))$ Take any string $s \in L(A)$. Suppose that $s = a_1 a_2 \cdots a_n$. By definition, there is an execution $(x_{01}, x_{02}) a_1 (x_{11}, x_{12})\ a_2 (x_{21}, x_{22}) \cdots$

$a_n(x_{n1}, x_{n2})$ by the automaton A, where $x_{11}, x_{21} \cdots x_{n1} \in X_1$ and $x_{12}, x_{22} \cdots x_{n2} \in X_2$. This implies that $x_{01}a_1\ x_{11}a_2x_{21} \cdots a_nx_{n1}$ and $x_{02}a_1\ x_{12}a_2x_{22} \cdots a_nx_{n2}$ are executions of A_1 and A_2 respectively. Therefore $s = a_1a_2 \cdots a_n$ is in $L(A_1)$ and $L(A_2)$.

$(L(A) \supset L(A_1) \cap L(A_2))$ Take any string $s \in L(A_1) \cap L(A_2)$. Suppose that $s = a_1a_2 \cdots a_n$. By definition, there is an execution $x_{01}a_1\ x_{11}a_2x_{21} \cdots a_nx_{n1}$ of A_1 and $x_{02}a_1\ x_{12}a_2x_{22} \cdots a_nx_{n2}$ of A_2. It follows that $(x_{01}, x_{02})a_1\ (x_{11}, x_{12})\ a_2(x_{21}, x_{22}) \cdots a_n(x_{n1}, x_{n2})$ is an execution of A, and thus $s \in L(A)$.

$(L_\mathrm{m}(A) = L_\mathrm{m}(A_1) \cap L_\mathrm{m}(A_2))$ Similar as the two parts above, with an additional requirement that the terminal states of the executions are marked. $\square$

Example 3.6. Refer to Figure 3.1. We model a data buffer with the automaton on the top part of the figure. Denote this automaton as $A_1 = (\{0, 1, \mathrm{E}\}, \{\texttt{write}, \texttt{pop}\}, T_1, \{\mathrm{E}\}, 0)$, with T_1 the set of transitions according to the figure. The numbered states of A_1 indicate whether the buffer is full. Thus, 0 is empty and 1 is full. The marked state E is the error state which is reached when an attempt is made to write to a full buffer. Therefore, the language generated by A_1, $L(A_1)$, is the collection of all possible strings of events given executed by A_1, which is given by the following regular expression

$$L(A_1) = \left(\overline{\texttt{write.pop} + \texttt{write.write.write}^*\texttt{.pop}}\right). \tag{3.11}$$

The marked language of A_1, $L_\mathrm{m}(A_1)$ is the collection of all strings of events terminating in the error state E, given by the regular expression

$$L_\mathrm{m}(A_1) = (\texttt{write.pop})^*\ \texttt{write.write.write}^*\ \left(\texttt{pop}\,(\texttt{write.pop})^*\,\texttt{.write.write}\right)^*. \tag{3.12}$$

Because of this interpretation, in this example we are interested in the marked language. Further, suppose that there are two kinds of user. A well-behaved user writes a data and always pops the data before writing again. An arbitrary user attempts to write or pop the data arbitrarily, without any rule. The well-behaved user is modeled as an automaton $A_2 = (\{0, 1\}, \{\texttt{write}, \texttt{pop}\}, T_2, \{0, 1\}, 0)$, which is given as the bottom left figure. The arbitrary user is modeled as an automaton $A_3 = (\{0\}, \{\texttt{write}, \texttt{pop}\}, T_3, \{0\}, 0)$, which is given as the bottom right figure.
We can see that

$$L_\mathrm{m}(A_2) = \overline{(\texttt{write.pop})^*}, \tag{3.13}$$
$$L_\mathrm{m}(A_3) = (\texttt{write} + \texttt{pop})^*. \tag{3.14}$$

The use of the buffer by the user is modeled as synchronization. Thus the automata

$$A_4 := A_1 \parallel A_2 \text{ and } A_5 := A_1 \parallel A_3$$

model the well-behaved user and the arbitrary user using the buffer respectively. Notice that all states of the user are marked. This is because an error is made if

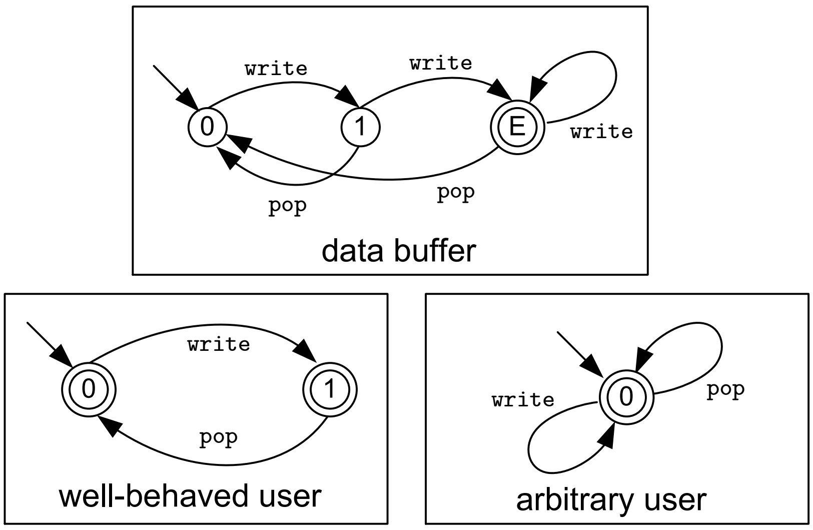


Figure 3.1: The automata discussed in Example 3.6.

the buffer reaches the error state regardless of the state of the user. Using Theorem 3.5, we obtain

$$L_{\mathrm{m}}(A_4) = L_{\mathrm{m}}(A_1) \cap L_{\mathrm{m}}(A_2) = \emptyset, \tag{3.15}$$

$$L_{\mathrm{m}}(A_5) = L_{\mathrm{m}}(A_1) \cap L_{\mathrm{m}}(A_3) = L_{\mathrm{m}}(A_1) \neq \emptyset. \tag{3.16}$$

This result tells us formally that a well-behaved user will never reach the error state, since there is no string of events terminating on a marked state. However, the arbitrary user may reach the error state. This, of course, confirms our intuition.

3.1.3 Hybrid systems

Behavior interconnection for hybrid systems can also be seen as synchronization of the representing hybrid behavioral automata (HBA).

As we have seen in the previous chapter, the type of a hybrid system represented with a hybrid behavioral automaton (HBA) is determined by the continuous variables and the set of labels defined for the automaton. Hence we shall now define synchronization of two HBA with the same set of continuous variables and labels.

Definition 3.7. Given two hybrid behavioral automata with the same set of continuous variables and events $A_i = (L_i, W, \mathfrak{B}_i, E, T_i, Inv_i)$, $i = 1, 2$. The **synchronization** of these two automata is a hybrid behavioral automaton $A = (L, W, \mathfrak{B}, E, T, Inv)$ such that the following holds.

(i) $L = L_1 \times L_2$.
(ii) $\mathfrak{B}(l_1, l_2) = \mathfrak{B}_1(l_1) \parallel \mathfrak{B}_2(l_2)$, for all $l_1 \in L_1$ and $l_2 \in L_2$.
(iii) If for $l_1 \in L_1$ and $l_2 \in L_2$ we denote $Inv_1(l_1) := (\nu, V)$ and $Inv_2(l_2) := (\iota, I)$, then $Inv(l_1, l_2) = ((\nu, \iota), V \times I)$. We denote this by $Inv(l_1, l_2) = Inv_1(l_1) \wedge Inv_2(l_2)$.
(iv) The set of transitions T consists of all transition $\delta = ((l_1, l_2), \mathbf{a}, (l_1', l_2'), G, R)$ such that there exist $\delta_1 = (l_1, \mathbf{a}, l_1', G_1, R_1) \in T_1$ and $\delta_2 = (l_2, \mathbf{a}, l_2', G_2, R_2) \in T_2$ such that $G = G_1 \wedge G_2$ and $R(w, t) = R_1(w, t) \cap R_2(w, t)$ for all $w \in \mathfrak{B}(l_1, l_2)$ and $t \in \mathbb{R}_+$.

The interpretation for this definition is the following. Basically the discrete part of the dynamics is synchronized in a similar fashion to the discrete event systems. This can be seen from the fact that we only allow a transition if the two HBA are ready to perform a transition with the same label. The continuous part of the dynamics is synchronized in a similar fashion to the LTI systems, in the sense that we only allow a trajectory if it is accepted by both automata. Invariants are combined to make sure that a trajectory can continue its evolution in one location only if it satisfies the invariant conditions associated with both automata. In a similar way, guards and resets are combined to make sure that a transition and a reset are possible if they are compatible with each of the automata.

As in the case with discrete event systems, we can prove a theorem analogous to Theorem 3.5.

Theorem 3.8. Given two hybrid behavioral automata with the same set of continuous variables and events $A_i = (L_i, W, \mathfrak{B}_i, E, T_i, Inv_i)$, $i = 1, 2$. We synchronize these automata to form $A = A_1 \parallel A_2$. The behavior of A is given by the following relation.

$$L(A) = L(A_1) \cap L(A_2). \tag{3.17}$$

Proof. ($L(A) \subset L(A_1) \cap L(A_2)$) Take any hybrid trajectory of A and denote it as ω. Corresponding to ω, there exists a hybrid execution of A. Denote this execution as ζ, and denote its hybrid time trajectory as T_ζ. By definition, we know that ζ satisfies all the four conditions (c1)-(c4) given in page 23. Therefore
(A1) The location is a piecewise constant function of time. It must constant between event times.
(A2) Denote the smallest event time in ε_ζ as τ_1. There is $(l^1, l^2) \in L = L_1 \times L_2$ and $w \in \mathfrak{B}\left(l^1, l^2\right)$ such that

$$\zeta(t, 0) = ((l^1, l^2), w(t)),\ 0 \leq t \leq \tau_1, \tag{3.18}$$
$$(w(t), t) \models Inv\left(l^1, l^2\right),\ 0 \leq t \leq \tau_1. \tag{3.19}$$

(A3) For any event time $\tau \in \varepsilon_\zeta$, denote $\zeta(\tau, n) =: ((l_n^1, l_n^2), \mathbf{a}_n, w_n)$, then for any n such that $0 < n \leq N_\zeta(\tau)$ there should exists a transition $\delta_n = ((l_{n-1}^1, l_{n-1}^2),$

$\mathbf{a}_n, (l_n^1, l_n^2), G_n, R_n) \in T$ such that

$$(w_{n-1}, \tau) \models G_n, \tag{3.20}$$

$$w_n \in R_n(w_{n-1}, \tau), \tag{3.21}$$

$$(w_n, \tau) \models Inv(l_n^1, l_n^2), \tag{3.22}$$

with the convention that (l_0^1, l_0^2) is the location immediately before the event time ε and w_0 is the trajectory in the interval before the event time ε.
(A4) After an event time ε_1 until the next event time (if exists) ε_2, denote the hybrid trajectory at the last event time $\zeta(\varepsilon_1, N_\zeta(\varepsilon_1)) := (\left(l^1, l^2\right), \mathbf{a}, w)$. Then for any t such that $\varepsilon_1 < t \leq \varepsilon_2$,

$$\zeta(t, 0) = (\left(l^1, l^2\right), w(t)), \tag{3.23}$$

$$(w(t), t) \models Inv\left(l^1, l^2\right). \tag{3.24}$$

Now we have to show the existence of hybrid executions of ζ_1 and ζ_2 of A_1 and A_2 that have the same hybrid trajectory as ζ. Form ζ_1 by removing the information about the location of A_2 from ζ. Similarly, we can form ζ_2 by removing the information about the location of A_1 from ζ. It can be verified that ζ_1 and ζ_2 are indeed executions of A_1 and A_2. Notice that when we create ζ_1 and ζ_2, we only change the information about the location in ζ. Consequently, ζ_1 and ζ_2 share the same hybrid trajectory as ζ.

$(L(A) \supset L(A_1) \cap L(A_2))$ Take any hybrid trajectory in $L(A_1) \cap L(A_2)$ and denote it as ω. Corresponding to ω, there are hybrid executions ζ_1 and ζ_2 of A_1 and A_2. These hybrid executions has the same hybrid time axis as that of ω. Denote this hybrid time axis as T_ζ. We shall form a hybrid execution of A, whose hybrid trajectory is ω according to the following table.

Time instant	Value of ζ_1	Value of ζ_2	Value of ζ
$(t, 0) \in T_\zeta$	$(l_1, w(t))$	$(l_2, w(t))$	$((l_1, l_2), w(t))$
$(t, n) \in T_\zeta, n > 0$	$(l_1, \mathbf{a}, w)$	$(l_2, \mathbf{a}, w)$	$((l_1, l_2), \mathbf{a}, w)$

It can be verified that ζ is indeed an execution of A. Also notice that ζ has the same hybrid trajectory as ζ_1 and ζ_2, namely ω. □

Example 3.9. Consider again the model of a room with air conditioner in Example 2.33. We are going to model a temperature control system, by using a thermostat system. The thermostat is also modeled as an HBA, $A_2 = (L_2, W_2, \mathfrak{B}_2, E_2, T_2, Inv_2)$. The set of locations $L_2 = \{\texttt{th\_on}, \texttt{th\_off}\}$. The set of continuous variable(s) $W_2 = \{\mathtt{x}\}$ taking value in $\mathbb{R}$. As in A_1, the variable $\mathtt{x}$ also represents the temperature in the room. The behaviors in each location are given as

$$\mathfrak{B}(\texttt{th\_on}) = \mathfrak{B}(\texttt{th\_on}) := \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}). \tag{3.25}$$

This means the laws of the thermostat basically do not govern the evolution of the temperature in the room. The thermostat also has two labels/events, `on` and `off`,

as is the case with A_1.
Recall that the type of the behavior corresponding to a hybrid behavioral automaton is determined by the continuous variables and the set of labels/events. Now that we know that the behavior of A_1 and A_2 are of the same type, we can model the action of thermostat on the air conditioned room by interconnecting them. Let us first continue with specifying the automaton A_2.
The set of transitions in A_2 is given by $T_2 = \{\delta_{21}, \delta_{22}\}$ where

$$\delta_{21} := (\texttt{th\_on}, \texttt{off}, \texttt{th\_off}, G_{21}, R_{21}), \tag{3.26}$$
$$\delta_{22} := (\texttt{th\_off}, \texttt{on}, \texttt{th\_on}, G_{22}, R_{22}). \tag{3.27}$$

The guards are dynamic predicates defined as follows.

$$G_{21} := (\gamma_{21}, \Gamma_{21}) \text{ and } G_{21} := (\gamma_{22}, \Gamma_{22}), \tag{3.28}$$
$$\gamma_{21}(x,t) = \gamma_{22}(x,t) := x(t), \tag{3.29}$$
$$\Gamma_{21} := \{y \in \mathbb{R} \mid y \leq 19\}, \tag{3.30}$$
$$\Gamma_{22} := \{y \in \mathbb{R} \mid y \geq 21\}. \tag{3.31}$$

Thus, δ_{21} is enabled if the temperature is less than of equal to 19 degrees, while δ_{22} is enabled if the temperature is higher than of equal to 21 degrees. The reset maps R_{21} and R_{22} are defined as

$$R_{21}(x,t) := \mathfrak{B}(\texttt{th\_off}),\ \forall x \in \mathfrak{B}(\texttt{th\_on}), t \in \mathbb{R}_+, \tag{3.32}$$
$$R_{22}(x,t) := \mathfrak{B}(\texttt{th\_on}),\ \forall x \in \mathfrak{B}(\texttt{th\_off}), t \in \mathbb{R}_+. \tag{3.33}$$

Hence any temperature trajectory can be reset to any other trajectory in the behavior of the target location. This means the thermostat basically does not specify anything about the reset.
The invariant condition Inv_2 is defined as the following dynamic predicate.

$$Inv_2(\texttt{th\_on}) := (\iota_1, I_1) \text{ and } Inv_2(\texttt{th\_off}) := (\iota_2, I_2), \tag{3.34}$$
$$\iota_1(x,t) = \iota_2(x,t) := x(t), \tag{3.35}$$
$$I_1 := \{y \in \mathbb{R} \mid y \geq 19\}, \tag{3.36}$$
$$I_2 := \{y \in \mathbb{R} \mid y \leq 21\}. \tag{3.37}$$

Hence an execution is allowed to stay in th_on only if the temperature is higher than or equal to 19 degrees. It is allowed to stay in th_off only if the temperature is lower than or equal to 21 degrees.
The executions of the thermostat automaton are those corresponding smooth trajectories. Every time the invariant condition is violated, a transition occurs and the location changes. Notice that the guards are defined such that whenever the invariant of a location is violated, a transition is enabled.
The dynamics of the room temperature when the thermostat is installed is modeled with another hybrid behavioral automaton A which is a synchronization of

A_1 and A_2. Denote $A = (L, W, \mathfrak{B}, E, T, Inv)$. By definition we have

$$L = L_1 \times L_2, \tag{3.38}$$
$$W = W_1 = W_2, \tag{3.39}$$
$$E = E_1 = E_2. \tag{3.40}$$

For brevity, we shall use the following notation for locations is L

$$\begin{aligned} l_{00} &:= (\texttt{without}, \texttt{th\_off}),\\ l_{01} &:= (\texttt{without}, \texttt{th\_on}),\\ l_{10} &:= (\texttt{with}, \texttt{th\_off}),\\ l_{11} &:= (\texttt{with}, \texttt{th\_on}). \end{aligned}$$

According to the definition of synchronization, the behaviors of each location are given by

$$\mathfrak{B}(l_{00}) = \mathfrak{B}(l_{01}) = \left\{ x \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}) \mid \frac{dx}{dt} = (30 - x) \right\}, \tag{3.41}$$
$$\mathfrak{B}(l_{10}) = \mathfrak{B}(l_{11}) = \left\{ x \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}) \mid \frac{dx}{dt} = (10 - x) \right\}. \tag{3.42}$$

Furthermore, we have that the set of transitions $T = \{\delta_1, \delta_2)$, where

$$\delta_1 = (l_{00}, \texttt{on}, l_{11}, G_1, R_1) \text{ and } \delta_2 = (l_{11}, \texttt{off}, l_{00}, G_2, R_2). \tag{3.43}$$

The guards are given by

$$G_1 := (\gamma_1, \Gamma_1) \text{ and } G_2 := (\gamma_2, \Gamma_2), \tag{3.44}$$
$$\gamma_1(x, t) = \gamma_2(x, t) := x(t), \tag{3.45}$$
$$\Gamma_1 := \{y \in \mathbb{R} \mid y \leq 19\}, \tag{3.46}$$
$$\Gamma_2 := \{y \in \mathbb{R} \mid y \geq 21\}. \tag{3.47}$$

The reset maps are given by

$$R_1(x, t) = \{x' \in \mathfrak{B}_1(l_{11}) \mid x'(t) = x(t)\},\ \forall x \in \mathfrak{B}_1(l_{00}), t \in \mathbb{R}_+, \tag{3.48}$$
$$R_2(x, t) = \{x' \in \mathfrak{B}_1(l_{00}) \mid x'(t) = x(t)\},\ \forall x \in \mathfrak{B}_1(l_{11}), t \in \mathbb{R}_+. \tag{3.49}$$

Finally, the invariant conditions for each location are given by

$$Inv(l_{00}) = Inv(l_{10}) := (\iota_{00}, I_{00}) \text{ and } Inv(l_{01}) = Inv(l_{11}) := (\iota_{11}, I_{11}), \tag{3.50}$$
$$\iota_{00}(x, t) = \iota_{11}(x, t) := x(t), \tag{3.51}$$
$$I_{00} := \{y \in \mathbb{R} \mid 21 \geq y \geq 10\}, \tag{3.52}$$
$$I_{11} := \{y \in \mathbb{R} \mid 30 \geq y \geq 19\}. \tag{3.53}$$

The executions of the hybrid automaton A can be described as follows. If the execution starts from location l_{00} or l_{11} the temperature will be well regulated, in the sense that starting from any initial temperature between 10 and 30 degrees, the temperature will evolve towards the interval $[19, 21]$ and remain there. Whenever the temperature reaches 21 degrees, the air conditioner will be switched on by the thermostat and the temperature will start dropping. Similarly, whenever the temperature reaches 19 degrees, the air conditioner will be switched off by the thermostat and the temperature will start rising. Notice that the execution can only reach locations l_{00} and l_{11}.
The fact that the execution starts from location l_{00} and l_{11} means that the state of thermostat matches the actual state of the air conditioning system. When there is a mismatch, that is executions starting from l_{01} of l_{10}, the temperature will converge exponentially to 10 degrees (for l_{10}) or to 30 degrees (for l_{01}). This is because the thermostat will not execute the necessary transition to turn the air conditioner on or off, as its state does not match the actual state of the air conditioner.
This undesired phenomenon in the modeling can be averted by, for example, introducing a notion of initial location to the automata [JSS03].

3.2 Projection and partial interconnection

So far we have been discussing interconnection of behaviors with the same type. In this section, we shall discuss the type of interconnection, where the types of the behaviors are not equal. First, we shall introduce an operation that can change the type of a behavior.

3.2.1 General behaviors

Behavior projections are mappings from a behavior to another. Formally, they are defined as follows.

Definition 3.10. [JS04a] A **projection** $\pi : \mathfrak{B}_1 \to \mathfrak{B}_2$ is a total function mapping a behavior $\mathfrak{B}_1$ of type $(\mathbb{T}_1, \mathbb{W}_1)$ to another behavior $\mathfrak{B}_2$ of type $(\mathbb{T}_2, \mathbb{W}_2)$.

Although projections are defined to be mappings between behaviors, we also naturally extend the use of projections to denote set-valued maps between behaviors. That is, for any $X \subset \mathfrak{B}_1$,

$$\pi(X) := \{w \in \mathfrak{B}_2 \mid \exists x \in X, \text{ such that } \pi(x) = w\}. \tag{3.54}$$

We also define the set-theoretic inverse of a projection as follows.

Definition 3.11. Given a projection $\pi : \mathfrak{B}_1 \to \mathfrak{B}_2$, the **set-theoretic inverse** of π, denoted as π^{-1} is defined for any $X \subset \mathfrak{B}_2$ as

$$\pi^{-1}(X) := \{w \in \mathfrak{B}_1 \mid \pi(w) \in X\}. \tag{3.55}$$

Given a behavior $\mathfrak{B}$ and a projection π acting on it. The projection π induces an equivalence relation in $\mathfrak{B}$. We define I_π, the equivalence relation generated by π by

$$(w_1, w_2) \in I_\pi :\Leftrightarrow \pi(w_1) = \pi(w_2), \tag{3.56}$$

for all $w_1, w_2 \in \mathfrak{B}$. With this definition, we can then define a partial ordering for all projections define on a given behavior.

Definition 3.12. Given a behavior $\mathfrak{B}$ and two projections π and γ acting on it. We define the **partial ordering** $\preceq$ as

$$\pi \preceq \gamma :\Leftrightarrow I_\pi \supseteq I_\gamma. \tag{3.57}$$

Two projections π and γ are equivalent if $\pi \preceq \gamma$ and $\gamma \succeq \pi$. We denote this fact by $\pi \doteq \gamma$.

The equivalence relation $\doteq$ induces a partitioning on the set of projections defined on a behavior. The partial ordering $\preceq$ can be thought of as a 'measure' of the information retained by the projection. Of course, the word 'measure' is not really appropriate, as we cannot compare every two projections. We can, however, identify the maximal and minimal class of projections with respect to $\preceq$. The maximal class consists of projections that are isomorphisms. The minimal class consists of projections that map $\mathfrak{B}$ to a singleton.

For any $\pi : \mathfrak{B}_1 \to \mathfrak{B}_2$ and $\gamma : \mathfrak{B}_2 \to \mathfrak{B}_3$, we can define a composite projection $\gamma \circ \pi$ as

$$(\gamma \circ \pi)(w) = \gamma(\pi(w)), \ \forall w \in \mathfrak{B}_1. \tag{3.58}$$

It can be proven that

$$(\gamma \circ \pi)^{-1}(X) = \pi^{-1}(\gamma^{-1}(X)), \ \forall X \subset \mathfrak{B}_3, \tag{3.59}$$

$$\pi \succeq (\gamma \circ \pi). \tag{3.60}$$

In fact, we also have the following relation.

Lemma 3.13. Given a behavior $\mathfrak{B}_1$ and two projections ϕ and γ acting on it. Suppose $\phi : \mathfrak{B}_1 \to \mathfrak{B}_2$ and $\gamma : \mathfrak{B}_1 \to \mathfrak{B}_3$. There exists a projection $\theta : \mathfrak{B}_2 \to \mathfrak{B}_3$ such that the following diagram commutes if and only if $\phi \succeq \gamma$.

$$\begin{tikzcd}
\mathfrak{B}_1 \arrow[r, "\phi"] \arrow[dr, "\gamma"'] & \mathfrak{B}_2 \arrow[d, "\theta"] \\
 & \mathfrak{B}_3
\end{tikzcd}$$

Another way to look at the partial ordering $\preceq$ is to relate it with observability. Given a behavior $\mathfrak{B}$ and a projection π acting on it, we say that the behavior $\mathfrak{B}$ is *observable* from the projection π if the equivalence relation I_π generated by π is the identity relation on $\mathfrak{B}$. That is, for any $w_1, w_2 \in \mathfrak{B}$, the following statement holds.

$$[\pi(w_1) = \pi(w_2)] \Rightarrow (w_1 = w_2). \tag{3.61}$$

When the equivalence relation I_π strictly contains the identity relation, we say that $\mathfrak{B}$ is *not observable* from the projection π. Otherwise stated, the behavior $\mathfrak{B}$ is observable from the projection π if and only if $\pi^{-1} \circ \pi$ is the identity map of $\mathfrak{B}$. This statement is not precise, since the codomain of π^{-1} is the power set of $\mathfrak{B}$ instead of $\mathfrak{B}$ itself. What is meant in fact, is that for any $w \in \mathfrak{B}$, $\left(\pi^{-1} \circ \pi\right)(w)$ returns the singleton $\{w\}$. If $\mathfrak{B}$ is not observable from the projection π, the range of $\pi^{-1} \circ \pi$ is a family of subsets of $\mathfrak{B}_1$, where each subset contains trajectories indistinguishable by the projection π.

Similarly, we extend the definition to observability of a projection γ from another projection π.

Definition 3.14. For any projections γ and π acting on the same behavior $\mathfrak{B}$, let I_π be the equivalence relation induced by π, and I_γ be that of γ. We say that γ is **observable** from π if $I_\pi \subseteq I_\gamma$. Equivalently, for any $w_1, w_2 \in \mathfrak{B}$, the following statement holds.

$$[\pi(w_1) = \pi(w_2)] \Rightarrow [\gamma(w_1) = \gamma(w_2)]\,. \tag{3.62}$$

The definition for observability above can be equivalently expressed as follows.

Lemma 3.15. For any two projections γ and π acting on the same behavior $\mathfrak{B}$, the following statements are equivalent
(i) γ is observable from π,
(ii) $\gamma \circ \pi^{-1} \circ \pi \circ \gamma^{-1}$ is the identity map on the range of γ,
(iii) $\gamma \preceq \pi$.

The whole discussion on observability so far is captured in Figure 3.2. In the illustration, we see that $\mathfrak{B}_1$ and γ are observable from π, but π is not observable from γ.

Take two behaviors $\mathfrak{B}_1$ and $\mathfrak{B}_2$ of type $(\mathbb{T}_1, \mathbb{W}_1)$ and $(\mathbb{T}_2, \mathbb{W}_2)$ respectively. Let the projections π_1 and π_2 map $\mathfrak{B}_1$ and $\mathfrak{B}_2$ to $\mathfrak{B}_1'$ and $\mathfrak{B}_2'$, which are behaviors of the same type, say $(\mathbb{T}, \mathbb{W})$. The interconnection between $\mathfrak{B}_1'$ and $\mathfrak{B}_2'$ can be defined as a total interconnection. Denote $\mathfrak{B} := \mathfrak{B}_1' \parallel \mathfrak{B}_2'$. The collection of trajectories of $\mathfrak{B}_1$ that are still allowed after the interconnection is $\pi_1^{-1}\mathfrak{B}$. Similarly, the collection of trajectories of $\mathfrak{B}_1$ that are still allowed after the interconnection is $\pi_2^{-1}\mathfrak{B}$. This is what we call *partial interconnection*. See Figure 3.3 for an illustration of this exposition.

In the previous section it was stated that interconnection of two behaviors can be thought of superposition of the laws governing each behavior. In other words, the behaviors involved restrict each other so that only trajectories compatible to both behaviors are allowed. When behaviors of different types are interconnected,

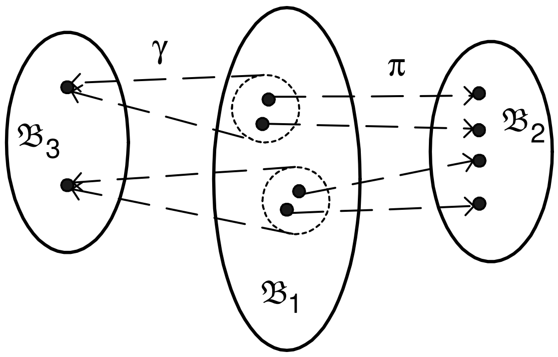


Figure 3.2: Illustration for observability.

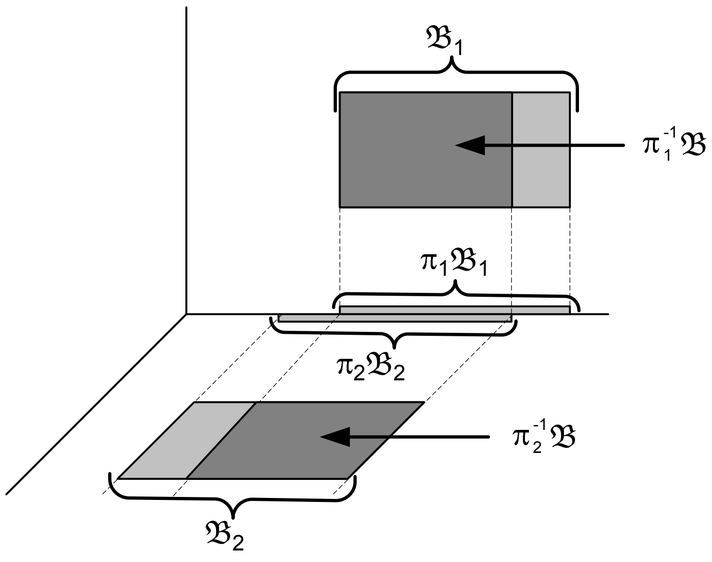


Figure 3.3: An illustration for partial interconnection. Here the behavior $\mathfrak{B} := (\pi_1\mathfrak{B}_1 \parallel \pi_2\mathfrak{B}_2)$.

the laws are expressed on different domains. Projection plays the role of determining how the information corresponding to the laws is to be translated to different domain.

The relation between projection and partial interconnection can also be summarized as follows. We use behaviors to represent system entities. These system entities can interact with its environment, or to be precise other system entities in the environment. Projection plays the role of defining how these entities interact, since one system can interact with different other entities in many different ways.

In the following subsections we shall see the general concept of projection applied to behaviors corresponding to linear time invariant systems, discrete event systems, and hybrid systems.

3.2.2 Linear time invariant systems

Projections in linear time invariant systems typically take the form of variable elimination. In modeling physical systems from first principles, one often needs to incorporate some auxiliary variables in the model to make the modeling easier. Think of, for example, an electric circuit with many components. One may be interested in modeling the relationship between voltages across certain nodes on the circuit. However, before obtaining the desired model, one might also need to derive a larger model that includes voltages across other nodes and electric currents in the circuit. These auxiliary variables are also typically called latent variables in the literature [Wil97, PW98]. Once the full model that includes the latent variables is obtained, one may want to eliminate the latent variables such that the model is only expressed in terms of the variables one is interested in. These variables are called manifest variables [Wil97, PW98].

The textbook [PW98] describes a procedure of eliminating latent variables for continuous time systems described in kernel representation. We shall include this result in this book, but we begin with the discrete time version.

Let $\mathfrak{B} \in \mathfrak{L}_{\mathrm{d}}^{\mathtt{w}+\mathtt{m}}$ be described by the following kernel representation.

$$\mathfrak{B} = \left\{ (w,m) \mid \begin{bmatrix} R(\sigma) & M(\sigma) \end{bmatrix} \begin{bmatrix} w \\ l \end{bmatrix} = 0 \right\}, \tag{3.63}$$

where $R(\xi) \in \mathbb{R}^{\mathtt{g}\times\mathtt{w}}[\xi]$ and $M(\xi) \in \mathbb{R}^{\mathtt{g}\times\ell}[\xi]$. Here $\mathbf{w}$ represents the manifest variables, and $\mathbf{l}$ the latent variables. Thus, the type of $\mathfrak{B}$ is $(\mathbb{Z}, \mathbb{R}^{\mathtt{w}+\ell})$. Define the projection $\pi_w : \mathfrak{B} \to \mathfrak{B}'$, where the type of $\mathfrak{B}'$ is $(\mathbb{Z}, \mathbb{R}^{\mathtt{w}})$, as

$$\pi_w((w,l)) = w, \forall (w,l) \in \mathfrak{B}. \tag{3.64}$$

This projection takes a trajectory in $\mathfrak{B}$ and removes the component corresponding to the latent variables. The projected behavior $\pi_w\mathfrak{B}$ can be represented by another kernel representation according to the following procedure.

1. Compute a unimodular matrix $U(\xi) \in \mathbb{R}^{\mathtt{g}\times\mathtt{g}}[\xi]$, such that

$$U(\xi)M(\xi) = \begin{bmatrix} M'(\xi) \\ 0 \end{bmatrix}, \tag{3.65}$$

with $M'(\xi) \in \mathbb{R}^{\mathtt{g}' \times \ell}[\xi]$ a full row rank matrix. The matrix M' has $\mathtt{g}'$ rows, where $\mathtt{g}' \leq \mathtt{g}$.

2. Premultiply $R(\xi)$ with $U(\xi)$ to yield

$$U(\xi)R(\xi) = \begin{bmatrix} R'(\xi) \\ R''(\xi) \end{bmatrix}, \tag{3.66}$$

with $R'(\xi) \in \mathbb{R}^{\mathtt{g}' \times \mathtt{w}}[\xi]$ and $R''(\xi) \in \mathbb{R}^{(\mathtt{g}-\mathtt{g}') \times \mathtt{w}}[\xi]$. If $\mathtt{g}' = \mathtt{g}$, then $R''(\xi) = 0$.

3. A kernel representation of $\pi_w \mathfrak{B}$ is given by

$$\pi_w \mathfrak{B} = \{ w \mid R''(\sigma) w = 0 \}. \tag{3.67}$$

To prove that this procedure really results in a kernel representation of $\pi_w \mathfrak{B}$, notice that the left unimodular transformation corresponding to $U(\xi)$ gives us another kernel representation of $\mathfrak{B}$ (see Theorem 2.20), namely

$$\mathfrak{B} = \left\{ (w, l) \mid \begin{bmatrix} R'(\sigma) & M'(\sigma) \\ R''(\sigma) & 0 \end{bmatrix} \begin{bmatrix} w \\ l \end{bmatrix} = 0 \right\}. \tag{3.68}$$

Define a behavior

$$\mathfrak{B}' := \{ w \mid R''(\sigma) w = 0 \}. \tag{3.69}$$

We shall show that $\pi_w \mathfrak{B} = \mathfrak{B}'$. First, notice that (3.68) already implies that $\pi_w \mathfrak{B} \subseteq \mathfrak{B}'$. To show that $\pi_w \mathfrak{B} \supseteq \mathfrak{B}'$ and hence prove that the two behaviors are equal, we need to prove that for any trajectory w satisfying the difference equation $R''(\sigma)w = 0$, we should always be able to find a trajectory l that solves the following difference equation.

$$M'(\sigma) l = R'(\sigma) w. \tag{3.70}$$

Lemma 3.16. Let $M(\xi) \in \mathbb{R}^{\mathtt{g} \times \ell}[\xi]$ be a full row rank matrix. Given any $R(\xi) \in \mathbb{R}^{\mathtt{g} \times \mathtt{w}}[\xi]$ and $w : \mathbb{Z} \to \mathbb{R}^{\mathtt{w}}$, it is always possible to find a trajectory $l : \mathbb{Z} \to \mathbb{R}^{\ell}$ such that the difference equation

$$M(\sigma) l = R(\sigma) w \tag{3.71}$$

is satisfied.

Proof. Define $w' := R(\sigma)w$. First, notice that if $M(\sigma)$ is just a polynomial (that is, 1-by-1 matrix), we can always find an $l : \mathbb{Z} \to \mathbb{R}^{\ell}$ such that

$$M(\sigma) l = w' \tag{3.72}$$

is satisfied. Now, we consider the multivariable case. Let $U(\xi) \in \mathbb{R}^{\mathtt{g} \times \mathtt{g}}[\xi]$ be such a unimodular matrix that $U(\xi)M(\xi)$ is upper triangular. That is, if we denote $U(\xi)M(\xi)$ as $M'(\xi)$ then

$$M'_{ij}(\xi) = 0, \text{ if } i > j. \tag{3.73}$$

Further, if we rearrange the variables in $\mathbf{l}$ properly, it is also possible to find a unimodular $U(\xi)$ such that not only (3.73) is satisfied but also

$$M'_{ii}(\xi) \neq 0,\ 1 \leq i \leq \mathrm{g}. \tag{3.74}$$

Now, notice that l satisfies (3.71) if and only if it satisfies

$$M'(\sigma)l = U(\sigma)w'. \tag{3.75}$$

Use the following notation,

$$l := \begin{bmatrix} l_1 & l_2 & \cdots & l_\ell \end{bmatrix}^{\mathrm{T}}, \tag{3.76}$$

$$w'' := U(\sigma)w' := \begin{bmatrix} w''_1 & w''_2 & \cdots & w''_{\mathrm{g}} \end{bmatrix}^{\mathrm{T}}. \tag{3.77}$$

We can find an l that solves (3.71) by setting $l_i = 0$, for all $i > \mathrm{g}$ and solving the following equations.

$$\left.\begin{aligned} M'_{\mathrm{gg}}(\sigma)l_{\mathrm{g}} &= w''_{\mathrm{g}}, \\ M'_{(\mathrm{g}-1)(\mathrm{g}-1)}(\sigma)l_{\mathrm{g}-1} &= w''_{\mathrm{g}-1} - M'_{(\mathrm{g}-1)\mathrm{g}}(\sigma)l_{\mathrm{g}}, \\ &\vdots \\ M'_{ii}(\sigma)l_i &= w''_i - \sum_{j=i+1}^{\mathrm{g}} M'_{ij}(\sigma)l_j, \\ &\vdots \\ M'_{11}(\sigma)l_1 &= w''_1 - \sum_{j=2}^{\mathrm{g}} M'_{ij}(\sigma)l_j. \end{aligned}\right\} \tag{3.78}$$

Notice that (3.78) is a set of coupled scalar difference equations, which we know to be solvable. □

With this lemma, we establish that (3.67) does indeed give a kernel representation of the projected behavior $\pi_w\mathfrak{B}$.

For continuous time behaviors corresponding to the strong solutions of a kernel representation, the same procedure applies and the proof follows by replacing the σ operator with the differential operator $\frac{d}{dt}$. For continuous time behaviors corresponding to the weak solutions of a kernel representation, the procedure generally does not work [Pol97]. Look at equation (3.70), with σ replaced by $\frac{d}{dt}$. The differential equation

$$M'\left(\frac{d}{dt}\right)l = R'\left(\frac{d}{dt}\right)w \tag{3.79}$$

can pose a smoothness condition on w. Thus, not all w such that $R''(\frac{d}{dt})w = 0$ can be matched with a latent trajectory l. Consider, for example, the simple case where $R'' = 0$.

When the behavior $\mathfrak{B} \in \bar{\mathfrak{L}}_c^{\mathtt{w}+\ell}$ given by the kernel representation

$$\mathfrak{B} = \left\{ (w,l) \mid \begin{bmatrix} R\left(\frac{d}{dt}\right) & M\left(\frac{d}{dt}\right) \end{bmatrix} \begin{bmatrix} w \\ l \end{bmatrix} = 0 \right\} \tag{3.80}$$

is such that the procedure above does actually compute a kernel representation of $\pi_w \mathfrak{B}$, we say the l is properly eliminable from $\mathfrak{B}$ [Pol97]. If l is not properly eliminable from $\mathfrak{B}$, then there does not exist any kernel representation of $\pi_w \mathfrak{B}$ and $\pi_w \mathfrak{B}$ is not closed [PW98]. Hence, in this case $\pi_w \mathfrak{B} \notin \bar{\mathfrak{L}}_c^{\mathtt{w}}$. However, if we still proceed according to the elimination procedure above, we still obtain a behavior $\mathfrak{B}'$, whose kernel representation is $R''\left(\frac{d}{dt}\right) w = 0$. In [PW98] it is proven that $\mathfrak{B}'$ is the closure of $\pi_w \mathfrak{B}$. Therefore, if l is properly eliminable from $\mathfrak{B}$, then $\pi_w \mathfrak{B}$ is closed and coincides with $\mathfrak{B}'$. Because of this, the condition where l is properly eliminable from $\mathfrak{B}$ is also called *exact elimination*.

Example 3.17. As an example, we consider the state elimination from an input-state-output representation of a linear time invariant system. Consider the continuous time system given by the following representation

$$\frac{d}{dt}x(t) = ax(t) + bu(t), \tag{3.81a}$$

$$y(t) = cx(t). \tag{3.81b}$$

We assume, for simplicity, that the state, input and output are all one dimensional. We also assume that $c \neq 0$. The behavior of this system can be represented by the following kernel representation.

$$\begin{bmatrix} \frac{d}{dt} - a & -b & 0 \\ c & 0 & -1 \end{bmatrix} \begin{bmatrix} x \\ u \\ y \end{bmatrix} = 0. \tag{3.82}$$

Suppose that we are interested in the weak solutions of this equation. Notice that by premultiplying this kernel representation by the unimodular matrix

$$U\left(\frac{d}{dt}\right) = \begin{bmatrix} 0 & 1 \\ c & -\frac{d}{dt} + a \end{bmatrix}, \tag{3.83}$$

we obtain another kernel representation

$$\begin{bmatrix} c & 0 & -1 \\ 0 & -bc & \frac{d}{dt} - a \end{bmatrix} \begin{bmatrix} x \\ u \\ y \end{bmatrix} = 0. \tag{3.84}$$

If we follow the analog of the proof of Lemma 3.16 for continuous time systems, it can be shown that the state is actually properly eliminable from this behavior and that the kernel representation of the behavior after the state is eliminated is given by the second row of (3.84),

$$\begin{bmatrix} -bc & \frac{d}{dt} - a \end{bmatrix} \begin{bmatrix} u \\ y \end{bmatrix} = 0. \tag{3.85}$$

We have shown that for first order systems, the state is properly eliminable. In fact, a more general result also holds. Namely, that the states are always properly eliminable for higher order systems [Pol97].

The concept of observability for linear time invariant systems follows from the general discussion in the previous subsection. Consider the following theorem.

Theorem 3.18. Let $\mathfrak{B}$ be a linear time invariant behavior represented by the kernel representation

$$\mathfrak{B} = \left\{ w \mid R\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{3.86}$$

Let π_1 and π_2 denote the projection of $\mathfrak{B}$ to $\mathfrak{B}_1$ and $\mathfrak{B}_2$ where the trajectories are defined as

$$w_1 = T_1\left(\frac{d}{dt}\right) w \text{ and } w_2 = T_2\left(\frac{d}{dt}\right) w. \tag{3.87}$$

The projection π_1 is observable from π_2 if and only if there exist F and G such that

$$T_1 = FR + GT_2. \tag{3.88}$$

Proof. By Definition 3.14, π_1 is observable from π_2 if and only if the equivalence class generated by π_1 is larger than that by π_2. This is equivalent to

$$\ker \begin{bmatrix} R\left(\frac{d}{dt}\right) \\ T_1\left(\frac{d}{dt}\right) \end{bmatrix} \supseteq \ker \begin{bmatrix} R\left(\frac{d}{dt}\right) \\ T_2\left(\frac{d}{dt}\right) \end{bmatrix}. \tag{3.89}$$

Equation (3.89) implies that

$$\ker T_1\left(\frac{d}{dt}\right) \supseteq \ker \begin{bmatrix} R\left(\frac{d}{dt}\right) \\ T_2\left(\frac{d}{dt}\right) \end{bmatrix}. \tag{3.90}$$

This is the case if and only if there exists a polynomial matrix M of appropriate size, such that

$$T_1(\xi) = M(\xi) \begin{bmatrix} R(\xi) \\ T_2(\xi) \end{bmatrix}. \tag{3.91}$$

We partition M into

$$M := \begin{bmatrix} F & G \end{bmatrix}, \tag{3.92}$$

and complete the proof. $\square$

This result also holds for discrete time linear systems, if we replace the operator $\frac{d}{dt}$ with σ.

As a special case of this result, we consider the situation where the projections π_1 and π_2 simply partition the variables into two groups. That is, $\mathrm{col}(T_1, T_2)$ is the identity matrix.

Theorem 3.19. (cf. Theorem 5.3.3 in [PW98]) Let $\mathfrak{B}$ be a linear time invariant behavior represented by the kernel representation

$$\mathfrak{B} = \left\{ (w_1, w_2) \mid \begin{bmatrix} R_1\left(\frac{d}{dt}\right) & R_2\left(\frac{d}{dt}\right) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = 0 \right\}, \tag{3.93}$$

or

$$\mathfrak{B} = \left\{ (w_1, w_2) \mid \begin{bmatrix} R_1(\sigma) & R_2(\sigma) \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \end{bmatrix} = 0 \right\}. \tag{3.94}$$

Let π_1 and π_2 denote the projection of $\mathfrak{B}$ corresponding to the elimination of w_2 and w_1 respectively. The projection π_1 is observable from π_2 if and only if the the (real) matrix $R_1(\xi)$ has full column rank for all complex numbers $\xi \in \mathbb{C}$.

As an application of this result, consider the linear system given in state-space representation

$$\frac{dx}{dt} = Ax + Bu, \tag{3.95a}$$

$$y = Cx + Du. \tag{3.95b}$$

The behavior of this system can be represented by the following kernel representation.

$$\begin{bmatrix} \frac{d}{dt}I - A & B & 0 \\ C & D & I \end{bmatrix} \begin{bmatrix} x \\ u \\ y \end{bmatrix} = 0. \tag{3.96}$$

Following Theorem 3.19, the necessary and sufficient condition for observability of the states from the inputs and outputs is that $\begin{bmatrix} \xi I - A \\ C \end{bmatrix}$ should have full row rank for all $\xi \in \mathbb{C}$. This condition coincides with the renowned Hautus test for observability. In fact, it is also equivalent with the famous Kalman rank condition for observability [PW98].

3.2.3 Discrete event systems

The most common projection for discrete event systems is similar to what is called the *natural projection* of a regular language [CL99]. The idea can be explained as follows. Let L be a regular language over the alphabet $\mathbf{A}$. Thus $L \subset \mathbf{A}^*$, and the type of the behavior corresponding to L is $(\mathbb{Z}_+, \mathbf{A})$. For any subset $\mathbf{A}' \subset \mathbf{A}$, the natural projection to $\mathbf{A}'$, denoted as $\pi_{\mathbf{A}'}L$, maps strings on $\mathbf{A}^*$ to strings on $(\mathbf{A}')^*$. The projection acting on L will thus produce a behavior of type $(\mathbb{Z}_+, \mathbf{A}')$.The projection is defined recursively as follows.

1. The empty string is projected to the empty string, thus $\pi_{\mathbf{A}'}\varepsilon = \varepsilon$.
2. For any string $s \in \mathbf{A}^*$ and $a \in \mathbf{A}$,

$$\pi_{\mathbf{A}'}(sa) = \begin{cases} (\pi_{A'}s)a, & \text{if } a \in \mathbf{A}', \\ \pi_{A'}s & \text{if } a \notin \mathbf{A}'. \end{cases} . \tag{3.97}$$

The projection removes all the occurrences of events that are not elements of $\mathbf{A}'$.

However, there is a difference between the natural projection and behavioral projection as it is introduced in Subsection 3.2.1. The difference lies in the notion of inverse projection. In the definition of natural projection [CL99], the inverse projection is defined as

$$\pi_{\mathbf{A}'}^{-1}(X) := \{s \in \mathbf{A}^* \mid \pi_{\mathbf{A}'}s \in X\}. \tag{3.98}$$

While according to Definition 3.11, it is defined as

$$\pi_{\mathbf{A}'}^{-1}(X) := \{s \in L \mid \pi_{\mathbf{A}'}s \in X\}. \tag{3.99}$$

Thus, a behavioral projection is defined with L as its domain while a natural projection has $\mathbf{A}^*$ as its domain.

Since we are interested only in regular languages, it is interesting to verify if projection preserves the regularity of the language. Indeed, this is the case.

Theorem 3.20. Given a regular language L over an alphabet $\mathbf{A}$. For any subset $\mathbf{A}' \subset \mathbf{A}$, the projected language $\pi_{\mathbf{A}'}L$ is also regular.

Proof. By Kleene's theorem, a language is regular if and only if it can be expressed in terms of a regular expression. This implies that L has a regular expression. We can obtain a regular expression for $\pi_{\mathbf{A}'}L$ by replacing any event that is not contained in $\mathbf{A}'$ by ε. Thus $\pi_{\mathbf{A}'}L$ is also regular. □

Observability between projections, as it is introduced in the general discussion is Subsection 3.2.1, can be interpreted in the following way. Given a regular language L over an alphabet $\mathbf{A}$. Let $\mathbf{A}_1$ and $\mathbf{A}_2$ be subsets of $\mathbf{A}$, and let π_1 and π_2 denote the projection with respect to $\mathbf{A}_1$ and $\mathbf{A}_2$ respectively. Imagine that L is the language marked (or generated) by a certain discrete event system. There are two observers, O_1 and O_2. The observer O_1 (resp. O_2) can only observe an event if it is contained in $\mathbf{A}_1$ (resp. $\mathbf{A}_2$). We say that the projection π_1 is observable from π_2 if the observer O_2 can always infer about the observation made by O_1, given the knowledge about L, $\mathbf{A}$, $\mathbf{A}_1$, $\mathbf{A}_2$, and the string s_2 that he observes. Notice that this notion of observability is different from the observability of a language from another language, which is a well-known concept in supervisory control of discrete-event systems [Ram87, RW89, CL99]. It is also not about observing the state of a representing automaton, given a string of events.

For a prefix closed language, the condition for observability between projections is given by the following simple lemma. This lemma is the analog of Theorem 3.18 for prefix closed regular languages.

Lemma 3.21. [JS04a] Given L a prefix closed language, with $\mathbf{A}$ as its alphabet. Assume that every event in $\mathbf{A}$ appears at least once[1] in L. Denote the projections with respect to the subsets of events $\mathbf{A}_1$ and $\mathbf{A}_2$ as π_1 and π_2 respectively. The projection π_1 is observable from π_2 if and only if $\mathbf{A}_1 \subseteq \mathbf{A}_2$.

Proof. (if) Define π' to be the projection with respect to the set of labels $\mathbf{A}_1$, acting on the codomain of π_2. Clearly, we have that $\pi_1 = \pi_2 \circ \pi'$. By (3.60) we can infer that $\pi_1 \preceq \pi_2$. Due to Lemma 3.15, this implies observability of π_1 from π_2. (only if) Suppose that $\mathbf{A}_1 \not\subseteq \mathbf{A}_2$. Let a be an element of $\mathbf{A}_1$, which is not in $\mathbf{A}_2$. Since L is prefix closed and a appears at least once in the language, there is a string (possibly empty) $s \in L$, such that $sa \in L$. The pair (s, sa) is in the equivalence relation induced by π_2 but not in that of π_1. Thus, $\pi_1 \not\preceq \pi_2$ and therefore π_1 is not observable from π_2. $\square$

Partial interconnection for discrete event systems typically takes the form of parallel composition [CL99].

Definition 3.22. Given two regular languages L_1 and L_2 over the alphabet $\mathbf{A}_1$ and $\mathbf{A}_2$ respectively. Assume that $\mathbf{A}_1 \cap \mathbf{A}_2 = \mathbf{A} \neq \emptyset$. Let $A_1 = (X_1, \mathbf{A}_1, T_1, X_{\mathrm{m}1}, x_{01})$ and $A_2 = (X_2, \mathbf{A}_2, T_2, X_{\mathrm{m}2}, x_{02})$ be automata that accept the language L_1 and L_2. We define the **parallel composition** of A_1 and A_2 as the automaton $A = (X, \mathbf{A}_1 \cup \mathbf{A}_1, T, X_{\mathrm{m}}, x_0)$ where

$$
\begin{aligned}
X &= X_1 \times X_2, && \text{(3.100a)}\\
T &= \{((x_1,x_2),\mathbf{a},(x_1',x_2')) \in X \times \mathbf{A} \times X \mid (x_i,\mathbf{a},x_i') \in T_i,\ i=1,2.\} \cup \\
&\quad \{((x_1,x_2),\mathbf{a},(x_1',x_2)) \in X \times (\mathbf{A}_1 - \mathbf{A}) \times X \mid (x_1,\mathbf{a},x_1') \in T_1.\} \cup \\
&\quad \{((x_1,x_2),\mathbf{a},(x_1,x_2')) \in X \times (\mathbf{A}_2 - \mathbf{A}) \times X \mid (x_2,\mathbf{a},x_2') \in T_2.\}, && \text{(3.100b)}\\
X_{\mathrm{m}} &= X_{\mathrm{m}1} \times X_{\mathrm{m}2}, && \text{(3.100c)}\\
x_0 &= (x_{01}, x_{02}). && \text{(3.100d)}
\end{aligned}
$$

This parallel composition is denoted as $A = A_1 \parallel A_2$.

The parallel composition thus forces the automata to synchronize on the events both automata have in common. The executions of the resulting automaton are related to executions of A_1 and A_2 as follows. Take any execution of A with length N, $(x_{01}, x_{02})a_1(x_{11}, x_{12})\ a_2(x_{21}, x_{22}) \cdots\ a_N(x_{N1}, x_{N2})$. From the definition of A, there exist unique I_1 and I_2, both subsets of $\{1, 2, \ldots, N\}$ such that
(i) $I_1 \cup I_2 = \{1, 2, \ldots, N\}$,
(ii) For any $1 \leq i \leq N$, $a_i \in \mathbf{A}_1 \cap \mathbf{A}_2$ if and only if $i \in I_1 \cap I_2$,
(iii) If we order (ascendingly) and denote the elements of I_1 and I_2 as $\{i_1, i_2, \ldots, i_n\}$ and $\{j_1, j_2, \ldots, j_m\}$, where n and m are the cardinalities of I_1 and I_2, then the string $x_{01}a_{i_1}x_{i_11}\ a_{i_2}x_{i_21} \cdots a_{i_n}x_{i_n1}$ is an execution of A_1 and the string $x_{02}a_{j_1}\ x_{j_12}a_{j_2}\ x_{j_22} \cdots a_{j_m}x_{j_m2}$ is an execution of A_2.

[1]This can be done without any loss of generality, since symbols that never appear can be discarded from the alphabet.

Thus, every event in an execution of A is in fact executed independently by A_1 or A_2 if that event is not in $\mathbf{A}_1 \cap \mathbf{A}_2$. If the event is in $\mathbf{A}_1 \cap \mathbf{A}_2$, then it is executed simultaneously by both A_1 and A_2. Thus, given any execution of A, we can always tell which is the part executed by A_1 and which by A_2.

Conversely, take any executions $\zeta_1 := x_{01}a_{11}x_{11}a_{21}x_{21} \cdots a_{n1}x_{n1}$ of A_1 and $\zeta_2 := x_{02}a_{12}x_{12}\ a_{22}x_{22} \cdots a_{m2}x_{m2}$ of A_2. Denote the strings of events corresponding to the executions as s_1 and s_2 respectively. Also, denote the projections that projects strings in $\mathbf{A}_1^*$ and $\mathbf{A}_2^*$ to $(\mathbf{A}_1 \cap \mathbf{A}_2)^*$ as π_1 and π_2. If $\pi_1 s_1 = \pi_2 s_2$, we can always construct an execution ζ of A that matches ζ_1 and ζ_2.

The relation between partial synchronization and partial behavior interconnection is given by the following theorem.

Theorem 3.23. Given two finite state automata $A_1 = (X_1, \mathbf{A}_1, T_1, X_{\mathrm{m}1}, x_{01})$ and $A_2 = (X_2, \mathbf{A}_2, T_2, X_{\mathrm{m}2}, x_{02})$ such that $\mathbf{A}_1 \cap \mathbf{A}_2 = \mathbf{A} \neq \emptyset$. Let $A = A_1 \parallel A_2$. Denote the projection that projects the marked language of A_1 and A to the event set $\mathbf{A}$ as π_1. Similarly, we denote the projection that projects the marked language of A_2 to the event set $\mathbf{A}$ as π_2. Further, let γ_1 and γ_2 be the projections that project the marked language of A to the event set $\mathbf{A}_1$ and $\mathbf{A}_2$ respectively. The following relations hold.

$$\gamma_1 L_{\mathrm{m}}(A) = \pi_1^{-1}(\pi_1 L_{\mathrm{m}}(A_1) \parallel \pi_2 L_{\mathrm{m}}(A_2)), \tag{3.101}$$

$$\gamma_2 L_{\mathrm{m}}(A) = \pi_2^{-1}(\pi_1 L_{\mathrm{m}}(A_1) \parallel \pi_2 L_{\mathrm{m}}(A_2)). \tag{3.102}$$

Proof. Because of symmetry, we do not have to prove both (3.101) and (3.102), but only one of them. We choose (3.101).

($\gamma_1 L_{\mathrm{m}}(A) \subseteq \pi_1^{-1}(\pi_1 L_{\mathrm{m}}(A_1) \parallel \pi_2 L_{\mathrm{m}}(A_2))$) Take any string $s_1 \in \gamma_1 L_{\mathrm{m}}(A)$. By definition, there exists an $s \in L_{\mathrm{m}}(A)$ such that $\gamma_1 s = s_1$. We need to prove that $\pi_1 s_1 \in (\pi_1 L_{\mathrm{m}}(A_1) \parallel \pi_2 L_{\mathrm{m}}(A_2))$. Corresponding to s, there should exists an execution $(x_{01}, x_{02})a_1(x_{11}, x_{12})\ a_2(x_{21}, x_{22}) \cdots\ a_N(x_{N1}, x_{N2})$ of A that terminates on a marked state. We can identify an execution from each of A_1 and A_2 that correspond to this execution. Denote these executions by ζ_1 and ζ_2 respectively. Since both ζ_1 and ζ_2 terminate on a marked state, obviously the strings of events they generate, denoted as s_1' and s_2', are in $L_{\mathrm{m}}(A_1)$ and $L_{\mathrm{m}}(A_2)$ respectively. Moreover, $\pi_1 s = \pi_1 s_1' = \pi_2 s_2'$ because any event in $\mathbf{A}$ must be executed jointly by A_1 and A_2. Therefore $\pi_1 s_1 \in (\pi_1 L_{\mathrm{m}}(A_1) \parallel \pi_2 L_{\mathrm{m}}(A_2))$.

($\gamma_1 L_{\mathrm{m}}(A) \supseteq \pi_1^{-1}(\pi_1 L_{\mathrm{m}}(A_1) \parallel \pi_2 L_{\mathrm{m}}(A_2))$) Take any string

$$s_1 \in \pi_1^{-1}(\pi_1 L_{\mathrm{m}}(A_1) \parallel \pi_2 L_{\mathrm{m}}(A_2)).$$

By definition, there exist strings $s_1' \in L_{\mathrm{m}}(A_1)$ and $s_2' \in L_{\mathrm{m}}(A_2)$ such that

$$\pi_1 s_1' = \pi_2 s_2' =: s, \tag{3.103}$$

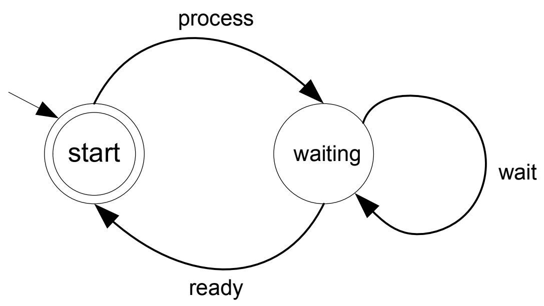
$$\pi_1 s_1 = s. \tag{3.104}$$

Now we need to prove that $s_1 \in \gamma_1 L_{\mathrm{m}}(A)$, that is, there is a string s' in $L_{\mathrm{m}}(A)$ such that $\gamma_1 s' = s_1$. First of all, there are executions ζ_1 and ζ_1' of A_1 corresponding to s_1

and s'_1. There is also an execution ζ'_2 of A_2 corresponding to s'_2. Obviously, since $\pi_1 s_1$ and $\pi_2 s'_2$ are equal, we can construct an execution ζ' of A that matches ζ_1 and ζ'_2. Denote the string of events corresponding to ζ' as s'. We obviously have that $s_1 = \gamma_1 s'$. Since $s_1 \in L_{\mathrm{m}}(A_1)$ and $s'_2 \in L_{\mathrm{m}}(A_2)$, we can infer that $s' \in L_{\mathrm{m}}(A)$. $\square$

The expressions $\gamma_1 L_{\mathrm{m}}(A)$ and $\gamma_2 L_{\mathrm{m}}(A)$ gives the strings of event terminating on a marked state of A, seen by A_1 and A_2. Although this theorem only discusses the marked languages of the automata, we can derive a similar result for the generated languages by considering a special case where all states of the automata are marked.

Example 3.24. Consider the automaton model of the production line in Example 2.28. That model gives an outsider point of view of the production line, as after a batch of raw material arrives, it is not clear whether the material is going to be processes immediately or has to wait before being processed. To give a more detailed description, we can model the machine that processes the raw material with a simple automaton $M = (X', E', T', \{\texttt{start}\}, \texttt{start})$, described by the following diagram.



There are two states of the machine. It is either in a state in which it is ready to process the raw material, or in a state where it is waiting for an event called `ready` to take place before it can process the next batch of raw material. The event `ready` can be thought of as a representation of the time the machine needs between each process. If the machine is in the waiting state and a batch of material arrives, then the material has to wait until the machine is ready.
We can form a more detailed model by taking the parallel composition of M and P. Notice that in this case, the two automata synchronizes on the event `process` and `wait`. The marked languages of the automata are

$$L_{\mathrm{m}}(P) = (\texttt{arrival}(\texttt{wait.process} + \texttt{process})\texttt{output})^*, \tag{3.105}$$
$$L_{\mathrm{m}}(M) = (\texttt{process.wait}^*\texttt{ready})^* . \tag{3.106}$$

The projected language to the set of joint events are

$$\pi_1 L_{\mathrm{m}}(P) = (\texttt{wait.process} + \texttt{process})^*,$$
$$\pi_2 L_{\mathrm{m}}(M) = (\texttt{process.wait}^*)^*. \tag{3.107}$$

Thus the intersection is

$$\pi_1 L_\mathrm{m}(P) \parallel \pi_2 L_\mathrm{m}(M) = \texttt{process}(\texttt{wait.process} + \texttt{process})^*. \tag{3.108}$$

We can also see that

$$\pi_1^{-1}(\pi_1 L_\mathrm{m}(P) \parallel \pi_2 L_\mathrm{m}(M)) = \texttt{arrival.process.output.}L_\mathrm{m}(P), \tag{3.109}$$

$$\pi_2^{-1}(\pi_1 L_\mathrm{m}(P) \parallel \pi_2 L_\mathrm{m}(M)) = (\texttt{process.}(\texttt{wait.ready} + \texttt{ready}))^* . \tag{3.110}$$

These are the marked language of the composed automaton seen by P and M respectively.

3.2.4 Hybrid systems

The notion of projection for hybrid systems can be thought of as a combination of that of continuous time dynamical systems and that of discrete event systems. By this, we mean that the projection operation on hybrid systems can eliminate some continuous variables, or erase some events, or be a combination of both.

There is, however, a difference between the way an event is erased in discrete event systems and in hybrid systems. In the previous subsection, we have seen that in discrete event systems, events are erased by just removing them from the strings. In hybrid systems, we remove events and time events because they are interrelated. So let us define precisely what is meant by hiding some events.

Given a hybrid system represented as a hybrid behavioral automaton $A = (L, W, \mathfrak{B}, E, T, Inv)$. Suppose that we want to erase the occurrences of events not contained in $E' \subset E$. Take any trajectory of A, and denote it as ω. Corresponding to ω is a hybrid time axis T_ω with event times ε_ω. We erase the occurrences of events that are not in E' by constructing another trajectory ω' such that

$$\omega'(t, 0) = \omega(t, 0), \ \forall t \in [0, T_\omega]. \tag{3.111}$$

For the event times, suppose that $t \in \varepsilon_\omega$ and the event density $N_\omega(t) = n$. We then form a subset of these n events, whose labels are in E'. Denote the number of elements in this subset as n'. Then the event density of ω' at time t is n', because we leave out events that are not in E'. If $n' = 0$ then t is not an event time for ω'. As a special case, consider the situation where we hide all the events. The resulting trajectory will be have only piecewise continuous part, with a hybrid time axis without any event times.

Example 3.25. Consider a system described by the following input-state-output representation

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & -1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 0 \\ 1 \end{bmatrix} u, \tag{3.112}$$

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}. \tag{3.113}$$

This can be thought of as a model for the dynamics of a point mass moving on a rough surface. The state x_1 denotes the position of the mass, and x_2 denotes its velocities. The input u is an external force exerted on the point mass. This is a simple linear time invariant system. Nevertheless, we can also cast this model as a hybrid system. It is a hybrid behavioral automaton with only one location, no transitions, empty set of labels, and an invariant condition that is always true. So we can write is as $A_1 = (L_1, W_1, \mathfrak{B}_1, \emptyset, \emptyset, \textbf{true})$. With L_1 the singleton $\{l\}$ and $\mathfrak{B}_1(l)$ is

$$\mathfrak{B}_1(l) := \{(x_1, x_2, u, y_1, y_2) \mid \text{(3.112)-(3.113) are weakly satisfied}\}. \tag{3.114}$$

We also assume that the trajectories are left-continuous. The reason why we model this system as a hybrid system is because we are going to control it with a hybrid system. The control interconnection will be expressed as a partial interconnection. Suppose that we want to make sure that the point mass will return to the origin quickly, and that we can attach $\mathbf{y}_1$, $\mathbf{y}_2$ and $\mathbf{u}$ to the controller. Also suppose that we have two controllers at our disposal. The first controller has a spring like behavior, as it is described by

$$u = -y_1. \tag{3.115}$$

The second controller has a spring-damper like behavior, with a very strong damping. It is described by

$$u = -y_1 - 10 \cdot y_2. \tag{3.116}$$

If we use only the first controller, the evolution of the position of the mass will be fast but oscillatory. On the other hand, using only the second controller results in a sluggish behavior without any oscillation. We then formulate a controller, which is given as a hybrid behavioral automaton $A_2 = (L_2, W_2, \mathfrak{B}_2, E_2, T_2, Inv_2)$ where

$$\begin{aligned}
L_2 &= \{l_1, l_2\},\\
W_2 &= \{\mathbf{u}, \mathbf{y}_1, \mathbf{y}_2\},\\
B_2(l_1) &= \{(u, y_1, y_2) \mid \text{(3.115) is weakly satisfied}\},\\
B_2(l_1) &= \{(u, y_1, y_2) \mid \text{(3.116) is weakly satisfied}\},\\
E_2 &= \{\mathtt{a}, \mathtt{b}\},
\end{aligned}$$

and $T = \{\delta_1, \delta_2\}$, where

$$\delta_1 = (l_1, \mathtt{a}, l_2, G_1, R_1) \text{ and } \delta_2 = (l_2, \mathtt{b}, l_1, G_2, R_2). \tag{3.117}$$

The guards are given by

$$G_1 := (\gamma_1, \Gamma_1) \text{ and } G_2 := (\gamma_2, \Gamma_2), \tag{3.118}$$

$$\gamma_1(u, y_1, y_2, t) = \gamma_2(u, y_1, y_2, t) := y_1(t), \tag{3.119}$$

$$\Gamma_1 := \{y \in \mathbb{R} \mid |y| \leq 0.1\}, \tag{3.120}$$

$$\Gamma_2 := \{y \in \mathbb{R} \mid |y| \geq 0.1\}. \tag{3.121}$$

The reset maps are given by

$$R_1(u, y_1, y_2, t) = \{(u', y_1', y_2') \in \mathfrak{B}_2(l_2) \mid y_1'(t) = y_1(t) \text{ and } y_2'(t) = y_2(t)\}, \quad (3.122)$$
$$R_2(u, y_1, y_2, t) = \{(u', y_1', y_2') \in \mathfrak{B}_2(l_1) \mid y_1'(t) = y_1(t) \text{ and } y_2'(t) = y_2(t)\}, \quad (3.123)$$

Finally, the invariant conditions for each location are given by

$$Inv(l_1) := (\iota_{00}, I_{00}) \text{ and } Inv(l_2) := (\iota_{11}, I_{11}), \quad (3.124)$$
$$\iota_{00}(u, y_1, y_2, t) = \iota_{11}(u, y_1, y_2, t) := y_1(t), \quad (3.125)$$
$$I_{00} := \{y \in \mathbb{R} \mid |y| \geq 0.1\}, \quad (3.126)$$
$$I_{11} := \{y \in \mathbb{R} \mid |y| \leq 0.1\}. \quad (3.127)$$

The controller is designed such that whenever the mass is further than 0.1 from the origin, the first controller is in effect. This is to ensure fast response so that the mass is brought to approach the origin as quickly as possible. Whenever the mass is closer than 0.1 from the origin, the second controller is in effect to ensure attenuation of the motion and to prevent oscillation.
We define π_1 as the projection acting on the behavior of A_1, $L(A_1)$, such that the state variables are eliminated, and π_2 as the projection acting on the behavior of A_2, $L(A_2)$, such that the events a and b are erased. Notice that $\pi_1 L(A_1)$ and $\pi_2 L(A_2)$ are now of the same type, as they share the same set of continuous variables, namely $\{u, y_1, y_2\}$ and the same set of events, namely the empty set. The control interconnection can then be expressed as $\pi_1 L(A_1) \parallel \pi_2 L(A_2)$. The effect of the interconnection, seen from A_1 is $\pi_1^{-1}(\pi_1 L(A_1) \parallel \pi_2 L(A_2))$. Refer to Figure 3.4 to see a comparison between the trajectories of the controlled system when the first controller, the second controller and the hybrid controller is used.

Example 3.26. We can also model a one degree-of-freedom juggling robot in this framework [ZB99]. A juggling robot can be modelled as an interconnection of two systems, namely a ball and a robot, as shown in Figure 3.5. Although the dynamics of both systems are continuous, they are modelled as hybrid systems. The reason is because of the collision event that can occur. Each system is modelled as a hybrid behavioral automaton with single location, as in the previous example. The ball is modelled as $A_1 = (L_1, W_1, \mathfrak{B}_1, E, T_1, Inv_1)$, where $L_1 = \{l_1\}$, $W_1 = \{\mathtt{x}_1, \mathtt{x}_2\}$, and the behavior $\mathfrak{B}_1(l_1)$ consists of all continuously differentiable trajectories that satisfy the following equation weakly.

$$\frac{d^2}{dt^2} x_1 + g = 0. \quad (3.128)$$

Thus the variable $\mathtt{x}_2$ is free, in the sense that its trajectory can be any continuously differentiable function. The event set $E = \{\mathtt{a}\}$, $T_1 = \{\delta_1\}$, where

$$\delta_1 = (l_1, \mathtt{a}, l_1, G_1, R_1). \quad (3.129)$$

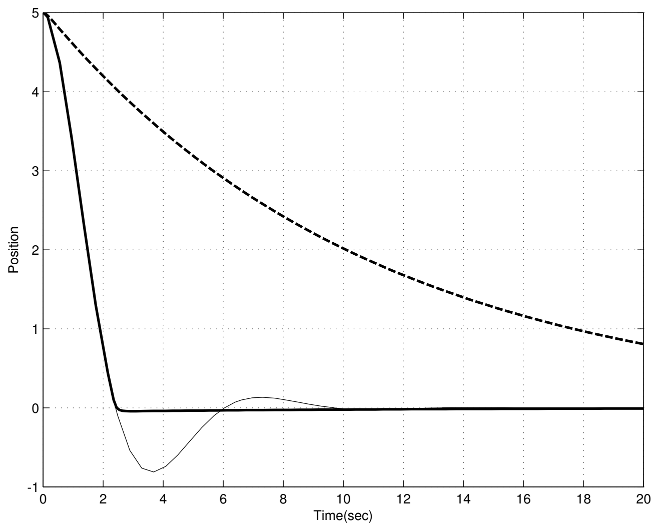


Figure 3.4: A comparison between trajectories of the position of the point mass in Example 3.25.Thin line: when the first controller is used. Dashed line: when the second controller is used. Thick line: when the hybrid controller is used.

The guard of δ_1 is given by

$$G_1 := (\gamma_1, \Gamma_1) \tag{3.130}$$

$$\gamma_1(x_1, x_2, t) := x_1(t) - x_2(t), \tag{3.131}$$

$$\Gamma_1 := \{y \in \mathbb{R} \mid y \leq 0\}. \tag{3.132}$$

The reset map is given by

$$R_1(x_1, x_2, t) = \left\{ (x_1', x_2') \in \mathfrak{B}_1(l_1) \mid \frac{dx_1'}{dt}(t) = \frac{(m-M)\frac{dx_1}{dt}(t) + 2M\frac{dx_2}{dt}(t)}{m+M} \right\}. \tag{3.133}$$

Here, m and M denote the mass of the ball and the robot respectively. The reset maps indicates that the collision is elastic.
The robot is modelled as $A_2 = (L_2, W_2, \mathfrak{B}_2, E, T_2, Inv_2)$, where $L_2 = \{l_2\}$, $W_2 = \{\mathbf{x}_1, \mathbf{x}_2, \mathbf{u}\}$, and the behavior $\mathfrak{B}_2(l_2)$ consists of all continuously differentiable trajectories that satisfy the following equation weakly.

$$M\frac{d^2}{dt^2}x_2 - u = 0. \tag{3.134}$$

Thus the variable $\mathbf{x}_1$ is free. The dynamics of the robot does not explicitly show the effect of gravity, because we can consider it to be absorbed into the input force

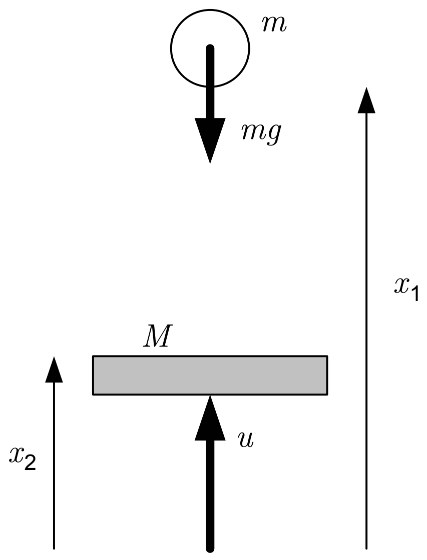


Figure 3.5: One degree-of-freedom juggling robot [ZB99].

u. The event set $E = \{\mathsf{a}\}, T_2 = \{\delta_2\}$, where

$$\delta_1 = (l_2, \mathsf{a}, l_2, G_2, R_2). \tag{3.135}$$

The guard of δ_2 is given by

$$G_2 := (\gamma_2, \Gamma_2) \tag{3.136}$$
$$\gamma_2(x_1, x_2, u, t) := x_1(t) - x_2(t), \tag{3.137}$$
$$\Gamma_2 := \{y \in \mathbb{R} \mid y \le 0\}. \tag{3.138}$$

The reset map is given by

$$R_2(x_1, x_2, t) = \left\{ (x_1', x_2') \in \mathfrak{B}_1(l) \mid \frac{dx_2'}{dt}(t) = \frac{(M-m)\frac{dx_2}{dt}(t) + 2m\frac{dx_1}{dt}(t)}{m+M} \right\}. \tag{3.139}$$

We define π as the projection acting on the behavior of A_2, $L(A_2)$, such that the input force u is eliminated. Now $L(A_1)$ and $\pi L(A_2)$ have the same type. The interconnection can be expressed as $L(A_1) \parallel \pi L(A_2)$.

3.3 Dynamic maps and state maps

3.3.1 Definition

The concept of *states* or *state variables* is present in almost all branches of dynamical systems theory. In areas as remotely connected as discrete event systems and linear time invariant systems we can observe that the notion of *states* is present. One may think that this is a mere coincidence, but this is not true. The different notions of states have something in common. They are all connected by the so called *state property* or the *axiom of state*. In short (and perhaps rather inaccurately), one can say that a quantity or variable possesses the state property (or satisfies the axiom of state) if it captures the necessary information about the evolution of the dynamical system. There is of course a more mathematically formal and rigor formulation of this property, for example in [PW98].

Following the earlier development in [JS03, RW97], our point of view, which is based on the behavioral approach, is that states are constructed out of the system trajectories (the behavior). In the behavioral point of view, the behavior (i.e. the collection of all possible trajectories) defines/identifies the system. Of course, it is required that the trajectories bear all information/variables on everything relevant to the discussion. Irrelevant variables/information, which in the case of linear time invariant behaviors are called *latent variables*, generally can be eliminated. See the previous section for a discussion on this issue.

To explain how states can be constructed from the system's trajectories, we need to introduce the concept of *dynamic maps* [JS04b].

Recall that a dynamical system Σ is defined by the triple $(\mathbb{T}, \mathbb{W}, \mathfrak{B})$. A dynamic map of the system Σ is a map that has $\mathfrak{B} \times \mathbb{T}$ as its domain. For example, $\phi : \mathfrak{B} \times \mathbb{T} \to \Phi$ is a dynamic map. We always assume that dynamic maps are surjective. Elements of the codomain of a dynamic map are called *points*. Since systems are characterized by their behavior and its type, hereafter we can also say that a dynamic map acts on a behavior whenever the type of the behavior is clear.

Example 3.27. Consider a behavior $\mathfrak{B} \in \mathfrak{L}_c^2$ given by

$$\mathfrak{B} := \left\{ (w_1, w_2) \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}) \mid \frac{dw_1}{dt} - w_2 = 0 \right\}. \tag{3.140}$$

Examples of a dynamic map defined on this system are $\phi_1 : \mathfrak{B} \times \mathbb{T} \to \mathbb{R}$, where

$$\phi_1(w, t) = w_1(t), \tag{3.141}$$

and $\phi_2 : \mathfrak{B} \times \mathbb{T} \to \mathbb{R}$, where

$$\phi_2(w, t) = \frac{\pi}{2} w_2(t - 1). \tag{3.142}$$

In fact, any surjective map that has $\mathfrak{B} \times \mathbb{T}$ as its domain can be taken as an example.

Example 3.28. Consider a finite state automaton $A = (X, E, T, X_{\mathrm{m}}, x_0)$. Suppose that the automaton is deterministic. Also suppose that the all states are reachable. Let $L(A)$ be the language generated by A, then the state can be thought of as a dynamic map acting on $L(A)$. To be precise, we can consider the map $\phi : L(A) \times \mathbb{Z}_+ \to X$, where $\phi(s, n)$ gives the terminal state after executing the first n events of s. If s has less than n events, then $\phi(s, n)$ is simply the last state reached by the execution.

Notation 3.29. Let $\phi : \mathfrak{B} \times \mathbb{T} \to \Phi$ be a dynamic map, and X and Y be subsets of $\mathfrak{B}$ and Φ respectively. A time-indexed dynamic map, notated as $\phi_t(\cdot)$, $t \in \mathbb{T}$, is defined as

$$\phi_t(w) := \phi(w; t) := \phi(w, t), w \in \mathfrak{B}.$$

Furthermore the following notations also apply in this book.

$$\phi_t(X) := \{y \in \Phi \mid \exists x \in X, \phi_t(x) = y\},$$
$$\phi_t^{-1}(Y) := \{x \in \mathfrak{B} \mid \exists y \in Y, \phi_t(x) = y\}.$$

The *point similarity operator* generated by the time-indexed dynamic map ϕ_t on $\mathfrak{B}$ is defined to be

$$\bar{\phi}_t : 2^{\mathfrak{B}} \to 2^{\mathfrak{B}},$$
$$\bar{\phi}_t(\cdot) := \phi_t^{-1}(\phi_t(\cdot)). \tag{3.143}$$

For any subset $X \subset \mathfrak{B}$, $\bar{\phi}_t(X)$ gives the largest subset of $\mathfrak{B}$ whose image under ϕ_t is $\phi_t(X)$.

We shall now define a partial ordering for dynamic maps.

Definition 3.30. Let ϕ and γ be two dynamic maps of $\mathfrak{B}$. We say $\phi \preccurlyeq \gamma$ if and only if for any $t_1, t_2 \in \mathbb{T}$, and $w_1, w_2 \in \mathfrak{B}$ the following implication holds.

$$(\gamma(w_1, t_1) = \gamma(w_2, t_2)) \Rightarrow (\phi(w_1, t_1) = \phi(w_2, t_2)). \tag{3.144}$$

Notice that given a behavior $\mathfrak{B}$ of type $(\mathbb{T}, \mathbb{W})$ and a dynamic map ϕ acting on it, we can also see ϕ as a projection acting on $\mathfrak{B}$ mapping it to a behavior of type $(\mathbb{T}, \Phi)$. That is, we can define

$$\phi(w)(t) := \phi(w, t), \forall w \in \mathfrak{B}, t \in \mathbb{T}. \tag{3.145}$$

Since dynamic maps are projections, they also inherit the partial ordering $\preccurlyeq$ that we defined for projections. The following lemma relates the partial ordering $\preccurlyeq$ and $\preceq$ for dynamic maps.

Lemma 3.31. Let ϕ and γ be two dynamic maps of $\mathfrak{B}$. The following relations hold.

$$(\phi \preccurlyeq \gamma) \Rightarrow (\phi \preceq \gamma). \tag{3.146}$$

Proof. Suppose that $\phi \preccurlyeq \gamma$. We need to prove that for any two trajectories $w_1, w_2 \in \mathfrak{B}$ such that $\gamma(w_1) = \gamma(w_2)$, we also have that $\phi(w_1) = \phi(w_2)$. Denote $\gamma(w_1)$ as w. Obviously, we have that for every $t \in \mathbb{T}$,

$$\gamma(w_1, t) = \gamma(w_2, t) = w(t). \tag{3.147}$$

This implies

$$\phi(w_1, t) = \phi(w_2, t), \forall t \in \mathbb{T}. \tag{3.148}$$

Thus $\phi(w_1) = \phi(w_2)$. To show that the converse is not necessarily true, suppose that $\phi \preceq \gamma$. Take two different trajectories $w_1, w_2 \in \mathfrak{B}$ and denote $\gamma(w_1) = w_1'$ and $\gamma(w_2) = w_2'$. Suppose that $w_1' \neq w_2'$ but there exist $\tau_1, \tau_2 \in \mathbb{T}$ where $w_1'(\tau_1) = w_2'(\tau_2)$. Denote $\phi(w_1) = w_1''$ and $\phi(w_2) = w_2''$. Since $\gamma(w_1) \neq \gamma(w_2)$, there is no restriction on w_1'' and w_2''. However, if $\phi \preccurlyeq \gamma$, then we should have that $w_1''(\tau_1) = w_2''(\tau_2)$. Thus, $\preccurlyeq$ is stronger than $\preceq$. □

The partial ordering $\preccurlyeq$ also induces an equivalence relation, which is denoted by $\approx$.

$$(\phi \approx \gamma) :\Leftrightarrow (\phi \preccurlyeq \gamma) \text{ and } (\phi \succcurlyeq \gamma). \tag{3.149}$$

Analogous to Lemma 3.13, we have the following result.

Lemma 3.32. Given a system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathfrak{B})$ and two dynamic maps ϕ and γ defined on it. Suppose $\phi : \mathfrak{B} \times \mathbb{T} \to \Phi$ and $\gamma : \mathfrak{B} \times \mathbb{T} \to \Gamma$. There exists a surjective mapping $\theta : \Phi \to \Gamma$ such that the following diagram commutes if and only if $\phi \succcurlyeq \gamma$.

$$\begin{tikzcd}
\mathfrak{B} \times \mathbb{T} \arrow[r, "\phi"] \arrow[dr, "\gamma"'] & \Phi \arrow[d, "\theta"] \\
 & \Gamma
\end{tikzcd}$$

As is the case with the partial ordering $\preceq$ (see Subsection 3.2.1), the partial ordering $\preccurlyeq$ can also be thought of as a 'measure' of information. This is apparent when we look at Lemma 3.32 above. If $\phi \succcurlyeq \gamma$, there always exists a function θ with which we can obtain the image of the dynamic map γ, given that of ϕ. Thus, in some sense, the image of ϕ contains more information than that of γ. Also notice that when $\phi \approx \gamma$, the function θ can always be taken as a one-to-one mapping. This suggests that the image of ϕ contains as much information as that of γ.

For any system Σ there exists a *unique maximal dynamic map* (up to $\approx$), namely the identity map (or any other isomorphism) from $\mathfrak{B} \times \mathbb{T}$ to itself. There also exists a *unique minimal dynamic map* (up to $\approx$), namely the one that maps $\mathfrak{B} \times \mathbb{T}$ to a singleton.

The dynamic maps of a system Σ forms a lattice structure. This is because there are two binary operations, $\wedge$ (called *'meet'* or *'greatest lower bound'*) and $\vee$ (called *'join'* or *'least upper bound'*) defined on them with some special properties [Büc89]. For any ϕ and γ dynamic maps of Σ the following hold.

(meet1) $\phi \succcurlyeq (\phi \wedge \gamma)$ and $\gamma \succcurlyeq (\phi \wedge \gamma)$.

(meet2) For any π such that $\pi \preccurlyeq \phi$ and $\pi \preccurlyeq \gamma$, $\pi \preccurlyeq (\phi \wedge \gamma)$.

(join1) $\phi \preccurlyeq (\phi \vee \gamma)$ and $\gamma \preccurlyeq (\phi \vee \gamma)$.

(join2) For any π such that $\pi \succcurlyeq \phi$ and $\pi \succcurlyeq \gamma$, $\pi \preccurlyeq (\phi \vee \gamma)$.

These operations are uniquely defined on the equivalence classes generated by $\approx$.

$$((\phi \approx \phi') \text{ and } (\gamma \approx \gamma')) \Rightarrow ((\phi \wedge \gamma) \approx (\phi' \wedge \gamma')) , \tag{3.150}$$

$$((\phi \approx \phi') \text{ and } (\gamma \approx \gamma')) \Rightarrow ((\phi \vee \gamma) \approx (\phi' \vee \gamma')) . \tag{3.151}$$

Take any dynamic maps $\phi : \mathfrak{B} \times \mathbb{T} \to \Phi$ and $\gamma : \mathfrak{B} \times \mathbb{T} \to \Gamma$. Suppose that I_ϕ and I_γ are the equivalence relation generated by ϕ and γ, that is, for any $w_1, w_2 \in \mathfrak{B}$ and $t_1, t_2 \in \mathbb{T}$,

$$((w_1, t_1), (w_2, t_2)) \in I_\phi \Leftrightarrow \phi(w_1, t) = \phi(w_2, t), \tag{3.152}$$

$$((w_1, t_1), (w_2, t_2)) \in I_\gamma \Leftrightarrow \gamma(w_1, t) = \gamma(w_2, t). \tag{3.153}$$

Notice that the equivalence relations I_ϕ and I_γ uniquely identify the classes oh dynamic maps equivalent to ϕ and γ.

The class of dynamic maps corresponding to $\phi \vee \gamma$ is identified by the equivalence relation $I_{\phi\vee\gamma}$, where

$$I_{\phi\vee\gamma} := I_\phi \cap I_\gamma. \tag{3.154}$$

A representation of this class can be constructed by augmenting ϕ and γ. Thus $(\phi \vee \gamma) : \mathfrak{B} \times \mathbb{T} \to (\Phi \times \Gamma)$, where

$$(\phi \vee \gamma)(w, t) := (\phi(w, t), \gamma(w, t)). \tag{3.155}$$

The class of dynamic maps corresponding to $\phi \wedge \gamma$ is identified by the equivalence relation $I_{\phi\wedge\gamma}$, where

$$I_{\phi\wedge\gamma} := (I_\phi \cup I_\gamma)^{\mathrm{e}} . \tag{3.156}$$

The symbol $(R)^{\mathrm{e}}$ denotes the *transitive closure* of the relation R.

$$(R)^{\mathrm{e}} := R \cup (R \circ R) \cup (R \circ R \circ R) \cup \cdots \tag{3.157}$$

The fact that these constructions satisfy the special properties for meet and join follows from the fact that $I_{\phi\vee\gamma}$ is the largest equivalence relation contained in both I_ϕ and I_γ, and that $I_{\phi\wedge\gamma}$ is the smallest equivalence relation containing both I_ϕ and I_γ.

Refer to Figure 3.6 for an illustration of the discussion about the lattice structure of the dynamic maps. It should be noticed that the lattice shown in the figure does not really reflect the actual structure of the lattice of dynamic maps, rather it only serves as an illustration.

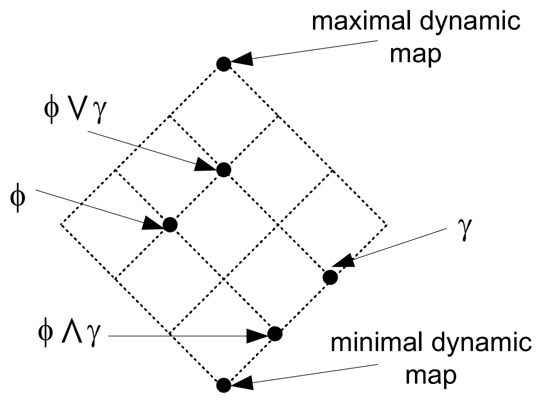


Figure 3.6: Illustration for the lattice structure of the dynamic maps.

3.3.2 Some subclasses of dynamic maps

We can introduce some properties to define subclasses of dynamic maps.

Definition 3.33. A dynamic map $\phi : (\mathfrak{B}, \mathbb{T}) \to \Phi$ is called **Markovian** or is said to possess the **Markov property** if for any $w_1, w_2 \in \mathfrak{B}$, $\tau_1, \tau_2 \in \mathbb{T}$ and $\delta \in \mathbb{T}^+$, the following implication holds.

$$\begin{aligned}\{(\phi(w_1,\tau_1) = \phi(w_2,\tau_2)) \text{ and } (w_1|_{\tau_1<t\le\tau_1+\delta} = w_2|_{\tau_2<t\le\tau_2+\delta})\} \Rightarrow \\ \{\phi(w_1,\tau_1+\delta) = \phi(w_2,\tau_2+\delta)\}. \end{aligned} \quad (3.158)$$

The expression $(w_1|_{\tau_1<t\le\tau_1+\delta} = w_2|_{\tau_2<t\le\tau_2+\delta})$ means for all $t \in \mathbb{T}$ such that $\tau_1 < t \le \tau_1 + \delta$,

$$w_1(t) = w_2(t - \tau_1 + \tau_2). \quad (3.159)$$

In words, a dynamic map is Markovian if whenever two trajectories that are not distinguishable by the dynamic map at a certain time and they proceed with the same segment of trajectory, they should remain indistinguishable. Notice that both the maximal and minimal dynamic maps are Markovian.

Example 3.34. The dynamic map ϕ_1 in Example 3.27 and ϕ in Example 3.28 are Markovian. The dynamic map ϕ_2 in Example 3.27 is not Markovian.

Other important subclasses of dynamic maps are characterized by the following two properties.

Definition 3.35. A dynamic map $\phi : (\mathfrak{B}, \mathbb{T}) \to \Phi$ is called **past-induced** if for any $w_1, w_2 \in \mathfrak{B}$ and $\tau \in \mathbb{T}$, the following implication holds.

$$(w_1|_{t\le\tau} = w_2|_{t\le\tau}) \Rightarrow (\phi(w_1,\tau) = \phi(w_2,\tau)). \quad (3.160)$$

Definition 3.36. A dynamic map $\phi : (\mathfrak{B}, \mathbb{T}) \to \Phi$ is called **future-induced** if for any $w_1, w_2 \in \mathfrak{B}$ and $\tau_1, \tau_2 \in \mathbb{T}$, the following implication holds.

$$(w_1|_{t>\tau_1} = w_2|_{t>\tau_2}) \Rightarrow (\phi(w_1, \tau_1) = \phi(w_2, \tau_2)) .$$

The expression $(w_1|_{t>\tau_1} = w_2|_{t>\tau_2})$ means for all $t \in \mathbb{T}$ such that $\tau_1 < t$,

$$w_1(t) = w_2(t - \tau_1 + \tau_2). \tag{3.161}$$

The definition of past and future inducedness have appeared earlier in the literatures, e.g. [Sch84].

Lemma 3.37. Let ϕ and γ be two dynamic maps of Σ. Suppose that $\phi \succcurlyeq \gamma$. If ϕ is a past-induced (resp. future-induced), then γ is also past-induced (resp. future-induced).

Proof. We shall produce the proof for the past-inducedness property, as that for the future-inducedness follows analogously. Suppose that $\phi \succcurlyeq \gamma$ and ϕ is past-induced. We have that for any $w_1, w_2 \in \mathfrak{B}$ and $\tau \in \mathbb{T}$, the following implication holds.

$$(w_1|_{t\leq\tau} = w_2|_{t\leq\tau}) \Rightarrow (\phi(w_1, \tau) = \phi(w_2, \tau)) . \tag{3.162}$$

However, we also have that

$$(\phi(w_1, \tau) = \phi(w_2, \tau)) \Rightarrow (\gamma(w_1, \tau) = \gamma(w_2, \tau)) . \tag{3.163}$$

Together they imply

$$(w_1|_{t\leq\tau} = w_2|_{t\leq\tau}) \Rightarrow (\gamma(w_1, \tau) = \gamma(w_2, \tau)) , \tag{3.164}$$

which means that γ is also past-induced. □

Example 3.38. The dynamic maps ϕ_1 and ϕ_2 in Example 3.27 and ϕ in Example 3.28 are past induced. Furthermore, ϕ_1 is future induced, ϕ_2 is not future induced, and ϕ is not necessarily future induced, depending on the automaton A.

It can be proven that for any system Σ, there exists a *unique maximal past-induced dynamic map* (up to $\approx$). In the literatures, this map is called the *Nerode state construction*. By symmetry, there also exists a *unique maximal future-induced dynamic map* (up to $\approx$). We shall call this map the *Dual Nerode state construction*.

The Nerode state construction for a system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathfrak{B})$ is characterized by the equivalence relation R where for any $w_1, w_2 \in \mathfrak{B}$ and $\tau \in \mathbb{T}$,

$$(w_1, w_2) \in R \Leftrightarrow (w_1|_{t\leq\tau} = w_2|_{t\leq\tau}) . \tag{3.165}$$

Similarly, the Dual Nerode state construction is characterized by the family of equivalence relations R where for any $w_1, w_2 \in \mathfrak{B}$ and $\tau_1, \tau_2 \in \mathbb{T}$,

$$(w_1, w_2) \in R \Leftrightarrow (w_1|_{t>\tau_1} = w_2|_{t>\tau_2}) . \tag{3.166}$$

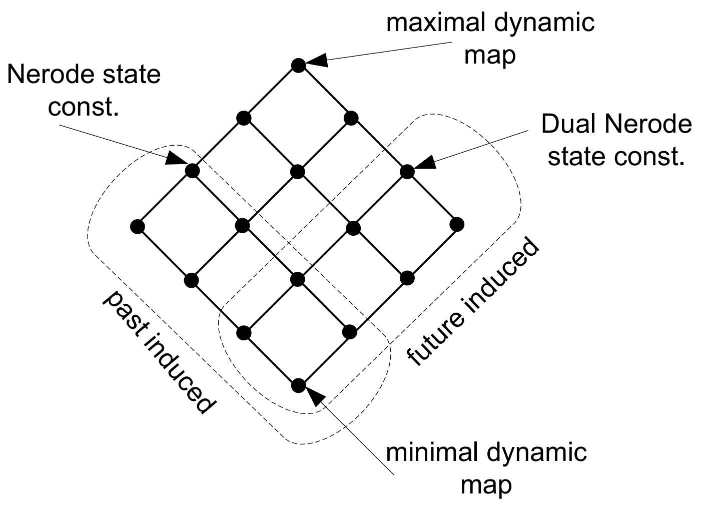


Figure 3.7: An illustration of the past and future inducedness property.

Unique minimal past-induced and future-induced dynamic maps also exist. As a consequence of Lemma 3.37, they coincide with the minimal dynamic map. Moreover, it can also be proven that if ϕ and γ are the Nerode state construction and the Dual Nerode state construction of Σ, then $(\phi \vee \gamma)$ is the maximal dynamic map. The discussion in this paragraph can be summarized into the illustration in Figure 3.7.

The most important subclass of dynamic maps that we discuss is arguably the so called *state maps*. State maps are dynamic maps that has the *state property*. This property is generally known and discussed in many basic systems theory literatures. It is also known as the *axiom of state*. See, for example, [PW98].

Definition 3.39. A **state map** $x : (\mathfrak{B}, \mathbb{T}) \to \mathbb{X}$ is a dynamic map such that, for any $w_1, w_2 \in \mathfrak{B}$ and $\tau_1, \tau_2 \in \mathbb{T}$, the following implication holds.

$$\{x(w_1, \tau) = x(w_2, \tau)\} \Rightarrow \{w_3 := (w_1 \wedge_{\tau_2}^{\tau_1} w_2) \in \mathfrak{B}\}. \qquad (3.167)$$

The set $\mathbb{X}$ is called the *state space* of the state map x.

The concept of state maps for LTI behaviors has appeared, for example, in [RW97]. Since state maps are basically dynamic maps, they also inherit the ordering $\preccurlyeq$. Elements of the codomain of a state map are called *states*. Thus *states* are a special case of *points*. The following lemma says something about the structure of state maps in the lattice of dynamic maps.

Lemma 3.40. Let ϕ and γ be two dynamic maps of Σ. Suppose that $\phi \preccurlyeq \gamma$. If ϕ is a state map, then γ is also a state map.

Proof. Suppose that ϕ is a state map, then for any $w_1, w_2 \in \mathfrak{B}$ and $\tau_1, \tau_2 \in \mathbb{T}$, the following implication holds.

$$\{\phi(w_1, \tau_1) = \phi(w_2, \tau_2)\} \Rightarrow \{w_3 := (w_1 \wedge_{\tau_2}^{\tau_1} w_2) \in \mathfrak{B}\}. \tag{3.168}$$

We also have that

$$\{\gamma(w_1, \tau_1) = \gamma(w_2, \tau_2)\} \Rightarrow \{\phi(w_1, \tau_1) = \phi(w_2, \tau_2)\}. \tag{3.169}$$

Therefore

$$\{\gamma(w_1, \tau_1) = \gamma(w_2, \tau_2)\} \Rightarrow \{w_3 := (w_1 \wedge_{\tau_2}^{\tau_1} w_2) \in \mathfrak{B}\}.$$

□

In fact, we can always guarantee that for any system Σ, there always exists a state map, as we can (trivially) prove that the maximal dynamic map, the Nerode state construction and the Dual Nerode state construction are state maps. It is obvious that the maximal dynamic map also acts as the unique maximal state map.

The minimal state map(s) is a more interesting object to study. For example, about its uniqueness. Some necessary and sufficient conditions for the existence of a unique minimal state map (up to $\approx$) were given in [JS03]. If such state map exists, we shall call it the *canonical minimal* state map. It is the only state map (again, up to $\approx$) that satisfies the canonical state property, where the implication in (3.167) is replaced with a biimplication [JS03]. Furthermore, we can relate the canonical minimal state map with the past and future inducedness properties through the following result.

Theorem 3.41. Take any system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathfrak{B})$. Let us denote the Nerode state construction and the Dual Nerode state construction as ϕ and γ respectively. The system Σ admits a canonical minimal state map, if and only if $\phi \wedge \gamma$ is a state map. Moreover, if $\phi \wedge \gamma$ is a state map, it is also the canonical minimal state map.

Proof. (if) Assume that $(\phi \wedge \gamma)$ is a state map. We need to show that it is the canonical minimal state map. Denote $(\phi \wedge \gamma) =: \xi$. This means that we have to show that for any $w_1, w_2 \in \mathfrak{B}$ and $\tau_1, \tau_2 \in \mathbb{T}$, the following implication holds.

$$\{\xi(w_1, \tau_1) = \xi(w_2, \tau_2)\} \Leftarrow \{w_3 := (w_1 \wedge_{\tau_2}^{\tau_1} w_2) \in \mathfrak{B}\}. \tag{3.170}$$

Notice that $\phi(w_1, \tau_1) = \phi(w_3, \tau_1)$ and $\gamma(w_2, \tau_2) = \gamma(w_3, \tau_1)$. From the definition of ξ, it follows that $\xi(w_1, \tau_1) = \xi(w_2, \tau_2)$. (only if) Suppose that a canonical minimal state map exists. Denote it by ξ. This means for any $w_1, w_2 \in \mathfrak{B}$ and $\tau_1, \tau_2 \in \mathbb{T}$, the following biimplication holds.

$$\{\xi(w_1, \tau_1) = \xi(w_2, \tau_2)\} \Leftrightarrow \{w_3 := (w_1 \wedge_{\tau_2}^{\tau_1} w_2) \in \mathfrak{B}\}. \tag{3.171}$$

Notice that the following two implications are always satisfied.

$$\{\phi(w_1, \tau_1) = \phi(w_2, \tau_2)\} \Rightarrow \{w_3 := (w_1 \wedge_{\tau_2}^{\tau_1} w_2) \in \mathfrak{B}\}, \tag{3.172}$$
$$\{\gamma(w_1, \tau_1) = \gamma(w_2, \tau_2)\} \Rightarrow \{w_3 := (w_1 \wedge_{\tau_2}^{\tau_1} w_2) \in \mathfrak{B}\}. \tag{3.173}$$

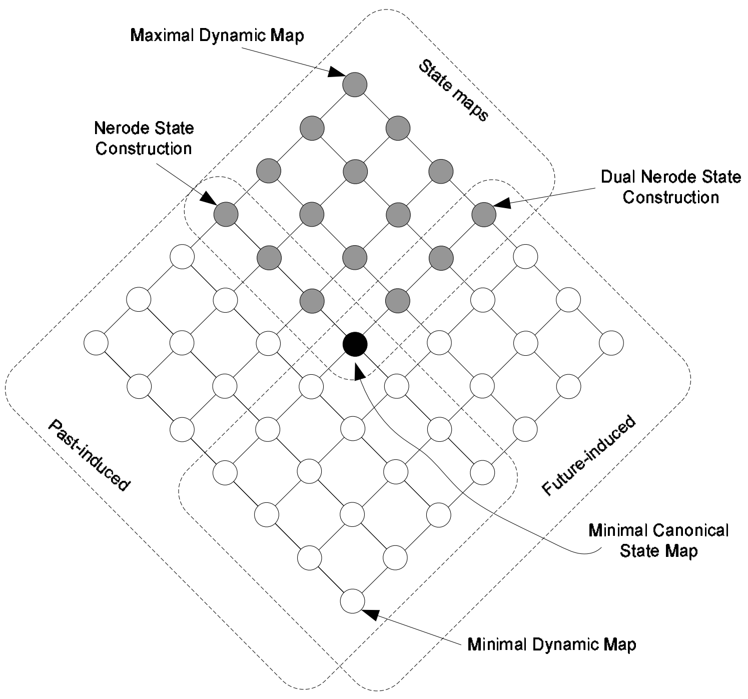

Maximal Dynamic Map
State maps
Nerode State Construction
Dual Nerode State Construction
Past-induced
Future-induced
Minimal Canonical State Map
Minimal Dynamic Map

Figure 3.8: An illustration of the relations between maps in the lattice, when the canonical minimal state map exists. White circles represent dynamic maps. Shaded circles represent state maps. The black circle represents the minimal state map.

Together they imply $\phi \succcurlyeq \xi$ and $\gamma \succcurlyeq \xi$. Consequently, we also have that $(\phi \wedge \gamma) \succcurlyeq \xi$. By Lemma 3.40, this implies that $(\phi \wedge \gamma)$ is a state map. $\square$

An immediate consequence of Theorem 3.41 is the following corollary.

Corollary 3.42. Take any system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathfrak{B})$. If the system admits a canonical minimal state map ξ, then ξ is the only state map that has both the past-inducedness and future-inducedness properties.

Figure 3.8 and 3.9 illustrate the results about state maps we have discussed so far. As in previous illustrations, in both figures the lattice of dynamic maps is portrayed as a planar finite lattice, which is inaccurate. But still, they capture the main story. Figure 3.8 depicts the situation where the canonical minimal state map exists, and in Figure 3.9, it does not.

Example 3.43. The dynamic map ϕ in Example 3.28 is a state map. Thus, the term 'states' that we use for automaton conform with the concept of state maps.

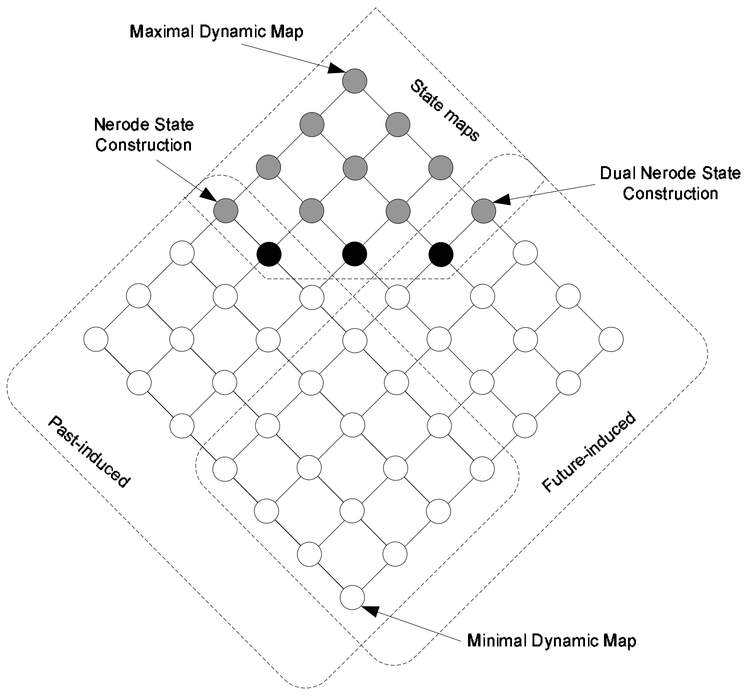


Figure 3.9: An illustration of the relations between maps in the lattice, when the canonical minimal state map does not exist. White circles represent dynamic maps. Shaded circles represent state maps. The black circles represent the minimal state maps.

When we discuss the concept of dynamic maps and state maps for continuous time LTI systems, it is most convenient to restrict ourselves to the class $\overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$. Consider the following example as an illustration.

Example 3.44. Consider a simple behavior $\mathfrak{B} \in \overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{1}$ given by the following kernel representation.

$$\mathfrak{B} := \left\{ (u, y) \mid \frac{dy}{dt} = 0 \right\}. \tag{3.174}$$

Consider the dynamic map $x : \mathfrak{B} \times \mathbb{R} \rightarrow \mathbb{R}$ defined as

$$x((y, u), t) := y(t). \tag{3.175}$$

From our experience with LTI systems we would expect x to be a state map of the behavior. This is indeed true. To see this, notice that every $w \in \mathfrak{B}$ must be a constant function for all $t \in \mathbb{R}$. In fact, we can also see that x is also the canonical minimal state map of the behavior.
However, if we had chosen $\mathfrak{B}$ to be the collection of strong solutions, i.e. $\mathfrak{B} \in \mathfrak{L}_{\mathrm{c}}^{1}$ with the same kernel representation, then x is not a state map. This is because to concatenate two infinitely differentiable functions we need to have that all the derivatives of any finite order are matched. Therefore, in this case a state map of $\mathfrak{B}$ should contain the information about all the derivatives of finite order of both y and u.
Similarly, if we had chosen $\mathfrak{B}$ to be the collection of weak solutions, i.e. $\mathfrak{B} \in \bar{\mathfrak{L}}_{\mathrm{c}}^{1}$ with the same kernel representation, then the state map x is not well defined. This is because the immediate value of a trajectory in $\mathfrak{L}_{1}^{\mathrm{loc}}(\mathbb{R}, \mathbb{R})$ at a particular time is immaterial. By requiring left-continuity, this technical difficulty is averted.

By the exposition in the example above, we can formulate the following result.

Theorem 3.45. Given a behavior $\mathfrak{B} \in \mathfrak{L}_{\mathrm{c}}^{\mathrm{q}}$ given as

$$\mathfrak{B} = \{w \mid R\left(\frac{d}{dt}\right) w = 0\}. \tag{3.176}$$

This behavior admits a canonical minimal state map, namely the one defined by

$$x(w, t) := \begin{bmatrix} w(t) \\ \frac{d}{dt} w(t) \\ \frac{d^2}{dt^2} w(t) \\ \vdots \end{bmatrix}. \tag{3.177}$$

Proof. First, we prove that x is a state map. Take any $w_1, w_2 \in \mathfrak{B}$ and $t_1, t_2 \in \mathbb{R}$ such that $x(w_1, t_1) = x(w_2, t_2)$. We need to prove that $w := w_1 \wedge_{t_2}^{t_1} w_2$ is also in $\mathfrak{B}$. However, this fact follows immediately from the fact that
(i) w is infinitely differentiable and

(ii) w_1 and w_2 are strong solutions of the differential equation in (3.176). To prove that x is in fact the canonical minimal state map, take any $w_1, w_2 \in \mathfrak{B}$ and $t_1, t_2 \in \mathbb{R}$ such that $w := w_1 \wedge_{t_2}^{t_1} w_2$. We need to prove that $x(w_1, t_1) = x(w_2, t_2)$. However, this fact follows immediately from the fact that w needs to be infinitely differentiable. □

Behaviors in $\overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$ also admit canonical minimal state map, namely the observable state space representation (see Section 6.4 in [PW98]).

Theorem 3.46. Given a behavior $\mathfrak{B} \in \overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$ given as

$$\mathfrak{B} = \{w \mid R\left(\frac{d}{dt}\right) w = 0\}. \tag{3.178}$$

Suppose that we split $\mathbf{w}$ into $\mathbf{y}$ and $\mathbf{u}$ such that we have the following state space representation

$$\frac{d}{dt}x = Ax + Bu, \tag{3.179a}$$

$$y = Cx + Du. \tag{3.179b}$$

If the state $\mathbf{x}$ is observable from $\mathbf{u}$ and $\mathbf{y}$, then $\mathbf{x}$ is a canonical minimal state map of $\mathfrak{B}$.

Proof. The state space representation (3.179) can be written in the kernel representation as

$$\begin{bmatrix} \frac{d}{dt}I - A & -B & 0 \\ C & D & -I \end{bmatrix} \begin{bmatrix} x \\ u \\ y \end{bmatrix} = 0. \tag{3.180}$$

Since the representation is observable, from Theorem 3.19 we know that

$$\begin{bmatrix} \xi I - A \\ C \end{bmatrix}$$

has full column rank for all $\xi \in \mathbb{C}$. Consequently, there exists a unimodular matrix U such that

$$U(\xi) \begin{bmatrix} \xi I - A \\ C \end{bmatrix} = \begin{bmatrix} I \\ 0 \end{bmatrix}. \tag{3.181}$$

If we premultiply the kernel representation (3.180) with U, we get a kernel representation of the form

$$\begin{bmatrix} I & R'_{11}\left(\frac{d}{dt}\right) & R'_{12}\left(\frac{d}{dt}\right) \\ 0 & R'_{21}\left(\frac{d}{dt}\right) & R'_{22}\left(\frac{d}{dt}\right) \end{bmatrix} \begin{bmatrix} x \\ u \\ y \end{bmatrix} = 0. \tag{3.182}$$

This representation shows that $\mathbf{x}$ is a dynamic map of $\mathfrak{B}$, that is

$$x = -R'_{11}\left(\frac{d}{dt}\right) u - R'_{12}\left(\frac{d}{dt}\right) y. \tag{3.183}$$

The fact that $\mathbf{x}$ has the state property is quite a standard result and the proof can be found, for example in [PW98]. We shall now prove the $\mathbf{x}$ is canonical minimal. Suppose that $w_1 = (u_1, y_1)$ and $w_2 = (u_2, y_2)$ are both elements of $\mathfrak{B}$ such that $w := w_1 \wedge_{\tau_2}^{\tau_1} w_2$ is also in $\mathfrak{B}$, for some $\tau_1, \tau_2 \in \mathbb{R}$. Denote x, x_1 and x_2 as the trajectory of the state map corresponding to w, w_1 and w_2 respectively. We shall prove that $\mathbf{x}$ is canonical minimal by showing that $x_1(\tau_1) = x_2(\tau_2)$. Equation (3.179a) implies that state trajectories are continuous. Thus the following relations hold.

$$\begin{aligned} x_1(\tau_1) &= \lim_{t \uparrow \tau_1} x_1(t) = \lim_{t \uparrow \tau_1} x(t) \\ &= \lim_{t \downarrow \tau_1} x(t) = \lim_{t \downarrow \tau_2} x_2(t) = x_2(\tau_2). \end{aligned} \tag{3.184}$$

□

The dimension of the state space in the observable state space representation is called the *McMillan degree* of the behavior [PW98].

The analog of Theorem 3.46 for behaviors in $\mathfrak{L}_\mathrm{d}^\mathrm{q}$ can be stated as follows.

Theorem 3.47. Given a behavior $\mathfrak{B} \in \mathfrak{L}_\mathrm{d}^\mathrm{q}$ given as

$$\mathfrak{B} = \{w \mid R(\sigma)\, w = 0\}. \tag{3.185}$$

Suppose that we split $\mathbf{w}$ into $\mathbf{y}$ and $\mathbf{u}$ such that we have the following state space representation

$$x(k+1) = Ax(k) + Bu(k+1), \tag{3.186a}$$
$$y(k+1) = Cx(k) + Du(k+1). \tag{3.186b}$$

If the state $\mathbf{x}$ is observable from $\mathbf{u}$ and $\mathbf{y}$, then $\mathbf{x}$ is a canonical minimal state map of $\mathfrak{B}$.

Proof. The state space representation (3.186) can be written in the kernel representation as

$$\begin{bmatrix} \sigma I - A & -\sigma B & 0 \\ C & \sigma D & -\sigma I \end{bmatrix} \begin{bmatrix} x \\ u \\ y \end{bmatrix} = 0. \tag{3.187}$$

As in the proof of Theorem 3.46, since the representation is observable, we know that there exists a unimodular matrix U such that

$$U(\xi) \begin{bmatrix} \xi I - A \\ C \end{bmatrix} = \begin{bmatrix} I \\ 0 \end{bmatrix}. \tag{3.188}$$

Following the same reasoning as in the proof of Theorem 3.46, we can show that $\mathbf{x}$ is a dynamic map of $\mathfrak{B}$. That is, there exist R'_{11} and R'_{22} (as in (3.183)) such that

$$x = -R'_{11}(\sigma)\, u - R'_{12}(\sigma)\, y. \tag{3.189}$$

To show that $\mathbf{x}$ has the state property, consider two trajectories (x_1, u_1, y_1) and (x_2, u_2, y_2) that satisfy (3.186). We need to show that for any $t_1, t_2 \in \mathbb{Z}$ such that

$$x_1(t_1) = x_2(t_2), \tag{3.190}$$

the trajectory $(u_1 \wedge_{t_2}^{t_1} u_2, y_1 \wedge_{t_2}^{t_1} y_2) \in \mathfrak{B}$. That is, there is a state trajectory $\tilde{x}$ such that $(\tilde{x}, u_1 \wedge_{t_2}^{t_1} u_2, y_1 \wedge_{t_2}^{t_1} y_2)$ satisfies (3.186). We construct $\tilde{x}$ as

$$\tilde{x} := x_1 \wedge_{t_2}^{t_1} x_2. \tag{3.191}$$

Define $\tilde{u} := u_1 \wedge_{t_2}^{t_1} u_2$ and $\tilde{y} := y_1 \wedge_{t_2}^{t_1} y_2$. Thus, we need to prove that the trajectory $(\tilde{x}, \tilde{u}, \tilde{y})$ satisfies (3.186). For $k < t_1$ and $k > t_1$, we can see that $(\tilde{x}, \tilde{u}, \tilde{y})$ satisfies (3.186) by construction. For $k = t_1$, we need to show that

$$\tilde{x}(t_1 + 1) = A\tilde{x}(t_1) + B\tilde{u}(t_1 + 1), \tag{3.192a}$$
$$\tilde{y}(t_1 + 1) = C\tilde{x}(t_1) + D\tilde{u}(t_1 + 1), \tag{3.192b}$$

or equivalently

$$x_2(t_2 + 1) = Ax_1(t_1) + Bu_2(t_2 + 1), \tag{3.193a}$$
$$y_2(t_2 + 1) = Cx_1(t_1) + Du_2(t_2 + 1). \tag{3.193b}$$

But this is also true, since $x_1(t_1) = x_2(t_2)$. Now we shall prove that $\mathbf{x}$ is canonical minimal. We do it by showing that $\mathbf{x}$ is both past induced and future induced. Take any two trajectories (x_1, u_1, y_1) and (x_2, u_2, y_2) that satisfy (3.186). To demonstrate past inducedness we need to show that if $t \in \mathbb{Z}$ is such that for all $k \leq t$,

$$u_1(k) = u_2(k), \tag{3.194}$$
$$y_1(k) = y_2(k), \tag{3.195}$$

then $x_1(t) = x_2(t)$. Because of linearity, this is equivalent to showing that for any (x, u, y) satisfying (3.186) such that for all $k \leq t$,

$$u(t) = 0, \tag{3.196}$$
$$y(t) = 0, \tag{3.197}$$

we should have $x(t) = 0$. Substituting (3.196) and (3.197) to (3.186), we infer that for all $k < t$ the following equations hold.

$$x(k + 1) = Ax(k), \tag{3.198}$$
$$0 = Cx(k). \tag{3.199}$$

Because of observability, we can conclude that $x(t) = 0$. Following the same reasoning, to demonstrate future inducedness we need to show that for any (x, u, y) satisfying (3.186) such that for all $k > t$,

$$u(t) = 0, \tag{3.200}$$
$$y(t) = 0, \tag{3.201}$$

we should have $x(t) = 0$. We know that for all $k \geq t$ the following equations hold.

$$x(k+1) = Ax(k), \tag{3.202}$$
$$0 = Cx(k). \tag{3.203}$$

Again, because of observability, we can conclude that $x(t) = 0$. $\square$

As in the case of $\overrightarrow{\mathfrak{L}}^{\mathrm{q}}_{\mathrm{c}}$, the dimension of the state space in the observable state space representation is called the *McMillan degree* of the behavior [PW98]. Notice that in the case of behaviors in $\overrightarrow{\mathfrak{L}}^{\mathrm{q}}_{\mathrm{c}}$, the dynamic maps of interest typically take the form of linear combination of the trajectory and its derivatives. In the case of $\mathfrak{L}^{\mathrm{q}}_{\mathrm{d}}$, the dynamic maps of interest typically take the form of linear combination of the trajectory and its shifted version. State maps of these forms are called *linear differential map* and *linear difference maps* respectively.

Remark 3.48. The proposed state space representation (3.186) differs from the usual representation

$$x(k+1) = Ax(k) + Bu(k), \tag{3.204a}$$
$$y(k) = Cx(k) + Du(k). \tag{3.204b}$$

This is because we can prove that $\mathbf{x}$ in the state space representation (3.204) is not a state map. It would be a state map if concatenation was defined differently. Namely, if $w_1 \wedge^{t_1}_{t_2} w_2$ was defined as

$$\left(w_1 \wedge^{t_1}_{t_2} w_2\right)(t) = \begin{cases} w_1(t), & t < t_1 \\ w_2(t - t_1 + t_2), & t \geq t_1 \end{cases}. \tag{3.205}$$

Notice the difference in the inequality signs with the actual definition. The reason we do not use this definition is because the definition that we are using now is more convenient to use with hybrid systems.
Nevertheless, despite of the difference between (3.186) and (3.204), we have seen that, for example, in terms of characterization of observability, they both have the same characterization.

As the final result in this chapter, we present the relation between the canonical minimal state map and full interconnection.

Theorem 3.49. Let two behaviors $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be of the same type $(\mathbb{T}, \mathbb{W})$. Furthermore, assume that both behaviors are not disjoint and they admit canonical minimal state maps, namely ϕ_1 and ϕ_2. The behavior $\mathfrak{B} := \mathfrak{B}_1 \parallel \mathfrak{B}_2$ has $\phi_1|_{\mathfrak{B}\times\mathbb{T}} \vee \phi_2|_{\mathfrak{B}\times\mathbb{T}}$ as its canonical minimal state map. The symbols $\phi_i|_{\mathfrak{B}\times\mathbb{T}}$, $i = 1, 2$, signifies the dynamic map ϕ_i restricted to $\mathfrak{B} \times \mathbb{T}$.

Proof. For brevity we denote $\phi := \phi_1|_{\mathfrak{B}\times\mathbb{T}} \vee \phi_2|_{\mathfrak{B}\times\mathbb{T}}$. Take any two trajectories w_1 and w_2 in $\mathfrak{B}$ and any $\tau_1, \tau_2 \in \mathbb{T}$. Notice that $\phi(w_1, \tau_1) = \phi(w_2, \tau_2)$ is equivalent to

$$\phi_1(w_1, \tau_1) = \phi_1(w_2, \tau_2), \tag{3.206}$$
$$\phi_2(w_1, \tau_1) = \phi_2(w_2, \tau_2). \tag{3.207}$$

These relations are equivalent to

$$w_1 \wedge_{\tau_2}^{\tau_1} w_2 \in \mathfrak{B}_1, \tag{3.208}$$

$$w_1 \wedge_{\tau_2}^{\tau_1} w_2 \in \mathfrak{B}_2. \tag{3.209}$$

Thus $\phi(w_1, t) = \phi(w_2, t)$ is equivalent to $w_1 \wedge_{\tau_2}^{\tau_1} w_2 \in \mathfrak{B}$. Therefore ϕ is a canonical minimal state map of $\mathfrak{B}$. □

3.4 Summary

In this chapter we mainly discuss the concept of interconnection of behaviors. We begin with the notion of full interconnection, that is, interconnection of behaviors with the same type. In this case, interconnection simply means set-theoretic intersection.

We then proceed to discuss interconnection of behaviors with different types. To do so, first the concept of behavior projection is introduced. A projection is a mapping that maps a behavior to another behavior, possibly with a different type. With projections, two behaviors of different types can be mapped to two other behaviors with the same type. From here, the interconnection is done in the usual way.

In the last section, the concept of dynamic maps is introduced. We show that the dynamic maps of a system has a lattice structure. We also introduce some subclasses of dynamic maps, which are characterized by special properties. The most important subclass of dynamic maps is the state maps. State maps are dynamic maps that possess the state property.

4

Control as interconnection

"...and now for something completely different, a man with three noses." - Monty Python's Flying Circus

4.1 Introduction

In this chapter we discuss about control problems in the behavioral setting. This is not the first time control problem is discussed in the behavioral framework. We would like to refer the reader to the literature, in which control problems for linear systems [Wil97, TW99, Tre99, BT02, Bel03, WBJT], nonlinear systems [PP04], discrete event systems [Sme87, Sme89], and hybrid systems [MR99, JSS03]. This list of literature is by no means exhaustive.

There are many variants of control problems, which share the same salient feature. Control problems in the behavioral setting can be expressed as follows.

Control problem Given a system called the *plant*. The problem is to find a behavior (called the *controller*), which when interconnected with the plant behavior in a certain manner yields some desired properties, usually given in terms of another behavior (called the *specification*).

We shall discuss some variants of the control problem and their conditions of solvability. A control problem is solvable if it is possible to find a controller that meets the requirement of the problem.

4.1.1 Full interconnection control problems

Full interconnection control problems are the simplest variant. The behaviors involved are of the same type, so that no projection is necessary. The problem is typically expressed as the following.

Problem 4.1. Given the plant $\mathcal{P}$ and the specification $\mathcal{S}$, find a controller $\mathcal{C}$ such that

$$\mathcal{P} \parallel \mathcal{C} = \mathcal{S}. \tag{4.1}$$

Notice that (4.1) implicitly suggests that all the behaviors involved are of the same type. Examples of control problems of this type are state feedback control problem (where the controller can use all the plant variables) and supervisory control of discrete event systems where all events are observable and controllable [Ram87, RW89, CL99].

A controller $\mathcal{C}$ is said to *achieve* the desired specification $\mathcal{S}$, if it satisfies (4.1). The specification $\mathcal{S}$ is said to be *achievable*, if there exists a controller that achieves it. In the literature, the terms *implement* and *implementability* are often used instead of *achieve* and *achievable.*

From the definition of full interconnection, it is clear that $\mathcal{S}$ is achievable if and only if $\mathcal{S} \subset \mathcal{P}$. In fact, if $\mathcal{S}$ is achievable, it can be seen easily that $\mathcal{C} := \mathcal{S}$ achieves $\mathcal{S}$.

4.1.2 Partial interconnection control problems

This variant of control problems involves behaviors of different types.

Problem 4.2. The plant $\mathcal{P}$ is a given behavior of type $(\mathbb{T}_P, \mathbb{W}_P)$, and the desired specification $\mathcal{S}$ is a given behavior of type $(\mathbb{T}_S, \mathbb{W}_S)$. The candidate controllers are of type $(\mathbb{T}_C, \mathbb{W}_C)$. Given two projections π_c and π_s that map $\mathcal{P}$ to behaviors of type equal to that of $\mathcal{C}$ and $\mathcal{S}$ respectively, find a controller $\mathcal{C}$ of this type such that

$$\pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \mathcal{C}) = \mathcal{S}. \tag{4.2}$$

Notice that if these two projections are the identity map, then the problem is essentially reduced into a full interconnection problem discussed in the previous subsection. A controller $\mathcal{C}$ is said to achieve the specification $\mathcal{S}$ if it solves the problem.

Examples of control problem in this type are:

- Control of linear systems, where only a part of the variables are available as control variables. This situation is depicted in Figure 4.1.
- Supervisory control of discrete event systems, where not all events are observable and controllable [RW89, CL99].
- Control of hybrid systems, where not all events or continuous variables are available for interconnection.

Control problems of this type have been discussed in somewhat different representation in, for example [SJ02, Sch03a]. There, the conditions for solving the problem as well as a candidate solution, called the *canonical controller*, were presented.

Definition 4.3. Given a control problem as in (4.3). The **first canonical controller** is defined as follows [WBJT03].

$$\mathcal{C}'_{\mathrm{can}} := \{c \in \pi_c \mathcal{P} \mid \pi_s \pi_c^{-1} c \subset \mathcal{S}\}. \tag{4.3}$$

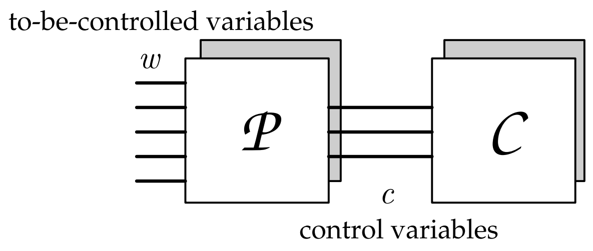


Figure 4.1: A partial interconnection in a control problem.

The first canonical controller possesses an important property, stated in the following proposition.

Proposition 4.4. The controller $\mathcal{C}'_{\text{can}}$ achieves the maximal achievable behavior contained in $\mathcal{S}$.

Proof. From (4.3) we can readily conclude that

$$\pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \mathcal{C}'_{\text{can}}) \subset \mathcal{S}. \tag{4.4}$$

To show that $\mathcal{C}'_{\text{can}}$ achieves the maximal achievable behavior in $\mathcal{S}$, consider any other controller $\mathcal{C}'$ such that

$$\pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \mathcal{C}') \subset \mathcal{S}. \tag{4.5}$$

We shall show that $(\pi_c \mathcal{P} \parallel \mathcal{C}') \subset (\pi_c \mathcal{P} \parallel \mathcal{C}'_{\text{can}})$. Take any $c \in (\pi_c \mathcal{P} \parallel \mathcal{C}')$. Because of (4.5), we know that $\pi_s \pi_c^{-1} c \subset \mathcal{S}$. Therefore, by the construction in (4.3), c is also contained in $(\pi_c \mathcal{P} \parallel \mathcal{C}'_{\text{can}})$. It follows that the behavior achieved by $\mathcal{C}'$ is contained in that of $\mathcal{C}'_{\text{can}}$. □

Theorem 4.5. The specification $\mathcal{S}$ in (4.2) is achievable, if and only if $\mathcal{C}'_{\text{can}}$ achieves $\mathcal{S}$.

Proof. We only have to prove the only if part, since the converse is obvious. From Proposition 4.4 we know that $\mathcal{C}'_{\text{can}}$ achieves the maximal achievable behavior contained in $\mathcal{S}$. This means that if $\mathcal{S}$ itself is achievable then $\mathcal{C}'_{\text{can}}$ achieves $\mathcal{S}$. □

Although the first canonical controller is powerful, its construction is defined as a set-theoretical construct. To obtain a representation of this controller we need to perform some computation. Later we shall introduce a weaker version of canonical controller, which can be constructed as an interconnection.

We define the *homogeneity* property as follows.

Definition 4.6. The plant $\mathcal{P}$ is said to satisfy the **homogeneity property** [SJ02] with respect to the projections π_s and π_c, if for any (s_1, c_1) and (s_2, c_1) in $\pi_s \mathcal{P} \times \pi_c \mathcal{P}$,

$$(s_1, c_2) \in \pi_s \mathcal{P} \times \pi_c \mathcal{P} \Rightarrow (s_2, c_2) \in \pi_s \mathcal{P} \times \pi_c \mathcal{P},\ \forall c_2 \in \pi_c \mathcal{P}. \tag{4.6}$$

Remark 4.7. In terms of observability, the homogeneity property can be also expressed as follows. The homogeneity property is satisfied if and only if $\pi_s \pi_c^{-1} \pi_c = \pi_c \pi_s^{-1} \pi_s$. Both projections are defined to be acting on $\mathcal{P}$.

Remark 4.8. Homogeneity can also be understood as follows. The behavior $\mathcal{P}$ can be seen as a (graph of a) relation between $\mathbb{W}_S^{\mathbb{T}_S}$ and $\mathbb{W}_C^{\mathbb{T}_C}$ (see Figure 4.3). A relation is called *independent* if its graph can be written as a Cartesian product of its projections on the related domains. The behavior $\mathcal{P}$ has the homogeneity property if it can be written as a union of independent relations.

The conditions for solvability of a control problem involving a plant with homogeneity property is given in the following theorem, which was proved in [SJ02].

Theorem 4.9. For a plant $\mathcal{P}$ satisfying the homogeneity property with respect to π_s and π_c, the specification $\mathcal{S}$ as in (4.2) is achievable if and only if
(i) $\mathcal{S} \subset \pi_s \mathcal{P}$, and
(ii) $\mathcal{S} = \pi_s \pi_c^{-1} \pi_c \pi_s^{-1} \mathcal{S}$.

Statement (ii) in the theorem above is equivalent to the following statement. For any $w, w' \in \mathcal{P}$, if $\pi_c w = \pi_c w'$, then the following biimplication holds.

$$\pi_s w \in \mathcal{S} \Leftrightarrow \pi_s w' \in \mathcal{S}. \tag{4.7}$$

If the plant $\mathcal{P}$ does not have the homogeneity property, then conditions (i) and (ii) in Theorem 4.9 are just sufficient for achievability of the specification $\mathcal{S}$, and not necessary.

We define the *second canonical controller* as follows[1].

Definition 4.10. Given a control problem as in (4.3). The **second canonical controller** is defined as follows [SJ02, Sch03a].

$$\mathcal{C}''_{\text{can}} := \pi_c \pi_s^{-1} (\pi_s \mathcal{P} \parallel \mathcal{S}). \tag{4.8}$$

Notice that in (4.8) the controller $\mathcal{C}''_{\text{can}}$ is defined in terms of an interconnection. To make the exposition clearer, the interconnection diagram is depicted in Figure 4.2. Notice that in the block diagram, the canonical controller $\mathcal{C}''_{\text{can}}$ has a copy of the plant. In set theoretic notation, the second canonical controller $\mathcal{C}''_{\text{can}}$ is characterized as follows.

$$\mathcal{C}''_{\text{can}} = \{c \in \mathbb{W}_C^{\mathbb{T}_C} \mid \exists w \in \mathcal{P} \text{ such that } \pi_c w = c \text{ and } \pi_s w \in \mathcal{S}\}. \tag{4.9}$$

A comparison between the first and second canonical controller and the behaviors that they achieve is shown in Figure 4.3.

The second canonical controller has the following special property.

[1] In [SJ02, Sch03a] the second canonical controller is called canonical controller. The first canonical controller is introduced later in [WBJT03].

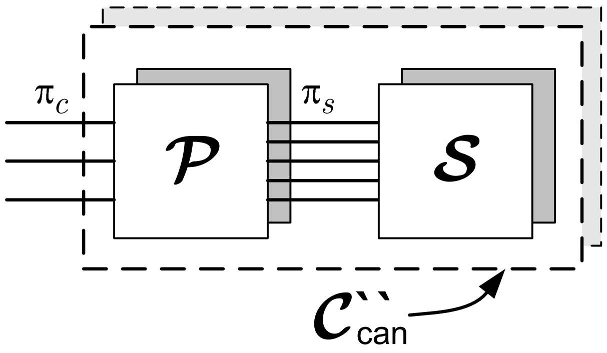


Figure 4.2: The canonical controller $\mathcal{C}''_{\text{can}}$ as an interconnection. $\mathcal{C}''_{\text{can}} := \pi_c\pi_s^{-1}(\pi_s\mathcal{P} \parallel \mathcal{S})$.

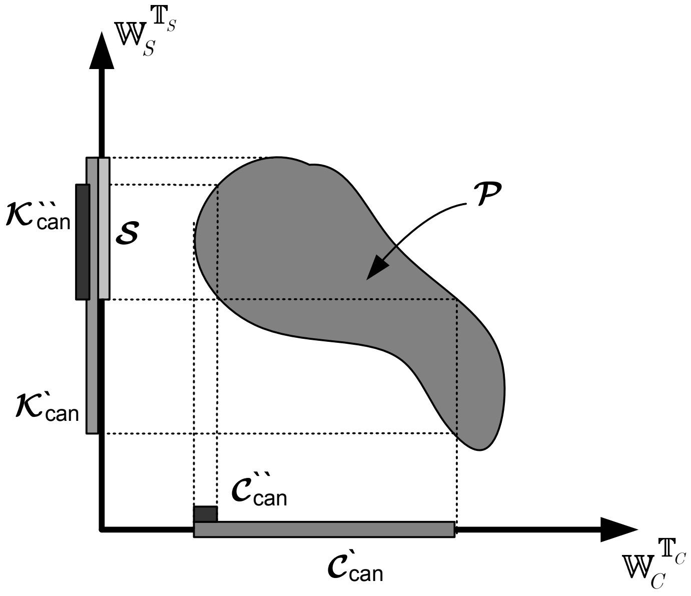

$\mathbb{W}_S^{\mathbb{T}_S}$
$\mathcal{P}$
$\mathcal{K}''_{\text{can}}$
$\mathcal{S}$
$\mathcal{K}'_{\text{can}}$
$\mathcal{C}''_{\text{can}}$
$\mathcal{C}'_{\text{can}}$
$\mathbb{W}_C^{\mathbb{T}_C}$

Figure 4.3: A comparison between the two canonical controllers. Here $\mathcal{C}'_{\text{can}}$ and $\mathcal{C}''_{\text{can}}$ denote the first and second canonical controller respectively. The behaviors that they achieve are denoted by $\mathcal{K}'_{\text{can}}$ and $\mathcal{K}''_{\text{can}}$ respectively.

Proposition 4.11. For a plant $\mathcal{P}$ satisfying the homogeneity property with respect to π_s and π_c, the second canonical controller $\mathcal{C}''_{\text{can}}$ achieves the minimal achievable behavior containing $(\mathcal{S} \parallel \pi_s\mathcal{P})$.

Proof. Denote $\mathcal{K}''_{\text{can}} := \pi_s\pi_c^{-1}(\pi_c\mathcal{P} \parallel \mathcal{C}''_{\text{can}})$. Since $\mathcal{C}''_{\text{can}} \subset \pi_c\mathcal{P}$,

$$\begin{aligned}\mathcal{K}''_{\text{can}} &= \pi_s\pi_c^{-1}(\mathcal{C}''_{\text{can}}),\\ &= \pi_s\pi_c^{-1}\pi_c\pi_s^{-1}(\pi_s\mathcal{P} \parallel \mathcal{S}),\\ &\supset (\pi_s\mathcal{P} \parallel \mathcal{S}).\end{aligned} \tag{4.10}$$

Take any other controller $\mathcal{C}'$ such that $(\pi_s\mathcal{P} \parallel \mathcal{S}) \subset \pi_s\pi_c^{-1}(\pi_c\mathcal{P} \parallel \mathcal{C}')$. Denote $\mathcal{K}' := \pi_s\pi_c^{-1}(\pi_c\mathcal{P} \parallel \mathcal{C}')$. Take any $s \in \mathcal{K}''_{\text{can}}$. We shall prove that $s \in \mathcal{K}'$. If $s \in (\pi_s\mathcal{P} \parallel \mathcal{S})$ then $s \in \mathcal{K}'$ since $(\pi_s\mathcal{P} \parallel \mathcal{S}) \subset \mathcal{K}'$. If $s \notin (\pi_s\mathcal{P} \parallel \mathcal{S})$, then there exist w and w' in $\mathcal{P}$ such that $\pi_s w = s$, $\pi_c w =: c \in \mathcal{C}''_{\text{can}}$, $\pi_c w' = c$, and $\pi_s w' =: s' \in (\pi_s\mathcal{P} \parallel \mathcal{S})$. We are going to show that $s \notin \mathcal{K}'$ is a contradiction. Suppose that it is true, then $\pi_c\pi_s^{-1}s \parallel \mathcal{C}' = \emptyset$. But because of the homogeneity property, $\pi_c\pi_s^{-1}s = \pi_c\pi_s^{-1}s'$. Hence $\pi_c\pi_s^{-1}s' \parallel \mathcal{C}' = \emptyset$ and $s' \notin \mathcal{K}'$, which is a contradiction since $s' \in (\pi_s\mathcal{P} \parallel \mathcal{S})$. □

The following corollary, follows from Theorem 4.9 and Proposition 4.11.

Corollary 4.12. For a plant $\mathcal{P}$ satisfying the homogeneity property with respect to π_s and π_c, $\mathcal{S}$ is achievable if and only if $\mathcal{C}''_{\text{can}}$ achieves it.

Remark 4.13. Obviously, the stronger part of the corollary above is the 'only if' part. If the plant $\mathcal{P}$ does not possess the homogeneity property, it can be that a certain specification $\mathcal{S}$ can be achieved by the second canonical controller $\mathcal{C}''_{\text{can}}$. However, generally we cannot guarantee that a specification is achievable only if the $\mathcal{C}''_{\text{can}}$ achieves it.

Stronger than this corollary, we can state the following proposition.

Proposition 4.14. For a plant $\mathcal{P}$ satisfying the homogeneity property with respect to π_s and π_c, if $\mathcal{S}$ is achievable then $\mathcal{C}'_{\text{can}} = \mathcal{C}''_{\text{can}}$.

Proof. ($\mathcal{C}'_{\text{can}} \subseteq \mathcal{C}''_{\text{can}}$) Take any $c \in \mathcal{C}'_{\text{can}}$. By Definition 4.3, $\pi_s\pi_c^{-1}c \subset \mathcal{S}$. Notice that since $\mathcal{S}$ is achievable, (4.8) can be written as

$$\mathcal{C}''_{\text{can}} = \pi_c\pi_s^{-1}\mathcal{S}. \tag{4.11}$$

This implies $c \in \mathcal{C}''_{\text{can}}$.

($\mathcal{C}'_{\text{can}} \supseteq \mathcal{C}''_{\text{can}}$) Take any $c \in \mathcal{C}''_{\text{can}}$. Since $\mathcal{S}$ is achievable, we know that $\mathcal{C}''_{\text{can}}$ is constructed based on (4.11). Moreover, we also have the following relation

$$\mathcal{S} = \pi_s\pi_c^{-1}\mathcal{C}''_{\text{can}}. \tag{4.12}$$

This implies $\pi_s\pi_c^{-1}c \subset \mathcal{S}$, which means $c \in \mathcal{C}'_{\text{can}}$. □

Notice that for control problems, in which the homogeneity property holds, both canonical controllers are equivalent. However, the second controller is expressed in terms of interconnection of systems, as opposed to the first canonical controller that is expressed as a set theoretical construct.

Requiring that the plant has the homogeneity property might seem restrictive, but actually this requirement is satisfied by quite a large class of problems. The following are examples of cases where the homogeneity property is satisfied.

1. All cases where $\mathcal{P}$ is a linear system and the projections are linear differential (or difference) maps. To verify that this is the case, construct the following set
$$\{(s,c) \mid \exists w \in \mathcal{P} \text{ such that } \pi_s w = s \text{ and } \pi_c w = c\}.$$
The homogeneity property follows from the fact that this set is a linear space.

2. All supervisory control cases, where the set of controllable events coincides with the observable ones[2] and the specification is given in terms of the whole alphabet. To verify that this is the case, notice that π_s is the identity map.

For these cases we can readily apply Theorem 4.9. Consider the following corollary for supervisory control of discrete event systems, where the set of controllable events coincides with the observable ones.

Corollary 4.15. Let L be the generated language of a finite state automaton with alphabet E. Let $Z \subset E$ be the set of observable and controllable events. Define π_z to be the projection acting on L that eliminates the occurrence of events in $E\backslash Z$. There exists a supervisor language C such that the prefix-closed specification $\mathcal{S}$ is achieved, i.e. $\pi_z^{-1}(\pi_z L \parallel C) = \mathcal{S}$, if and only if
(i) $\mathcal{S} \subset L$, and
(ii) $\mathcal{S} = \pi_z^{-1}\pi_z\mathcal{S}$.

This corollary is an application of Theorem 4.9, where L is the plant behavior and π_s is the identity map. Hereby we recover a known result in the area of supervisory control of discrete event systems. Conditions (i) and (ii) are equivalent to a property which, in other literature such as [CL99], is called *normality* of the language $\mathcal{S}$ with respect to the projection associated to the set of events Z.

Remark 4.16. Ideas similar to the canonical controllers have also appeared in other literatures, for example, [Sme87] and [Roc02].

The following result tells about achievability of the specification if the problem is altered by changing the projection π_c.

Proposition 4.17. Given a plant $\mathcal{P}$ satisfying the homogeneity property with respect to π_s and π_c, and a specification $\mathcal{S}$ that is achievable. If π_c is replaced with any $\phi_c \succeq \pi_c$ such that the homogeneity property is still satisfied, then $\mathcal{S}$ is still achievable.

[2]The terms observable and controllable refer to the usage in [CL99].

Proof. It is sufficient to show that condition (ii) in Theorem 4.9 remains satisfied, even if π_c is replaced with $\phi_c \succeq \pi_c$. Notice that in general $\mathcal{S} \subset \pi_s \pi_c^{-1} \pi_c \, \pi_s^{-1} \mathcal{S}$. So it remains to prove that

$$\left(\mathcal{S} \supset \pi_s \pi_c^{-1} \pi_c \pi_s^{-1} \mathcal{S}\right) \Rightarrow \left(\mathcal{S} \supset \pi_s \phi_c^{-1} \phi_c \pi_s^{-1} \mathcal{S}\right) . \tag{4.13}$$

Since $\phi_c \succeq \pi_c$ implies $\phi_c^{-1}\phi_c \subset \pi_c^{-1}\pi_c$, (4.13) is satisfied and hence $\mathcal{S}$ is still achievable. □

This result is actually very intuitive and it can be explained as follows. It has been mentioned before that the partial ordering $\succeq$ is related to the amount information retained by the projection. If a projection $\phi_c \succeq \pi_c$, it means the projection ϕ_c retains more information than the projection π_c. The role of π_c in the control problem is to determine the extent, to which the controller can interact with the plan. Replacing π_c with ϕ_c means allowing higher amount of information to be used in the interconnection between the plant and the controller, and hence making it easier to achieve the desired specification. We can think of it as using a larger interaction channel between the plant and the controller. This interpretation is evident, for example, in the following cases.

1. For control problems of linear systems, the projection π_c typically takes the form of projecting the behavior to a set of control variables $\mathbf{c}$ [Wil97, Bel03]. Replacing π_c with ϕ_c, where $\phi_c \succeq \pi_c$, can mean using a larger set of control variables. Obviously if the control problem is solvable with the smaller set of control variables, it is also solvable using the bigger one.

2. For supervisory control problems of discrete event systems, where the set of controllable events coincides with the observable ones, the projection π_c typically denotes the projection of the language of the plant to the set of controllable-observable events. Replacing π_c with ϕ_c, where $\phi_c \succeq \pi_c$, can mean allowing more events to be controllable and observable. Obviously if the control problem is solvable with the smaller set of controllable-observable events, it is also solvable using the bigger one [RW89, CL99].

3. For control of hybrid behavioral automata, replacing π_c with ϕ_c, where $\phi_c \succeq \pi_c$, can mean allowing more events to be controllable and observable and/or more continuous variables to be used as control variables. Again, we can see that this actually makes the control problem easier to solve.

We have seen that the bigger π_c is, the easier it is to solve the control problem. Based on this fact, we can identify an interesting research problem. Instead of replacing π_c with something bigger, we replace it with something smaller. Obviously this will affect the solvability of the control problem in a negative way. The problem is then to find out to extent to which π_c can be reduced while retaining the solvability of the control problem. This problem is called *control with minimal interaction*, a version of which we shall discuss later in this chapter.

4.1.3 Control problems with a tolerance gap

In both variants of control problems that we have seen above, the goal of the control problem is to achieve a given specification $\mathcal{S}$. In this subsection, we shall consider the variant where the requirement is relaxed, by requiring that the controlled behavior lies between two specification bounds, $\mathcal{S}_r$ and $\mathcal{S}_a$. The idea is that $\mathcal{S}_r$ is the minimal required behavior and that $\mathcal{S}_a$ is the maximal allowed behavior [CL99]. To avoid ill-posed problems, it is always assumed that $\mathcal{S}_r \subset \mathcal{S}_a$.

When such a tolerance gap is present, the full interconnection control problem discussed in Subsection 4.1.1 becomes

Problem 4.18. Given the plant $\mathcal{P}$ and two specification limits $\mathcal{S}_r$ and $\mathcal{S}_a$, find a controller $\mathcal{C}$ such that

$$\mathcal{S}_r \subset (\mathcal{P} \parallel \mathcal{C}) \subset \mathcal{S}_a. \tag{4.14}$$

The following theorem states the conditions for solvability (i.e. existence of a solution for $\mathcal{C}$) of such problem. A proof is not included since it is trivial.

Theorem 4.19. The control problem associated with (4.14) is solvable if and only if $\mathcal{S}_r \subset \mathcal{P}$.

The problem becomes more interesting when some projections are involved.

Problem 4.20. Given a plant $\mathcal{P}$ of type $(\mathbb{T}_P, \mathbb{W}_P)$, and two specification limits $\mathcal{S}_r$ and $\mathcal{S}_a$ of type $(\mathbb{T}_S, \mathbb{W}_S)$. The candidate controllers are of type $(\mathbb{T}_C, \mathbb{W}_C)$. Find a controller $\mathcal{C}$ of this type such that

$$\mathcal{S}_r \subset \pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \mathcal{C}) \subset \mathcal{S}_a. \tag{4.15}$$

The projections π_c and π_s are also given. They map $\mathcal{P}$ to behaviors of type equal to that of $\mathcal{C}$ and $\mathcal{S}$ respectively.

As is the case with the discussion in the previous subsection, we utilize the idea of canonical controller to solve this problem. The analog of the first canonical controller defined in Definition 4.3, for solving Problem 4.20 is defined as follows.

$$\mathcal{C}'_{\text{can}} = \{c \in \pi_c \mathcal{P} \mid \pi_s \pi_c^{-1} c \subset \mathcal{S}_a\}. \tag{4.16}$$

This canonical controller also possesses the property analogous to the one described in Theorem 4.5.

Theorem 4.21. Problem 4.20 is solvable, if and only if $\mathcal{C}'_{\text{can}}$, which is constructed according to (4.16), solves it.

Proof. We only need to prove the "only if"part. We use Proposition 4.4 to establish that $\mathcal{C}'_{\text{can}}$ achieves the maximal achievable behavior in $\mathcal{S}_a$. Suppose that $\mathcal{C}'_{\text{can}}$ does not solve the problem, then

$$\mathcal{S}_r \not\subset \pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \mathcal{C}'_{\text{can}}). \tag{4.17}$$

Since $\mathcal{C}'_{\text{can}}$ achieves the maximal achievable behavior in $\mathcal{S}_a$, (4.17) implies that the problem is not solvable. □

If the plant $\mathcal{P}$ possesses the homogeneity property with respect to π_c and π_s (see previous subsection), we again construct the *second canonical controller*, in a way analog to the construction in the previous subsection, as follows.

$$\mathcal{C}''_{\mathrm{can}} = \pi_c \pi_s^{-1}(\pi_s \mathcal{P} \parallel \mathcal{S}_r). \tag{4.18}$$

Theorem 4.22. For a plant $\mathcal{P}$ satisfying the homogeneity property with respect to π_s and π_c, all the following statements are equivalent.
(i) Problem 4.20 is solvable.
(ii) $\mathcal{S}_r \subset \pi_s \mathcal{P}$, and $\mathcal{S}_a \supset \pi_s \pi_c^{-1} \pi_c \pi_s^{-1} S_r$.
(iii) The second canonical controller $\mathcal{C}''_{\mathrm{can}}$ solves the control problem.

Proof. (i)⇒(ii) Since all achievable behaviors must be contained in $\pi_s \mathcal{P}$, necessarily $\mathcal{S}_r \subset \pi_s \mathcal{P}$. The second canonical controller is then (see (4.18)) $\mathcal{C}''_{\mathrm{can}} = \pi_c \pi_s^{-1} \mathcal{S}_r$. Suppose that $\mathcal{S}_a \not\supset \pi_s \pi_c^{-1} \pi_c \pi_s^{-1} S_r$. We are going to show that this is a contradiction. The behavior achieved by the second canonical controller is $\mathcal{K}''_{\mathrm{can}} := \pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \pi_c \pi_s^{-1} \mathcal{S}_r)$. Since $\mathcal{S}_r \subset \pi_s \mathcal{P}$, we also have that $\pi_c \pi_s^{-1} \mathcal{S}_r \subset \pi_c \mathcal{P}$. Hence

$$\mathcal{K}''_{\mathrm{can}} = \pi_s \pi_c^{-1} \pi_c \pi_s^{-1} S_r \not\subset \mathcal{S}_a. \tag{4.19}$$

From Proposition 4.11, and since $\mathcal{S}_r \subset \pi_s \mathcal{P}$, we know that $\mathcal{K}''_{\mathrm{can}}$ is the minimal achievable behavior containing $\mathcal{S}_r$. Therefore (4.19) implies that the control problem is not solvable.

(ii)⇒(iii) From the previous paragraph, it follows that if (ii) is satisfied then $\mathcal{S}_r \subset \mathcal{K}''_{\mathrm{can}} \subset \mathcal{S}_a$.

(iii)⇒(i) Trivial. □

An example of control problems of this type can be obtained, for example by replacing the control problem in Corollary 4.15 with the version with a tolerance gap. In this case, $\mathcal{S}_r$ is present to make sure that the supervised language possess some "liveliness"property, and $\mathcal{S}_a$ is to prevent some undesirable executions to take place (for example, for safety reasons).

4.2 Compatibility constraint

4.2.1 Constraint formulation

In the previous section we have discussed control problems in the behavioral setting without any constraints. That is, we consider the problem of finding *any* controller that when interconnected with the plant in a certain way (which is defined by the projection π_c) yields some desirable behavior. In the remaining part of this chapter we shall discuss constrained control problems. By constrained we mean that the controller not only solves the control problem as defined in the previous section, but also satisfies some additional requirements.

The constraint that we are going to discuss in this section is *compatibility* between the plant and the controller. The nature of the compatibility issues can be explained as follows.

Take any two dynamical systems Σ_1 and Σ_2, whose behaviors are $\mathfrak{B}_1$ and $\mathfrak{B}_2$ respectively. Recall that by definition, the behavior resulting from the interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is $\mathfrak{B}_1 \cap \mathfrak{B}_2$. This means the trajectories that are accepted in the interconnection obey the laws of both systems at all time. For physical systems, this can be regarded as an approximation, for the following reason. If both systems already exist before the interconnection, the interconnection must be realized at a particular time instant, say $t \in \mathbb{T}$. The consequences are:

1. The trajectories of the interconnected systems only have to obey the laws of Σ_1 and Σ_2 simultaneously after time t.
2. It is possible that a certain trajectory w of Σ_1 (or Σ_2) cannot continue its evolution after time t.

Consider the following example.

Example 4.23. An oscillating point mass is modelled with a behavior $\mathfrak{B}_1 \in \mathfrak{L}_c^1$ given by

$$\mathfrak{B}_1 = \left\{ w \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}) \mid \frac{d^2 w}{dt^2} + w = 0 \right\}. \tag{4.20}$$

The point mass is isolated, which means that there is no external force influencing the point mass. Now, consider another behavior

$$\mathfrak{B}_2 = \{ w \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}) \mid w = 0 \}. \tag{4.21}$$

Suppose that the interconnection is formed at time $t = 0$. It can be verified that the only trajectory that satisfies the laws of both systems is the zero trajectory. This results in a conflict in $\mathfrak{B}_1$ because only the zero trajectory is allowed after time $t = 0$, while prior to that time, the isolated point mass can be oscillating.

Let $\mathcal{G} = (G, +, <)$ be the underlying totally ordered commutative group of the time axis $\mathbb{T}$. Recall Lemma 2.8 that says that for any $\tau_1, \tau_2 \in G$, the set $\{t \in G \mid t > \tau_1\}$ and $\{t \in G \mid t > \tau_2\}$ are isomorphic. We introduce the following time axis

$$\mathbb{T}^+ := \{ t \in G \mid t > 0 \} \tag{4.22}$$

as a representation for all such left-open-right-unbounded time intervals.

The fact that the interconnection is formed at a certain time can be modelled as a partial interconnection.

Definition 4.24. Given a behavior $\mathfrak{B}$ of type $(\mathbb{T}, \mathbb{W})$. The projection that gives the future behavior of $\mathfrak{B}$ after time $\tau \in \mathbb{T}$, is denoted as π_τ, and defined as

$$\pi_\tau : \mathfrak{B} \to \mathbb{W}^{\mathbb{T}^+}, \tag{4.23}$$

$$[\pi_\tau(w)](t) := w(t - \tau), \ \forall w \in \mathfrak{B}, t \in \mathbb{T}^+. \tag{4.24}$$

Thus the projection maps $\mathfrak{B}$ to a behavior of type $(\mathbb{T}^+, \mathbb{W})$.

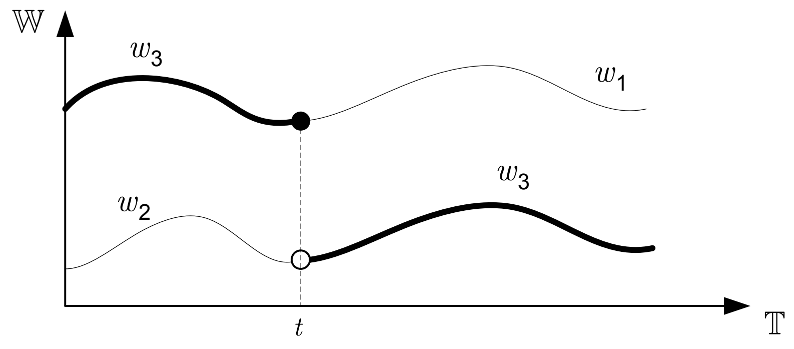


Figure 4.4: An illustration of directability. The trajectory w_1 is directable to w_2 at time t if w_3 (thick curve) is also an element of the behavior.

The future interconnection of $\mathfrak{B}_1$ and $\mathfrak{B}_2$, both of type $(\mathbb{T}, \mathbb{W})$, can be written as

$$\mathfrak{B} = \pi_{\tau_1}\mathfrak{B}_1 \parallel \pi_{\tau_2}\mathfrak{B}_2. \tag{4.25}$$

Here we accommodate the possibility of different time instants of interconnection of the two systems, namely τ_1 and τ_2. The reason is the fact that possibly there is no synchronicity in the timing of both systems. Thus, before the interconnection the time in both systems may run independently.

Before we can define compatibility, we need the following definition.

Definition 4.25. Given a dynamical system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathfrak{B})$. Let $w_1, w_2 \in \mathfrak{B}$ and $t \in \mathbb{T}$. We say that w_1 is **directable** to w_2 at time t if $w_3 := (w_1 \wedge_t^t w_2)$ is an element of $\mathfrak{B}$. Notice that if Σ admits a canonical minimal state map ϕ, then w_1 is *directable* to w_2 if and only if $\phi(w_1, t) = \phi(w_2, t)$.

The fact that w_1 is *directable* to w_2 at time t is written as $w_1 D_{\mathfrak{B}}(t) w_2$. Thus, the directability relation is represented by the symbol $D_{\mathfrak{B}}(t)$. See Figure 4.4 for an illustration of directability.

In the following we shall discuss how past and future induced state maps are related with directability.

Proposition 4.26. Let a behavior $\mathfrak{B}$ be of type $(\mathbb{T}, \mathbb{W})$. Let α and ω be any past induced and future induced state maps respectively. The symbols $\bar{\alpha}_t$ and $\bar{\omega}_t$ are defined as follows.

$$\bar{\alpha}_t(w) := \{w' \in \mathfrak{B} \mid \alpha(w', t) = \alpha(w, t)\}, \tag{4.26}$$

$$\bar{\omega}_t(w) := \{w' \in \mathfrak{B} \mid \omega(w', t) = \omega(w, t)\}, \tag{4.27}$$

for all $w \in \mathfrak{B}$. The following relation holds for all $w_1, w_2 \in \mathfrak{B}$ and $t \in \mathbb{T}$.

$$(w_1 D_{\mathfrak{B}}(t) w_2) \Leftrightarrow \bar{\alpha}_t(w_1) \cap \bar{\omega}_t(w_2) \neq \emptyset. \tag{4.28}$$

Proof. ($\Rightarrow$)Let w_{i-} and w_{i+} denote the past and future of w_i, $i = 1, 2$. That is, $w_{i-} \in \mathfrak{B}|_{(-\infty,t]}$ and $w_{i+} \in \mathfrak{B}|_{(t,\infty)}$, if $\mathbb{T} = \mathbb{R}$. Denote $w_3 := w_1 \wedge_t^t w_2$. The past of w_3 is w_{1-} and its future is w_{2+}. Consequently we have that

$$w_3 \in \bar{\alpha}_t(w_1), \tag{4.29a}$$
$$w_3 \in \bar{\omega}_t(w_2). \tag{4.29b}$$

Hence $\bar{\alpha}_t(w_1) \cap \bar{\omega}_t(w_2) \neq \emptyset$.

($\Leftarrow$) Assume that $\bar{\alpha}_t(w_1) \cap \bar{\omega}_t(w_2) \neq \emptyset$. Take an element from this set and call it w_3. We know from the state property that

$$w_4 := (w_1 \wedge_t^t w_3) \in \mathfrak{B}.$$

Since the future of w_4 is the same as that of w_3, necessarily $\omega(w_4, t) = \omega(w_3, t) = \omega(w_2, t)$. Hence we can construct w_5 such that

$$w_5 := (w_4 \wedge_t^t w_2) \in \mathfrak{B}.$$

Notice that w_5 has the past of w_1 and the future of w_2. Hence

$$w_5 = (w_1 \wedge_t^t w_2) \in \mathfrak{B}.$$

Notice that we do not require $w_1, \cdots, w_5$ to be distinct. □

Corollary 4.27. Let a behavior $\mathfrak{B}$ be of type $(\mathbb{T}, \mathbb{W})$. Let α and ω be any past induced and future induced state maps respectively. The following relation holds

$$(w_1 D_{\mathfrak{B}}(t) w_2) \Leftrightarrow w_2 \in (\bar{\omega}_t \circ \bar{\alpha}_t)(w_1), \tag{4.30}$$
$$\Leftrightarrow w_1 \in (\bar{\alpha}_t \circ \bar{\omega}_t)(w_2). \tag{4.31}$$

Motivated by this corollary, we introduce the set valued maps $D_{\mathfrak{B},t}(\cdot)$ and $D^{-1}_{\mathfrak{B},t}(\cdot)$ as

$$D_{\mathfrak{B},t}(\cdot) := (\bar{\omega}_t \circ \bar{\alpha}_t)(\cdot), \tag{4.32a}$$
$$D^{-1}_{\mathfrak{B},t}(\cdot) := (\bar{\alpha}_t \circ \bar{\omega}_t)(\cdot). \tag{4.32b}$$

We can define compatibility as follows.

Definition 4.28. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be behaviors of type $(\mathbb{T}, \mathbb{W})$. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is **compatible** if there exist $\tau_1, \tau_2 \in \mathbb{T}$ such that for any $w_i \in \mathfrak{B}_i$, $i = 1, 2$, there exists a $\tilde{w}_i \in \pi^{-1}_{\tau_i}(\pi_{\tau_1}\mathfrak{B}_1 \parallel \pi_{\tau_2}\mathfrak{B}_2)$, $i = 1, 2$, such that
(i) $w_i D_{\mathfrak{B}_i}(\tau_i)\tilde{w}_i$, $i = 1, 2$, and
(ii) $\pi_{\tau_1}\tilde{w}_1 = \pi_{\tau_2}\tilde{w}_2$ (see Definition 4.24)

The meaning of this definition is that in a compatible interconnection, any pair of trajectories from the individual systems is directable to a trajectory accepted by the interconnection at the time the interconnection is formed.

Lemma 4.29. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be behaviors of type $(\mathbb{T}, \mathbb{W})$. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is compatible if and only if there exist $t_1, t_2 \in \mathbb{T}$ such that

$$D_{\mathfrak{B}_2,t_2}\left(\pi_{t_2}^{-1}\left((\pi_{t_1} D_{\mathfrak{B}_1,t_1}(w_1)) \cap \mathfrak{B}\right)\right) = \mathfrak{B}_2, \forall w_1 \in \mathfrak{B}_1, \tag{4.33}$$

$$D_{\mathfrak{B}_1,t_1}\left(\pi_{t_1}^{-1}\left((\pi_{t_2} D_{\mathfrak{B}_2,t_2}(w_2)) \cap \mathfrak{B}\right)\right) = \mathfrak{B}_1, \forall w_2 \in \mathfrak{B}_2, \tag{4.34}$$

where $\mathfrak{B} := \pi_{t_1}\mathfrak{B}_1 \parallel \pi_{t_2}\mathfrak{B}_2$.

The statement in the lemma above might look cumbersome, but is merely a rewriting of Definition 4.28.

We can strengthen the definition of compatibility by requiring that the interconnection can be made at any time. Formally, we define it as follows.

Definition 4.30. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be behaviors of type $(\mathbb{T}, \mathbb{W})$. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is **uniformly compatible** if for any $\tau_i \in \mathbb{T}$ and $w_i \in \mathfrak{B}_i$, $i = 1, 2$, there exists a $\tilde{w}_i \in \pi_{\tau_i}^{-1}(\pi_{\tau_1}\mathfrak{B}_1 \parallel \pi_{\tau_2}\mathfrak{B}_2)$, $i = 1, 2$, such that
(i) $w_i D_{\mathfrak{B}_i}(\tau_i)\tilde{w}_i$, $i = 1, 2$, and
(ii) $\pi_{\tau_1}\tilde{w}_1 = \pi_{\tau_2}\tilde{w}_2$.

We can formulate the following result, as an analog of Lemma 4.29.

Lemma 4.31. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be behaviors of type $(\mathbb{T}, \mathbb{W})$. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is uniformly compatible if and only if for any $t_1, t_2 \in \mathbb{T}$, the following holds.

$$D_{\mathfrak{B}_2,t_2}\left(\pi_{t_2}^{-1}\left((\pi_{t_1} D_{\mathfrak{B}_1,t_1}(w_1)) \cap \mathfrak{B}\right)\right) = \mathfrak{B}_2, \forall w_1 \in \mathfrak{B}_1, \tag{4.35}$$

$$D_{\mathfrak{B}_1,t_1}\left(\pi_{t_1}^{-1}\left((\pi_{t_2} D_{\mathfrak{B}_2,t_2}(w_2)) \cap \mathfrak{B}\right)\right) = \mathfrak{B}_1, \forall w_2 \in \mathfrak{B}_2, \tag{4.36}$$

where $\mathfrak{B} := \pi_{t_1}\mathfrak{B}_1 \parallel \pi_{t_2}\mathfrak{B}_2$.

Suppose that $\mathfrak{B}_1$ and $\mathfrak{B}_2$ each admit a canonical minimal state map ϕ_1 and ϕ_2. The condition in Lemma 4.31 is then equivalent to the fact that of the state maps are independent. Before we can precisely define this statement, we need the following definition.

Definition 4.32. Given a behavior $\mathfrak{B}$ of type $(\mathbb{T}, \mathbb{W})$. The **suffix behavior** of $\mathfrak{B}$, denoted as $\mathfrak{B}_\omega$ is defined as

$$\mathfrak{B}_\omega := \bigcup_{t \in \mathbb{T}} \pi_t \mathfrak{B}. \tag{4.37}$$

The suffix behavior thus contains all possible suffices of trajectories in $\mathfrak{B}$. The suffices might be obtained by chopping the trajectories at different time instants. However, since they are all trajectories of the same type, we can group them together.

The canonical minimal state map of a behavior is future-induced. This has been established in Corollary 3.42. As a consequence, the canonical minimal state map ϕ of a behavior $\mathfrak{B}$ (if exists) can be considered as a map acting on $\mathfrak{B}_\omega$. This is done as follows. Take any $w \in \mathfrak{B}_\omega$. We know from definition that there exist $\tilde{w} \in \mathfrak{B}$ and $t \in \mathbb{T}$ such that $w = \pi_t \tilde{w}$. We then define

$$\phi(w) := \phi(\tilde{w}, t). \tag{4.38}$$

Now, the statement about the independence of the canonical minimal state maps in a uniformly compatible interconnection can be formally expressed as follows.

Theorem 4.33. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be behaviors of type $(\mathbb{T}, \mathbb{W})$. Suppose that $\mathfrak{B}_1$ and $\mathfrak{B}_2$ each admit a canonical minimal state map ϕ_1 and ϕ_2. Denote the state space of the respective state maps as Φ_1 and Φ_2. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is uniformly compatible if and only if the following holds. For any $x_1 \in \Phi_1$ and $x_2 \in \Phi_2$, there exists a $w \in \mathfrak{B}_{1\omega} \cap \mathfrak{B}_{2\omega}$ such that

$$\phi_1(w) = x_1, \tag{4.39}$$

$$\phi_2(w) = x_2. \tag{4.40}$$

Proof. (if) We need to prove that for any $\tau_i \in \mathbb{T}$ and $w_i \in \mathfrak{B}_i$, $i = 1, 2$, there exists a $\tilde{w}_i \in \pi_{\tau_i}^{-1}(\pi_{\tau_1}\mathfrak{B}_1 \parallel \pi_{\tau_2}\mathfrak{B}_2)$, $i = 1, 2$, such that
(i) $w_i D_{\mathfrak{B}_i}(\tau_i)\tilde{w}_i$, $i = 1, 2$, and
(ii) $\pi_{\tau_1}\tilde{w}_1 = \pi_{\tau_2}\tilde{w}_2$.
Denote $x_i := \phi_i(w_i, t_i)$, $i = 1, 2$. Take $w \in \mathfrak{B}_{1\omega} \cap \mathfrak{B}_{2\omega}$ such that (4.39-4.40) are satisfied. We construct $\tilde{w}_1$ and $\tilde{w}_2$ by replacing the future of w_1 and w_2 at time τ_1 and τ_2 with w.

(only if) Suppose that there exist $x_1 \in \Phi_1$ and $x_2 \in \Phi_2$ such that there does not exists a $w \in \mathfrak{B}_{1\omega} \cap \mathfrak{B}_{2\omega}$ such that (4.39-4.40) are satisfied. We know that there exist $\tau_i \in \mathbb{T}$ and $w_i \in \mathfrak{B}_i$, $i = 1, 2$, such that $\phi_i(w_i, \tau_i) = x_i$, $i = 1, 2$. By the property of the canonical minimal state map, we cannot construct $\tilde{w}_i \in \pi_{\tau_i}^{-1}(\pi_{\tau_1}\mathfrak{B}_1 \parallel \pi_{\tau_2}\mathfrak{B}_2)$, $i = 1, 2$, such that
(i) $w_i D_{\mathfrak{B}_i}(\tau_i)\tilde{w}_i$, $i = 1, 2$, and
(ii) $\pi_{\tau_1}\tilde{w}_1 = \pi_{\tau_2}\tilde{w}_2$.
Thus the interconnection cannot be uniformly compatible. □

Now we are going to discuss a more general kind of compatibility. Notice that compatibility requires that the interconnection can be made instantaneously, without any kind of preparation. A more general criterion would allow some preparation stage to take place prior to the interconnection, and thus accommodating more interconnections.

Definition 4.34. Let $w_1, w_2 \in \mathfrak{B}$ and $\tau \in \mathbb{T}$. We say that w_1 is **weakly directable** to w_2 at time t if there exists a trajectory $w_3 \in \mathfrak{B}$ and a $t' \leq t$ such that

$$w_3(\tau) = \begin{cases} w_1(t), & \tau \leq t', \\ w_2(t) & \tau > t. \end{cases}$$

Similar to the case of (strong) directability, we shall use a shorthand notation for weak directability. The fact that w_1 is *weakly directable* to w_2 at time t can be written

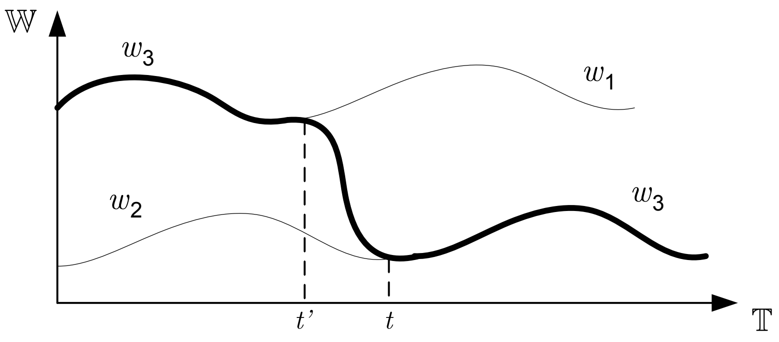


Figure 4.5: An illustration of weak directability. The trajectory w_1 is weakly directable to w_2 at time t if there exists a trajectory w_3 (thick curve) that has the past of w_1 and the future of w_2 possibly with a transitional phase as indicated in the picture.

as $w_1 D^*_{\mathfrak{B}}(t) w_2$. It is interesting to realize that the time indicated in the relation is such that for any $t' \geq t$,

$$w_1 D^*_{\mathfrak{B}}(t) w_2 \Rightarrow w_1 D^*_{\mathfrak{B}}(t') w_2.$$

See Figure 4.5 for an illustration of weak directability.

The concept of weak directability is related to *controllability* of the behavior.

Definition 4.35. Let a behavior $\mathfrak{B}$ have the type $(\mathbb{T}, \mathbb{W})$. The behavior is **controllable** if there exists a time instant $T \in \mathbb{T}$ such that for every $w_1, w_2 \in \mathfrak{B}$, there exists a $t \leq T$ such that $w_1 D^*_{\mathfrak{B}}(t) w_2$. Here T acts as a time upper bound, before which all directing must be done.

The definition of weak directability leads to the definition of weak compatibility.

Definition 4.36. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be behaviors of type $(\mathbb{T}, \mathbb{W})$. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is **weakly compatible** if there exist $\tau_1, \tau_2 \in \mathbb{T}$ such that for any $w_i \in \mathfrak{B}_i$, $i = 1, 2$, there exists a $\tilde{w}_i \in \pi^{-1}_{\tau_i}(\pi_{\tau_1} \mathfrak{B}_1 \parallel \pi_{\tau_2} \mathfrak{B}_2)$, $i = 1, 2$, such that
(i) $w_i D^*_{\mathfrak{B}_i}(\tau_i) \tilde{w}_i$, $i = 1, 2$, and
(ii) $\pi_{\tau_1} \tilde{w}_1 = \pi_{\tau_2} \tilde{w}_2$.

This definition can be interpreted as follows. If the interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is weakly compatible, then for any trajectories in $\mathfrak{B}_1$ and $\mathfrak{B}_2$, i.e. w_1 and w_2, we can always gradually direct w_1 and w_2 to a future that is accepted by the interconnection. This definition of weak compatibility is similar to the concept of *mergeable behaviors* introduced in [RW01] for LTI behaviors.

A relation between controllability and weak compatibility is presented as follows.

Theorem 4.37. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be behaviors of type $(\mathbb{T}, \mathbb{W})$. Assume that these behaviors are not disjoint. If $\mathfrak{B}_1$ and $\mathfrak{B}_2$ are both controllable, then the interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is weakly compatible.

Proof. For any $w_1 \in \mathfrak{B}_1$ and $w_2 \in \mathfrak{B}_2$, we pick any $w \in \mathfrak{B}_1 \parallel \mathfrak{B}_2$. Because of controllability of $\mathfrak{B}_1$ and $\mathfrak{B}_2$, there exist a $T_1, T_2 \in \mathbb{T}$ such that $w_1 D^*_{\mathfrak{B}_1}(T_1)w$ and $w_2 D^*_{\mathfrak{B}_2}(T_2)w$. Here w plays the role of $\tilde{w}_1$ and $\tilde{w}_2$ in Definition 4.36. Notice that by Definition 4.35, T_1 and T_2 can be chosen independently of w_1 and w_2. □

As the analog of uniform compatibility, we can define uniform weak compatibility as follows.

Definition 4.38. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be behaviors of type $(\mathbb{T}, \mathbb{W})$. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is **uniformly weakly compatible** if for any $\tau_i \in \mathbb{T}$ and $w_i \in \mathfrak{B}_i$, $i = 1, 2$, there exists a $\tilde{w}_i \in \pi^{-1}_{\tau_i}(\pi_{\tau_1}\mathfrak{B}_1 \parallel \pi_{\tau_2}\mathfrak{B}_2)$, $i = 1, 2$, such that
(i) $w_i D^*_{\mathfrak{B}_i}(\tau_i)\tilde{w}_i$, $i = 1, 2$, and
(ii) $\pi_{\tau_1}\tilde{w}_1 = \pi_{\tau_2}\tilde{w}_2$.

Remark 4.39. The notions of compatible and weakly compatible interconnections are built upon the following perception. There are two systems and they are going to be interconnected. By this statement we mean that the interconnection starts to take effect after a particular time instant t which is an element of the time axis. As an alternative to this perception, one can also think of the interconnected systems as something that exists since the beginning of the time axis. In this alternative point of view, compatibility is not an issue.

4.2.2 Compatibility for linear systems

We shall now discuss the concept of uniformly compatible interconnection for the class of linear time invariant systems. We shall exclude the class $\bar{\mathfrak{L}}^{\mathrm{q}}_{\mathrm{c}}$ from the discussion, as we need to use the concept of dynamic maps. Thus, we restrict our attention to the classes $\mathfrak{L}^{\mathrm{q}}_{\mathrm{d}}$, $\mathfrak{L}^{\mathrm{q}}_{\mathrm{c}}$ and $\overrightarrow{\mathfrak{L}}^{\mathrm{q}}_{\mathrm{c}}$.

Behaviors in $\mathfrak{L}^{\mathrm{q}}_{\mathrm{c}}$ admit canonical minimal state map, namely value of its trajectory together with all its derivatives (see Theorem 3.45). Because of this, the canonical minimal state maps of two behaviors defined over the same set of variables can never be independent. Thus, the interconnection of two $\mathfrak{L}^{\mathrm{q}}_{\mathrm{c}}$ behaviors is never uniformly compatible, except when both behaviors are the zero behavior. In this case, the state space consists of one point, namely 0.

Behaviors in $\overrightarrow{\mathfrak{L}}^{\mathrm{q}}_{\mathrm{c}}$ and $\mathfrak{L}^{\mathrm{q}}_{\mathrm{d}}$ also admit canonical minimal state map, namely the observable state space representation (see Theorems 3.46 and 3.47). Thus, if we take any $\mathfrak{B}_1$ and $\mathfrak{B}_2$ both in $\overrightarrow{\mathfrak{L}}^{\mathrm{q}}_{\mathrm{c}}$ or $\mathfrak{L}^{\mathrm{q}}_{\mathrm{d}}$, and form the interconnection $\mathfrak{B} := \mathfrak{B}_1 \parallel \mathfrak{B}_2$, Theorem 4.33 stipulates that this interconnection is uniformly compatible if and only if the canonical minimal state maps of $\mathfrak{B}_1$ and $\mathfrak{B}_2$ are independent. The following result gives a characterization of uniformly compatible interconnection.

Theorem 4.40. For any $\mathfrak{B}_1, \mathfrak{B}_2 \in \overrightarrow{\mathfrak{L}}^{\mathrm{q}}_{\mathrm{c}}$, the interconnection $\mathfrak{B} := \mathfrak{B}_1 \parallel \mathfrak{B}_2$ is uniformly compatible if and only if

$$\mathtt{n}(\mathfrak{B}) = \mathtt{n}(\mathfrak{B}_1) + \mathtt{n}(\mathfrak{B}_2). \tag{4.41}$$

Here the symbol $\mathtt{n}()$ denotes the McMillan degree of the behavior.

Proof. Denote x_1 and x_2 as the canonical minimal state maps of $\mathfrak{B}_1$ and $\mathfrak{B}_2$ respectively. In Theorem 3.46 it is shown that x_1 and x_2 can be constructed as

$$x_1 = X_1\left(\frac{d}{dt}\right)w, \tag{4.42}$$

$$x_2 = X_2\left(\frac{d}{dt}\right)w, \tag{4.43}$$

for some full row rank polynomial matrices X_1 and X_2. The number of rows in X_1 and X_2 are $\mathtt{n}(\mathfrak{B}_1)$ and $\mathtt{n}(\mathfrak{B}_2)$ respectively.

Notice that because of time invariance, we have the following relation for the suffix behaviors

$$\mathfrak{B}_{1\omega} = \{w|_{t>0} \mid w \in \mathfrak{B}_1\}, \tag{4.44}$$

$$\mathfrak{B}_{2\omega} = \{w|_{t>0} \mid w \in \mathfrak{B}_2\}, \tag{4.45}$$

$$\mathfrak{B}_{1\omega} \cap \mathfrak{B}_{2\omega} = \{w|_{t>0} \mid w \in \mathfrak{B}_1 \cap \mathfrak{B}_2\}. \tag{4.46}$$

Following Theorem 3.49, the canonical minimal state map of $\mathfrak{B}$ can be constructed as

$$x = \begin{bmatrix} X_1\left(\frac{d}{dt}\right) \\ X_2\left(\frac{d}{dt}\right) \end{bmatrix} w. \tag{4.47}$$

We need to prove that x_1 and x_2 are independent if and only if $\mathtt{n}(\mathfrak{B}) = \mathtt{n}(\mathfrak{B}_1) + \mathtt{n}(\mathfrak{B}_2)$.

(only if) Suppose that the interconnection is uniformly compatible. From Theorem 4.33, we know that this implies that for any $x_1 \in \mathbb{R}^{\mathtt{n}(\mathfrak{B}_1)}$ and $x_2 \in \mathbb{R}^{\mathtt{n}(\mathfrak{B}_2)}$ we can find a $w \in \mathfrak{B}_{1\omega} \cap \mathfrak{B}_{2\omega}$ such that

$$\lim_{t\downarrow 0} X_1\left(\frac{d}{dt}\right)w(t) = x_1, \tag{4.48a}$$

$$\lim_{t\downarrow 0} X_2\left(\frac{d}{dt}\right)w(t) = x_2. \tag{4.48b}$$

On the other hand, since the canonical minimal state map of $\mathfrak{B}$ is defined as in (4.47), this implies that the state space of $\mathfrak{B}_1 \cap \mathfrak{B}_2$ spans the whole $\mathbb{R}^{\mathtt{n}(\mathfrak{B}_1)+\mathtt{n}(\mathfrak{B}_2)}$. Hence $\mathtt{n}(\mathfrak{B}) = \mathtt{n}(\mathfrak{B}_1) + \mathtt{n}(\mathfrak{B}_2)$.

(if) Suppose that $\mathtt{n}(\mathfrak{B}) = \mathtt{n}(\mathfrak{B}_1) + \mathtt{n}(\mathfrak{B}_2)$. This implies that for any $x_1 \in \mathbb{R}^{\mathtt{n}(\mathfrak{B}_1)}$ and $x_2 \in \mathbb{R}^{\mathtt{n}(\mathfrak{B}_2)}$ we can find a $w \in \mathfrak{B}_{1\omega} \cap \mathfrak{B}_{2\omega}$ such that (4.48) is satisfied. Therefore, according to Theorem 4.33, the interconnection is uniformly compatible. $\square$

The analog of Theorem 4.40 for behaviors in $\mathfrak{L}_{\mathrm{d}}^{\mathrm{q}}$ is as follows.

Theorem 4.41. For any $\mathfrak{B}_1, \mathfrak{B}_2 \in \mathfrak{L}_{\mathrm{d}}^{\mathrm{q}}$, the interconnection $\mathfrak{B} := \mathfrak{B}_1 \parallel \mathfrak{B}_2$ is uniformly compatible if and only if

$$\mathtt{n}(\mathfrak{B}) = \mathtt{n}(\mathfrak{B}_1) + \mathtt{n}(\mathfrak{B}_2). \tag{4.49}$$

Here the symbol $\mathtt{n}()$ denotes the McMillan degree of the behavior.

Proof. Denote x_1 and x_2 as the canonical minimal state maps of $\mathfrak{B}_1$ and $\mathfrak{B}_2$ respectively. In Theorem 3.47 it is shown that x_1 and x_2 can be constructed as

$$x_1 = X_1(\sigma)\, w, \tag{4.50}$$
$$x_2 = X_2(\sigma)\, w, \tag{4.51}$$

for some full row rank polynomial matrices X_1 and X_2. The number of rows in X_1 and X_2 are $\mathtt{n}(\mathfrak{B}_1)$ and $\mathtt{n}(\mathfrak{B}_2)$ respectively.

Notice that because of time invariance, we have the following relation for the suffix behaviors

$$\mathfrak{B}_{1\omega} = \{w|_{t>0} \mid w \in \mathfrak{B}_1\}, \tag{4.52}$$
$$\mathfrak{B}_{2\omega} = \{w|_{t>0} \mid w \in \mathfrak{B}_2\}, \tag{4.53}$$
$$\mathfrak{B}_{1\omega} \cap \mathfrak{B}_{2\omega} = \{w|_{t>0} \mid w \in \mathfrak{B}_1 \cap \mathfrak{B}_2\}. \tag{4.54}$$

Following Theorem 3.49, the canonical minimal state map of $\mathfrak{B}$ can be constructed as

$$x = \begin{bmatrix} X_1(\sigma) \\ X_2(\sigma) \end{bmatrix} w. \tag{4.55}$$

We need to prove that x_1 and x_2 are independent if and only if $\mathtt{n}(\mathfrak{B}) = \mathtt{n}(\mathfrak{B}_1) + \mathtt{n}(\mathfrak{B}_2)$.

(only if) Suppose that the interconnection is uniformly compatible. From Theorem 4.33, we know that this implies that for any $x_1 \in \mathbb{R}^{\mathtt{n}(\mathfrak{B}_1)}$ and $x_2 \in \mathbb{R}^{\mathtt{n}(\mathfrak{B}_2)}$ we can find a $w \in \mathfrak{B}_1 \cap \mathfrak{B}_2$ such that

$$X_1(\sigma)\, w(k)|_{k=0} = x_1, \tag{4.56a}$$
$$X_2(\sigma)\, w(k)|_{k=0} = x_2. \tag{4.56b}$$

On the other hand, since the canonical minimal state map of $\mathfrak{B}$ is defined as in (4.55), this implies that the state space of $\mathfrak{B}_1 \cap \mathfrak{B}_2$ spans the whole $\mathbb{R}^{\mathtt{n}(\mathfrak{B}_1)+\mathtt{n}(\mathfrak{B}_2)}$. Hence $\mathtt{n}(\mathfrak{B}) = \mathtt{n}(\mathfrak{B}_1) + \mathtt{n}(\mathfrak{B}_2)$.

(if) Suppose that $\mathtt{n}(\mathfrak{B}) = \mathtt{n}(\mathfrak{B}_1) + \mathtt{n}(\mathfrak{B}_2)$. This implies that for any $x_1 \in \mathbb{R}^{\mathtt{n}(\mathfrak{B}_1)}$ and $x_2 \in \mathbb{R}^{\mathtt{n}(\mathfrak{B}_2)}$ we can find a $w \in \mathfrak{B}_1 \cap \mathfrak{B}_2$ such that (4.56) is satisfied. Therefore, according to Theorem 4.33, the interconnection is uniformly compatible. $\square$

Theorems 4.40 and 4.41 provide us a characterization of uniformly compatible interconnection for behaviors in $\overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$ and $\mathfrak{L}_{\mathrm{d}}^{\mathrm{q}}$.

The type of interconnections characterized by the fact that the McMillan degree of the systems add up to McMillan degree of the interconnected system is known as *regular feedback interconnection [Wil97]*. This type of interconnections is related to some input-output partitioning of the variables.

Definition 4.42. *[Wil97]* Let $\mathfrak{B} \in \overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$, such that it is given by the following representation

$$\mathfrak{B} = \left\{ w \mid R\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.57}$$

Suppose that we reorder and partition the variables in $\mathbf{w}$ into $\mathbf{u}$ and $\mathbf{y}$ such that

$$\mathfrak{B} = \left\{ (u, y) \mid R_1\left(\frac{d}{dt}\right) u + R_2\left(\frac{d}{dt}\right) y = 0 \right\}, \tag{4.58}$$

where $[R_1\ R_2]$ is a full row rank matrix. This partition is called a **proper input-output** partition, where $\mathbf{u}$ is the input and $\mathbf{y}$ is the output if
(i) $R_2(\xi)$ is a square matrix with nonzero determinant,
(ii) $R_2^{-1}(\xi)R_1(\xi)$ is a proper rational matrix, i.e. its entries are proper rational functions.
The rational matrix $R_2^{-1}(\xi)R_1(\xi)$ is called the **transfer matrix** of the input-output partition.

The characterization of regular feedback interconnection in terms of the proper input-output partition is given as follows.

Theorem 4.43. [Wil97] Let $\mathfrak{B}_1, \mathfrak{B}_2$ be elements of $\overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$. If the interconnection $\mathfrak{B} := \mathfrak{B}_1 \parallel \mathfrak{B}_2$ is a regular feedback interconnection then the variables in the behaviors can be partitioned into $\mathbf{u}$, $\mathbf{y}_1$, and $\mathbf{y}_2$ such that
(i) in $\mathfrak{B}_1$, $\mathbf{u}$ and $\mathbf{y}_2$ are input, $\mathbf{y}_1$ is output and the transfer function is proper,
(ii) in $\mathfrak{B}_2$, $\mathbf{u}$ and $\mathbf{y}_1$ are input, $\mathbf{y}_2$ is output and the transfer function is proper,
(iii) in $\mathfrak{B}$, $\mathbf{u}$ is input, $\mathbf{y}_1$ and $\mathbf{y}_2$ are output and the transfer function is proper.

This theorem shows that a regular feedback interconnection can be seen as a standard feedback interconnection of input-output systems. This is evident if we look at the diagram in Figure 4.6.

Controllability of linear time invariant systems is a very well known concept, originating from the work of Kalman in the 1960s. The original definition of controllability was cast in the state space representation, where a system is controllable if it is possible to go from any state to any other state in a finite time. Controllability of linear time invariant systems is recast in the behavioral framework by Willems in e.g. [Wil97, PW98].

Definition 4.44. (cf. Definition 5.2.2 in [PW98]) Let $\mathfrak{B}$ be a behavior of a time invariant dynamical systems. The system is called **controllable** if for any two trajectories $w_1, w_2 \in \mathfrak{B}$ there exists a $t_1 \geq 0$ and a trajectory $w \in \mathfrak{B}$ such that

$$w(t) = \left\{ \begin{array}{ll} w_1(t) & t \leq 0 \\ w_2(t) & t > t_1 \end{array} \right. . \tag{4.59}$$

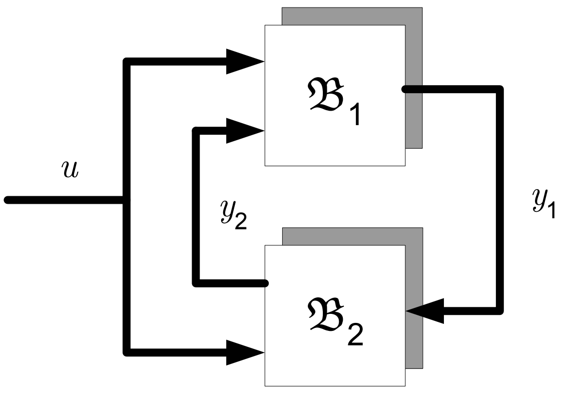


Figure 4.6: Regular feedback interconnection.

For LTI systems, Definition 4.35 and Definition 4.44 are equivalent.

Linear time invariant behaviors have a special property that to every behavior $\mathfrak{B}$ we can uniquely associate a behavior $\mathfrak{B}^{\mathrm{ctr}}$, which is the largest subbehavior of $\mathfrak{B}$ that is controllable [PW98]. This subbehavior is called the *controllable* part of $\mathfrak{B}$. Since $\mathfrak{B}^{\mathrm{ctr}}$ is a linear subspace of $\mathfrak{B}$, we can write

$$\mathfrak{B} = \mathfrak{B}^{\mathrm{ctr}} \oplus \mathfrak{B}^{\mathrm{aut}}, \tag{4.60}$$

for some $\mathfrak{B}^{\mathrm{aut}}$. Notice that although for every LTI behavior $\mathfrak{B}$, its controllable part $\mathfrak{B}^{\mathrm{ctr}}$ is uniquely defined, the subbehavior $\mathfrak{B}^{\mathrm{aut}}$ is not unique.

Theorem 4.37 gives us a sufficient condition for weak compatibility. As its consequence, we know that interconnection of controllable linear time invariant behaviors is always weakly compatible. Stronger yet, it is also uniformly weakly compatible, because of time invariance.

A necessary and sufficient condition for uniform weak compatibility for linear time invariant systems, i.e. $\mathfrak{L}^{\mathrm{q}}_{\mathrm{c}}$, $\mathfrak{L}^{\mathrm{q}}_{\mathrm{c}}$, $\bar{\mathfrak{L}}^{\mathrm{q}}_{\mathrm{c}}$, and $\overrightarrow{\mathfrak{L}}^{\mathrm{q}}_{\mathrm{c}}$, is given in the following theorem.

Theorem 4.45. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be LTI behaviors. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is uniformly weakly compatible if and only if

$$\mathfrak{B}_1 + \mathfrak{B}_2 = \mathfrak{B}_1^{\mathrm{ctr}} + \mathfrak{B}_2^{\mathrm{ctr}}, \tag{4.61}$$

where $\mathfrak{B}_i^{\mathrm{ctr}}$ is the controllable part of $\mathfrak{B}_i$, for $i = 1, 2$.

Proof. (only if) Take any w_1 and w_2 in $\mathfrak{B}_1$ and $\mathfrak{B}_2$ respectively. Define the set valued mapping $\varphi_i : \mathfrak{B}_i \to 2^{\mathfrak{B}_i}$, $i = 1, 2$, as

$$\varphi_i(w) := \{w' \in \mathfrak{B}_i \mid \forall t \in \mathbb{R},\ wD^*_{\mathfrak{B}_i}(t)w'\}.$$

It is known (see [PW98]) that

$$\varphi_i(w_i) = w_i + \mathfrak{B}_i^{\mathrm{ctr}},\ i = 1, 2.$$

If the interconnection is uniformly weakly compatible, then for any w_1 and w_2 in $\mathfrak{B}_1$ and $\mathfrak{B}_2$ respectively, the following relation holds.

$$\varphi_1(w_1) \cap \varphi_2(w_2) = (w_1 + \mathfrak{B}_1^{\mathrm{ctr}}) \cap (w_2 + \mathfrak{B}_2^{\mathrm{ctr}}) \neq \emptyset. \tag{4.62}$$

By taking $w_1 = 0$ and $w_2 = 0$ respectively, we can deduce that

$$\mathfrak{B}_1 = \mathfrak{B}_1^{\mathrm{ctr}} + \mathfrak{B}_2^{\mathrm{ctr}}, \tag{4.63a}$$

$$\mathfrak{B}_2 = \mathfrak{B}_1^{\mathrm{ctr}} + \mathfrak{B}_2^{\mathrm{ctr}}. \tag{4.63b}$$

Equation (4.61) follows straightforward from here.

(if) Assuming that (4.61) is true, we get (4.63). Consequently, we have that for any w_1 and w_2 in $\mathfrak{B}_1$ and $\mathfrak{B}_2$ respectively, we can write

$$w_1 = w_{11} + w_{12},$$
$$w_2 = w_{21} + w_{22},$$

where $w_{11}, w_{21} \in \mathfrak{B}_1^{\mathrm{ctr}}$ and $w_{12}, w_{22} \in \mathfrak{B}_2^{\mathrm{ctr}}$. To prove that the interconnection is uniformly weakly compatible, we need to show that (4.62) holds. Define $w := w_{21} + w_{12}$. Obviously we have that $w \in (w_i + \mathfrak{B}_i^{\mathrm{ctr}})$, $i = 1, 2$. Hence, (4.62) holds and the interconnection is uniformly weakly compatible. □

We relax the condition in Definition 4.42 to define *input-output partition*.

Definition 4.46. Let $\mathfrak{B}$ be a linear time invariant behavior given by the following representation

$$\mathfrak{B} = \{w \mid R(\sigma)w = 0\}, \tag{4.64}$$

if $\mathfrak{B}$ is discrete time or

$$\mathfrak{B} = \{w \mid R(\frac{d}{dt})w = 0\}, \tag{4.65}$$

if $\mathfrak{B}$ is continuous time. Suppose that we reorder and partition the variables in $\mathbf{w}$ into $\mathbf{u}$ and $\mathbf{y}$ such that

$$\mathfrak{B} = \{(u, y) \mid R_1(\sigma)u + R_2(\sigma)y = 0\}, \tag{4.66}$$

or

$$\mathfrak{B} = \{(u, y) \mid R_1\left(\frac{d}{dt}\right)u + R_2\left(\frac{d}{dt}\right)y = 0\},$$

where $[R_1 \; R_2]$ is a full row rank matrix. This partition is called an **input-output** partition, where $\mathbf{u}$ is the input and $\mathbf{y}$ is the output if $R_2(\xi)$ is a square matrix with nonzero determinant.

Clearly, any proper input-output partition is also an input-output partition. The difference is that we do not require the transfer matrix from the input to the output to be proper. The number of output variables in an LTI behavior is equal to the number of rows in its minimal kernel representation. The number of output variables in an LTI behavior $\mathfrak{B}$ is denoted as $\mathtt{p}(\mathfrak{B})$. The notion of *regular interconnection* is then defined as follows [Wil97, BT02].

Definition 4.47. [Wil97] Let $\mathfrak{B}_1, \mathfrak{B}_2$ be linear time invariant behaviors of the same type. The interconnection $\mathfrak{B} := \mathfrak{B}_1 \parallel \mathfrak{B}_2$ is a **regular interconnection** if

$$\mathtt{p}(\mathfrak{B}) = \mathtt{p}(\mathfrak{B}_1) + \mathtt{p}(\mathfrak{B}_2). \tag{4.67}$$

We can formulate a Theorem analogous to the characterization of regular feedback interconnection in Theorem 4.43.

Theorem 4.48. Let $\mathfrak{B}_1, \mathfrak{B}_2$ be linear time invariant behaviors of the same type. If the interconnection $\mathfrak{B} := \mathfrak{B}_1 \parallel \mathfrak{B}_2$ is a regular interconnection, then the variables in the behaviors can be partitioned into $\mathbf{u}$, $\mathbf{y}_1$, and $\mathbf{y}_2$ such that
(i) in $\mathfrak{B}_1$, $\mathbf{u}$ and $\mathbf{y}_2$ are input, $\mathbf{y}_1$ is output,
(ii) in $\mathfrak{B}_2$, $\mathbf{u}$ and $\mathbf{y}_1$ are input, $\mathbf{y}_2$ is output,
(iii) in $\mathfrak{B}$, $\mathbf{u}$ is input, $\mathbf{y}_1$ and $\mathbf{y}_2$ are output.

It turns out that uniform weak compatibility is a weaker notion than regularity of the interconnection. This result is presented in the following lemma.

Lemma 4.49. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be $\mathfrak{L}^{\mathtt{q}}_{\mathrm{c}}$ behaviors defined by

$$\mathfrak{B}_i := \left\{ w \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^{\mathtt{w}}) \mid R_i\left(\frac{d}{dt}\right) w = 0 \right\}, i = 1, 2,$$

where R_1 and R_2 are full-row-rank polynomial matrices with g_1 and g_2 rows respectively and q columns. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is uniformly weakly compatible if it is regular. The converse is generally not true.

Proof. Assume that the interconnection is regular. We shall use Theorem 4.45 to prove that it is also uniformly weakly compatible. The behavior $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ contains all $w \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^{\mathtt{w}})$ characterized by

$$\begin{bmatrix} R_1 \\ R_2 \end{bmatrix} \left(\frac{d}{dt}\right) w = 0. \tag{4.68}$$

We assume that at least one of $\mathfrak{B}_1$ and $\mathfrak{B}_2$ is uncontrollable. Otherwise the interconnection is uniformly weakly compatible by directly applying Theorem 4.45. Without any loss of generality, assume that $\mathfrak{B}_1$ is uncontrollable. We can always find two unimodular matrices U_1 and V such that $U_1 R_1 V$ is diagonal.

$$U_1 R_1 V =: \tilde{R}_1 = \begin{bmatrix} I & 0 & 0 \\ 0 & D(\xi) & 0 \end{bmatrix}, \tag{4.69}$$

with I the identity matrix and $D(\xi)$ some diagonal polynomial matrix. Denote $\tilde{R}_2 := R_2 V$. The interconnected behavior is given by the following kernel representation.

$$\begin{bmatrix} I & 0 & 0 \\ 0 & D & 0 \\ \tilde{R}_{21} & \tilde{R}_{22} & \tilde{R}_{23} \end{bmatrix} \left(\frac{d}{dt}\right) \tilde{w} = 0, \tag{4.70}$$

where

$$\tilde{w} := V^{-1}\left(\frac{d}{dt}\right) w. \tag{4.71}$$

Notice that since the interconnection is regular, $\tilde{R}_{23}$ is full row rank.

In the following, we shall show that $\mathfrak{B}_1 = \mathfrak{B}_1^{\text{ctr}} + \mathfrak{B}_2^{\text{ctr}}$. Take any trajectory $\tilde{w} := (\tilde{w}_1, \tilde{w}_2, \tilde{w}_3) \in \mathfrak{B}_1$. Here the variables in $\tilde{\mathbf{w}}$ are partitioned according to the columns of the polynomial matrix in (4.70). Necessarily, $\tilde{w}_1 = 0$ and $\tilde{w}_2 \in \ker D\left(\frac{d}{dt}\right)$. Now, since $\tilde{R}_{23}$ is full row rank, there exists a $\tilde{v}_3$ such that

$$\tilde{R}_{22}\left(\frac{d}{dt}\right)\tilde{w}_2 = \tilde{R}_{23}\left(\frac{d}{dt}\right)\tilde{v}_3,$$

and $\tilde{v} := (0, \tilde{w}_2, \tilde{v}_3) \in \mathfrak{B}_2^{\text{ctr}}$. Also notice that $\tilde{w} - \tilde{v} = (0, 0, \tilde{w}_3 - \tilde{v}_3) \in \mathfrak{B}_1^{\text{ctr}}$. Therefore, we have shown that

$$\mathfrak{B}_1 = \mathfrak{B}_1^{\text{ctr}} + \mathfrak{B}_2^{\text{ctr}}. \tag{4.72}$$

If $\mathfrak{B}_2$ is controllable, it is trivially true that

$$\mathfrak{B}_2 \subset \mathfrak{B}_1^{\text{ctr}} + \mathfrak{B}_2^{\text{ctr}}. \tag{4.73}$$

Combining (4.72) and (4.73), we get

$$\mathfrak{B}_1 + \mathfrak{B}_2 \subset \mathfrak{B}_1^{\text{ctr}} + \mathfrak{B}_2^{\text{ctr}},$$

which when combined with the trivial inclusion $\mathfrak{B}_1 + \mathfrak{B}_2 \supset \mathfrak{B}_1^{\text{ctr}} + \mathfrak{B}_2^{\text{ctr}}$ yields

$$\mathfrak{B}_1 + \mathfrak{B}_2 = \mathfrak{B}_1^{\text{ctr}} + \mathfrak{B}_2^{\text{ctr}}. \tag{4.74}$$

If $\mathfrak{B}_2$ is uncontrollable, applying the same procedure as we have done to $\mathfrak{B}_1$, we yield (see (4.72))

$$\mathfrak{B}_2 = \mathfrak{B}_1^{\text{ctr}} + \mathfrak{B}_2^{\text{ctr}}. \tag{4.75}$$

Again, combining (4.72) and (4.75) yields (4.74). Hence by Theorem 4.45, the interconnection is uniformly weakly compatible.

To prove that the converse is not true, consider the following counterexample. Take $R_1 = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$, and $R_2 = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$. Clearly $\mathfrak{B}_1$ and $\mathfrak{B}_2$ are controllable and, by Theorem 4.45, $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is uniformly weakly compatible. However, $\begin{bmatrix} R_1^T & R_2^T \end{bmatrix}^T$ does not have full row rank. Hence $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is not regular. $\square$

Remark 4.50. The counterexample in the proof of Lemma 4.49 is also an example of a regular feedback interconnection, which is not a regular interconnection.

Notice that the result in Lemma 4.49 is restricted to behaviors is $\mathfrak{L}_c^q$. We can easily extend this result to behaviors in $\overrightarrow{\mathfrak{L}}_c^q$ and $\bar{\mathfrak{L}}_c^q$ as follows.

Lemma 4.51. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be both $\bar{\mathfrak{L}}_c^q$ or $\overrightarrow{\mathfrak{L}}_c^q$ behaviors. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is uniformly weakly compatible if it is regular. The converse is generally not true.

Proof. First, we consider the case when both $\mathfrak{B}_1$ and $\mathfrak{B}_2$ are in $\bar{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$. Denote the infinitely differentiable part of $\mathfrak{B}_1$ and $\mathfrak{B}_2$ as

$$\tilde{\mathfrak{B}}_1 := \mathfrak{B}_1 \cap \mathfrak{C}^{\infty}(\mathbb{R}, \mathbb{R}^{\mathrm{q}}), \tag{4.76}$$

$$\tilde{\mathfrak{B}}_2 := \mathfrak{B}_2 \cap \mathfrak{C}^{\infty}(\mathbb{R}, \mathbb{R}^{\mathrm{q}}). \tag{4.77}$$

In Lemma 4.49 we have shown that if the interconnection is regular, then

$$\tilde{\mathfrak{B}}_1 + \tilde{\mathfrak{B}}_2 = \tilde{\mathfrak{B}}_1^{\mathrm{ctr}} + \tilde{\mathfrak{B}}_2^{\mathrm{ctr}}. \tag{4.78}$$

In [PW98], it is proven that $\mathfrak{B}_1^{\mathrm{ctr}}$ is the closure of $\tilde{\mathfrak{B}}_1^{\mathrm{ctr}}$ in the topology of locally integrable functions $\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R}, \mathbb{R}^{\mathrm{q}})$. Similarly $\mathfrak{B}_2^{\mathrm{ctr}}$ is the closure of $\tilde{\mathfrak{B}}_2^{\mathrm{ctr}}$. Therefore, by taking the closure of both sides of (4.78), we obtain

$$\mathfrak{B}_1 + \mathfrak{B}_2 = \mathfrak{B}_1^{\mathrm{ctr}} + \mathfrak{B}_2^{\mathrm{ctr}}. \tag{4.79}$$

Hence the interconnection is uniformly weakly compatible.

For the case when both $\mathfrak{B}_1$ and $\mathfrak{B}_2$ are in $\overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathrm{q}}$, consider the closure of $\mathfrak{B}_1$ and $\mathfrak{B}_2$, denoted as $\bar{\mathfrak{B}}_1$ and $\bar{\mathfrak{B}}_2$. We have shown that if the interconnection is regular then

$$\bar{\mathfrak{B}}_1 + \bar{\mathfrak{B}}_2 = \bar{\mathfrak{B}}_1^{\mathrm{ctr}} + \bar{\mathfrak{B}}_2^{\mathrm{ctr}}. \tag{4.80}$$

If we take the left continuous part of both sides of (4.80), we obtain (4.79).

Finally, to show that the converse statement of the lemma is generally not true, we can use the same counterexample as in the proof of Lemma 4.49. □

We can also pose a corollary of Lemma 4.49 for behaviors in $\mathfrak{L}_{\mathrm{d}}^{\mathrm{q}}$.

Corollary 4.52. Let $\mathfrak{B}_1$ and $\mathfrak{B}_2$ be behaviors in $\mathfrak{L}_{\mathrm{d}}^{\mathrm{q}}$ defined by

$$\mathfrak{B}_i := \{w \mid R_i(\sigma)\, w = 0\}, i = 1, 2, \tag{4.81}$$

where R_1 and R_2 are full-row-rank polynomial matrices with g_1 and g_2 rows respectively and q columns. The interconnection $\mathfrak{B}_1 \parallel \mathfrak{B}_2$ is uniformly weakly compatible if it is regular. The converse is generally not true.

A proof of this corollary can be constructed in a way analogous to that of Lemma 4.49.

So far we have discussed uniform compatibility and uniform weak compatibility of interconnections of linear systems. However, because of time invariance, uniform compatibility is equivalent to compatibility. Similarly, uniform weak compatibility is equivalent to weak compatibility. As a summary of the discussion about compatibility of linear behaviors, in Figure 4.7 we present a diagram that describes the relation between various notions of compatibility that we have discussed so far.

In the following example, we present a noncompatible interconnection of two behaviors, which is weakly compatible. This example is an adaptation from the door closing mechanism example in [Wil97].

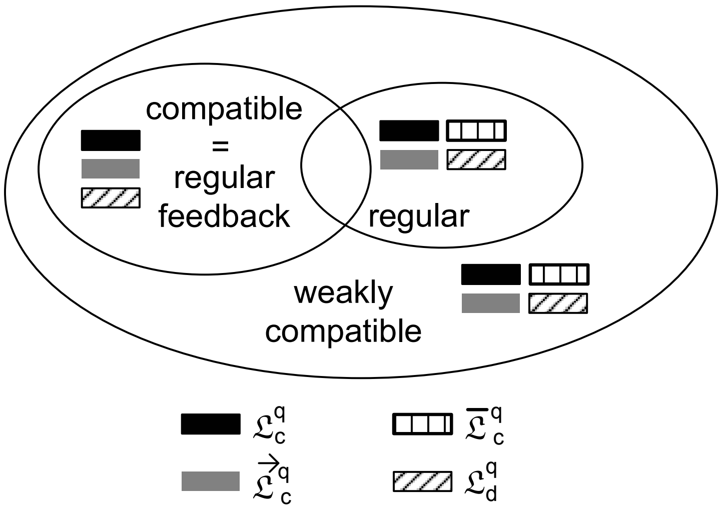


Figure 4.7: Various notions of compatibility of linear systems.

Example 4.53. Consider a moving mass that can slide without friction on a surface. We model the dynamics of this system as a behavior in $\overrightarrow{\mathfrak{L}}^2_c$ given by the following kernel representation

$$\mathfrak{B}_1 = \left\{(x, F) \mid \frac{d^2}{dt^2}x - F = 0\right\}. \tag{4.82}$$

Here the variables are $\mathbf{x}$ and $\mathbf{F}$: the position of the block mass and the total of external force acting on it respectively. This is a second order system, and thus its McMillan degree is 2. To the block mass we can attach a spring and damper system, whose dynamics can be modelled as

$$\mathfrak{B}_2 = \left\{(x, F) \mid \frac{d}{dt}x + x + F = 0\right\}.$$

This is a first order system, and thus its McMillan degree is 1. The interconnected system $\mathfrak{B} := \mathfrak{B}_1 \parallel \mathfrak{B}_2$ is represented by

$$\begin{bmatrix} \frac{d^2}{dt^2} & -1 \\ \frac{d}{dt} + 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ F \end{bmatrix} = 0. \tag{4.83}$$

It is easy to see that the McMillan degree of $\mathfrak{B}$ is 2. Thus, the interconnection is not compatible. However, it can also be verified that the interconnection is weakly compatible. Indeed, we intuitively know that before we can interconnect the mass and the spring-damper system, we must prepare the trajectory. This is done, for example, by matching x and $\frac{dx}{dt}$ in both behaviors [Wil97].

4.2.3 Achievability with compatibility constraint

In this subsection, we shall discuss how the compatibility constraint affects the achievability of a specification in a control problem. Since all the results we have so far are about linear systems, we shall restrict our attention to this class of systems.

In the presence of compatibility constraint, the formulation of the control problem becomes as follows.

Compatible control problem Given a system called the *plant*. The problem is to find a behavior (called the *controller*), with the following properties.
(i) The interconnection between the controller and the plant is compatible (or weakly compatible).
(ii) When interconnected with the plant behavior in a specified manner (in terms of a projection acting on the plant), the controller yields the *specification*.

In a more exact formulation, the problem can be described as illustrated in Figure 4.1. Given a plant to be controlled. It has two kinds of variables:
(i) *to-be-controlled variables*, which are denoted as $\mathbf{w}$ and
(ii) *control variables*, which are denoted as $\mathbf{c}$.
The size of $\mathbf{w}$ and $\mathbf{c}$ are denoted as $\mathtt{w}$ and $\mathtt{c}$ respectively. Since we are discussing about linear behaviors, the full plant behavior can be expressed in the kernel representation as

$$\mathcal{P} = \left\{ (w,c) \mid R\left(\frac{d}{dt}\right) w + M\left(\frac{d}{dt}\right) c = 0 \right\}, \tag{4.84}$$

or, in the discrete time case,

$$\mathcal{P} = \left\{ (w,c) \mid R(\sigma)\, w + M(\sigma)\, c = 0 \right\}. \tag{4.85}$$

The specification $\mathcal{S}$ is a behavior expressed in terms of the to-be-controlled variables. Thus it can be expressed as

$$\mathcal{S} = \left\{ w \mid S\left(\frac{d}{dt}\right) w = 0 \right\}, \tag{4.86}$$

or, in the discrete time case,

$$\mathcal{S} = \left\{ w \mid S(\sigma)\, w = 0 \right\}. \tag{4.87}$$

The control problem amounts to finding a controller $\mathcal{C}$, which is a behavior expressed in terms of the control variables, such that

$$\pi_s \pi_c^{-1} (\pi_c \mathcal{P} \parallel \mathcal{C}) = \mathcal{S}. \tag{4.88}$$

Here the projections π_s and π_c correspond to the elimination of $\mathbf{c}$ and $\mathbf{w}$ respectively. This is a formulation that we have discussed earlier in this chapter.

Recall our discussion about elimination of variables in the previous chapter. For behaviors in $\overrightarrow{\mathfrak{L}}_{\mathtt{c}}^{\mathtt{q}}$ and $\bar{\mathfrak{L}}_{\mathtt{c}}^{\mathtt{q}}$, exact elimination is not always possible. However, in the discussion about control problem, we shall assume that we can always do the elimination corresponding to π_s. Therefore, instead of getting the exact projection, we shall get the closure of the projection in the topology of $\mathfrak{L}_1^{\mathrm{loc}}(\mathbb{R},\mathbb{R}^{\mathtt{q}})$. This is not too bad, because it means that although we might not get the desired specification exactly, we can approximate any trajectories with any desired accuracy. We are not concerned by the elimination related to π_c, because we shall see later that the interconnection between the plant and the controller can be regarded as a full interconnection.

Now we shall incorporate a compatibility constraint into the control problem. As we have seen in the previous subsection, compatibility and weak compatibility are defined for full interconnection. To apply these notions in this problem, we use the fact that the behavior $\mathcal{C}$, which is a behavior of type $(\mathbb{R},\mathbb{R}^{\mathtt{c}})$ or $(\mathbb{Z},\mathbb{Z}^{\mathtt{c}})$, can be seen as a behavior of type $(\mathbb{R},\mathbb{R}^{\mathtt{w}+\mathtt{c}})$ or $(\mathbb{Z},\mathbb{Z}^{\mathtt{w}+\mathtt{c}})$. Suppose that $\mathcal{C}$ is represented as

$$\mathcal{C}=\left\{c \mid C\left(\frac{d}{dt}\right)c=0\right\} \text{ or } \mathcal{C}=\left\{c \mid C\left(\sigma\right)c=0\right\},$$

we can also express it as

$$\mathcal{C}'=\left\{(w,c) \mid C\left(\frac{d}{dt}\right)c=0\right\} \text{ or } \mathcal{C}'=\left\{(w,c) \mid C\left(\sigma\right)c=0\right\}.$$

Hence $\mathtt{w}$ is free in $\mathcal{C}'$. The compatibility constraint then takes the form of requiring the interconnection $\mathcal{P}\parallel\mathcal{C}'$ to be compatible or weakly compatible.

As we have seen earlier, compatible interconnection is equivalent to regular feedback interconnection. Thus, requiring $\mathcal{P}\parallel\mathcal{C}'$ to be compatible is equivalent to requiring that $\mathcal{P}\parallel\mathcal{C}'$ is a regular feedback interconnection. To find necessary and sufficient conditions for a control problem to be solvable with a regular feedback interconnection is not yet solved. The problem is called 'regular feedback implementability of linear differential behavior'. It is an open problem in the systems and control community [Tre04].

If we relax the requirement and aim to have a controller such that $\mathcal{P}\parallel\mathcal{C}'$ is weakly compatible, we can derive necessary and sufficient conditions for it. In fact, we shall show that the control problem is solvable by a weakly compatible controller if and only if it is solvable by a regular controller. By weakly compatible controller and regular controller we mean controllers whose interconnection with the plant $\mathcal{P}$ is weakly compatible and regular respectively.

Lemma 4.54. Let the plant be given as

$$\mathcal{P}=\left\{(w,c) \mid R\left(\frac{d}{dt}\right)w+M\left(\frac{d}{dt}\right)c=0\right\}, \tag{4.89}$$

and the controller as

$$\mathcal{C}'=\left\{(w,c) \mid C\left(\frac{d}{dt}\right)c=0\right\}. \tag{4.90}$$

The interconnection $\mathcal{P} \parallel \mathcal{C}'$ is weakly compatible if and only if $\pi_c\mathcal{P} \parallel \mathcal{C}$ is weakly compatible, where

$$\mathcal{C} = \left\{ c \mid C\left(\frac{d}{dt}\right) c = 0 \right\}. \tag{4.91}$$

Proof. (if) For brevity, denote $\mathcal{P}_c := \pi_c\mathcal{P}$. Suppose that $\mathcal{P}_c \parallel \mathcal{C}$ is weakly compatible. Then,

$$\mathcal{P}_c + \mathcal{C} = \mathcal{P}_c^{\mathrm{ctr}} + \mathcal{C}^{\mathrm{ctr}}. \tag{4.92}$$

We need to show that

$$\mathcal{P} + \mathcal{C}' = \mathcal{P}^{\mathrm{ctr}} + \mathcal{C}'^{\mathrm{ctr}}. \tag{4.93}$$

Suppose that $\mathcal{P}_c + \mathcal{C}$ is represented by $L(\frac{d}{dt})c = 0$, since $\mathbf{w}$ is free in $\mathcal{C}'$, we can represent $\mathcal{P} + \mathcal{C}'$ by

$$\mathcal{P} + \mathcal{C}' = \left\{ (w, c) \mid L\left(\frac{d}{dt}\right) c = 0 \right\}. \tag{4.94}$$

Take any $(w, c) \in \mathcal{P} + \mathcal{C}'$. We have that $c \in \mathcal{P}_c + \mathcal{C}$. Hence we can write $c = c_1 + c_2$, where $c_1 \in \mathcal{P}_c^{\mathrm{ctr}}$ and $c_2 \in \mathcal{C}^{\mathrm{ctr}}$. We can always find a w_1, such that $(w_1, c_1) \in \mathcal{P}^{\mathrm{ctr}}$. Furthermore, we have that $(w - w_1, c_2) \in \mathcal{C}'^{\mathrm{ctr}}$. Therefore,

$$(w, c) = (w_1, c_1) + (w - w_1, c_2), \tag{4.95}$$

and thus $(w, c) \in \mathcal{P}^{\mathrm{ctr}} + \mathcal{C}'^{\mathrm{ctr}}$.

(only if) Suppose that $\mathcal{P} \parallel \mathcal{C}'$ is weakly compatible. Then, (4.93) holds. We need to show that (4.92) also holds. Take any $c \in \mathcal{P}_c + \mathcal{C}$. Take any w, we have that $(w, c) \in \mathcal{P} + \mathcal{C}'$. By (4.93), we can find always $(w_1, c_1) \in \mathcal{P}^{\mathrm{ctr}}$ and $(w - w_1, c - c_1) \in \mathcal{C}'^{\mathrm{ctr}}$. However, this means $c_1 \in \mathcal{P}_c^{\mathrm{ctr}}$ and $(c - c_1) \in \mathcal{C}^{\mathrm{ctr}}$. Thus, $c \in \mathcal{P}_c^{\mathrm{ctr}} + \mathcal{C}^{\mathrm{ctr}}$. □

A similar result for discrete time linear systems can be derived, by replacing the $(\frac{d}{dt})$ operator with σ. From here we can conclude that requiring that $\mathcal{P} \parallel \mathcal{C}'$ is weakly compatible is equivalent to requiring that $\pi_c\mathcal{P} \parallel \mathcal{C}$ is weakly compatible. Thus, the control problem with weak compatibility constraint can be expressed as follows.

Problem 4.55. Given the plant $\mathcal{P}$, where

$$\mathcal{P} = \left\{ (w, c) \mid R\left(\frac{d}{dt}\right) w + M\left(\frac{d}{dt}\right) c = 0 \right\}, \tag{4.96}$$

and the specification $\mathcal{S}$, where

$$\mathcal{S} = \left\{ w \mid S\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.97}$$

Find a controller $\mathcal{C}$ of the form

$$\mathcal{C} = \left\{ c \mid C\left(\frac{d}{dt}\right) c = 0 \right\} \tag{4.98}$$

such that
(i) $\pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \mathcal{C}) = \mathcal{S}$,
(ii) $\pi_c \mathcal{P} \parallel \mathcal{C}$ is weakly compatible.

If we replace $\frac{d}{dt}$ in this problem by σ, we get its discrete time version. We then use the following lemma [JS03].

Lemma 4.56. Let $\mathcal{P}$ and $\mathcal{S}$ be linear behaviors characterized by

$$\mathcal{P} := \left\{ w \mid P\left(\frac{d}{dt}\right) w = 0 \right\}, \tag{4.99}$$

$$\mathcal{S} := \left\{ w \mid S\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.100}$$

There exists a behavior $\mathcal{C}$,

$$\mathcal{C} := \left\{ w \mid C\left(\frac{d}{dt}\right) w = 0 \right\},$$

such that $\mathcal{P} \parallel \mathcal{C} = \mathcal{S}$ and the interconnection is weakly compatible if and only if there also exists a $\mathcal{C}'$ such that $\mathcal{P} \parallel \mathcal{C}' = \mathcal{S}$ and the interconnection is regular.

Proof. (if) This is trivial since $\mathcal{P} \parallel \mathcal{C}'$ is also weakly compatible.

(only if) If $\mathcal{P} \parallel \mathcal{C} = \mathcal{S}$ and the interconnection is weakly compatible then necessarily for all $p \in \mathcal{P}$, there exists an $s \in \mathcal{S}$ such that p is weakly directable to s. Equivalently, this means for every $p \in \mathcal{P}$, there exist an $s \in \mathcal{S}$ and a $p' \in \mathcal{P}^{\text{ctr}}$ such that

$$p = s + p'.$$

$\mathcal{P}^{\text{ctr}}$ is the controllable part of $\mathcal{P}$. Therefore we can have the following relations.

$$\mathcal{S} \subset \mathcal{P}, \tag{4.101a}$$

$$\mathcal{S} + \mathcal{P}^{\text{ctr}} = \mathcal{P}. \tag{4.101b}$$

In [BT02], (4.101) is proven to be necessary and sufficient conditions for the existence of a $\mathcal{C}'$ such that $\mathcal{P} \parallel \mathcal{C}' = \mathcal{S}$ and the interconnection is regular. ☐

Again, a similar result for discrete time linear systems can be derived, by replacing the $(\frac{d}{dt})$ operator with σ. Thus, we can conclude even further, that the control problem with weak compatibility constraint can be expressed equivalently as follows.

Problem 4.57. Given the plant $\mathcal{P}$, where

$$\mathcal{P} = \left\{ (w, c) \mid R\left(\frac{d}{dt}\right) w + M\left(\frac{d}{dt}\right) c = 0 \right\}, \tag{4.102}$$

and the specification $\mathcal{S}$, where

$$\mathcal{S} = \left\{ w \mid S\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.103}$$

Find a controller $\mathcal{C}$ of the form

$$\mathcal{C} = \left\{ c \mid C\left(\frac{d}{dt}\right) c = 0 \right\} \tag{4.104}$$

such that
(i) $\pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \mathcal{C}) = \mathcal{S}$,
(ii) $\pi_c \mathcal{P} \parallel \mathcal{C}$ is regular.

The problem of finding necessary and sufficient conditions for solving this problem is known as the 'regular implementability' problem. It has been proven in [BT02, Bel03] that such conditions exist.

Theorem 4.58. (cf. Theorem 4 in [BT02]) Let the plant be given as

$$\mathcal{P} = \left\{ (w, c) \mid R\left(\frac{d}{dt}\right) w + M\left(\frac{d}{dt}\right) c = 0 \right\}, \tag{4.105}$$

and the specification as

$$\mathcal{S} = \left\{ w \mid S\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.106}$$

There exists a controller $\mathcal{C}$ such that $\pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \mathcal{C}) = \mathcal{S}$ and $\pi_c \mathcal{P} \parallel \mathcal{C}$ is regular if and only if
(i) $\mathcal{N} \subset \mathcal{S} \subset \pi_s \mathcal{P}$ and
(ii) $\mathcal{S} + (\pi_s \mathcal{P})^{\mathrm{ctr}} = \pi_s \mathcal{P}$.
Here the behavior $\mathcal{N}$ is the so called hidden behavior, represented by

$$\mathcal{N} = \left\{ w \mid R\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.107}$$

A specification $\mathcal{S}$ that can be achieved using a regular controller as in Theorem 4.58 is said to be *regularly achievable*. In the literature this term is also called *regularly implementable* [BT02, Bel03]. Combining Lemma 4.54, Lemma 4.56 and Theorem 4.58, we obtain the following result.

Theorem 4.59. Let the plant be given as

$$\mathcal{P} = \left\{ (w, c) \mid R\left(\frac{d}{dt}\right) w + M\left(\frac{d}{dt}\right) c = 0 \right\}, \tag{4.108}$$

and the specification as

$$\mathcal{S} = \left\{ w \mid S\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.109}$$

There exists a controller $\mathcal{C}'$,

$$\mathcal{C}' := \left\{ (w,c) \mid C\left(\frac{d}{dt}\right) c = 0 \right\} \tag{4.110}$$

such that $\pi_s(\mathcal{P} \parallel \mathcal{C}') = \mathcal{S}$ and $\mathcal{P} \parallel \mathcal{C}'$ is weakly compatible if and only if
(i) $\mathcal{N} \subset \mathcal{S} \subset \pi_s\mathcal{P}$ and
(ii) $\mathcal{S}+(\pi_s\mathcal{P})^{\mathrm{ctr}} = \pi_s\mathcal{P}$.
Here the behavior $\mathcal{N}$ is the so called hidden behavior, represented by

$$\mathcal{N} = \left\{ w \mid R\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.111}$$

A discrete time version of this result can be obtained by replacing the $(\frac{d}{dt})$ operator with σ. Thus, Problem 4.55 is solvable if and only if Problem 4.57 is solvable, and Theorem 4.59 (and Theorem 4.58) gives the necessary and sufficient condition for solving them.

4.3 Input-output partition constraint

4.3.1 Constraint formulation

In this section we present another kind of constraint that can arise when we interconnect two systems.

One of the features of the behavioral approach to systems theory is that no *a priori* distinction is made between input and output variables of a system [Wil91, PW98]. This means that given a certain law that describes the system, the system is identified by the collection of its trajectories as is. Therefore, it is not necessary to have any input-output structure when describing the system.

However, when two systems are interconnected, sometimes some input-output structure can emerged naturally as a constraint. Consider the following example.

Example 4.60. Consider a tank filled with water as shown in Figure 4.8. On top of the tank is an inlet from which a variable flow of water can get into the tank. This variable models, for example, rain fall. We denote the water flow from this inlet as $\mathbf{e}$. On the bottom of the tank, there is an opening connected to a pump that can pump water out of/into the tank. We denote the amount of water flow pumped out of the tank as $\mathbf{u}$. The tank is also equipped with a sensor that measures the change of volume of water inside the tank, the measurement of the sensor is denoted as $\mathbf{d}$. The mathematical model of this system can be simply written as

$$d(t) = e(t) - u(t). \tag{4.112}$$

Now consider the following control problem. Given $\mathbf{d}$ and $\mathbf{u}$ as control variables, we want to design a controller such that the level of water is constant, i.e. $e(t) = u(t)$. In other words, we aim at perfect tracking of $\mathbf{e}$ by $\mathbf{u}$. Intuitively, we know

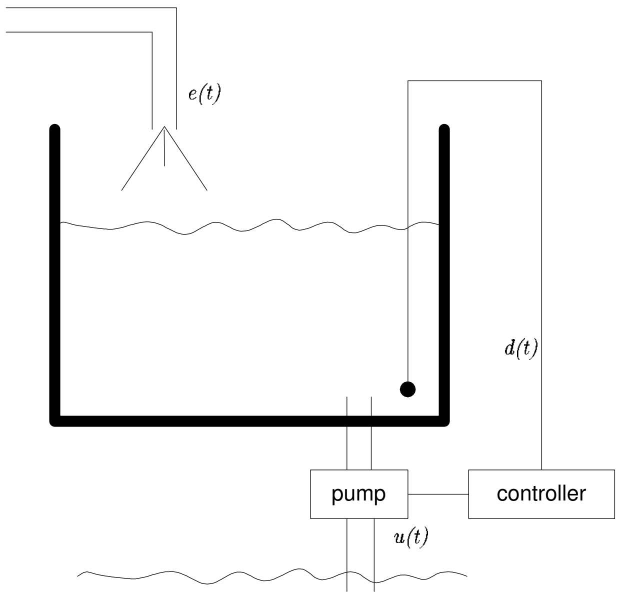


Figure 4.8: The water tank system in Example 4.60.

that such task cannot be accomplished. However, consider the following analysis. First we write the plant behavior in a kernel representation.

$$\mathcal{P} = \{(e,u,d) \mid e(t) - u(t) - d(t) = 0\}. \tag{4.113}$$

We then take a candidate controller $\mathcal{C}$ expressed by

$$\mathcal{C} = \{(u,d) \mid d(t) = 0\}. \tag{4.114}$$

The interconnection $\mathcal{P} \parallel \mathcal{C}$ is represented by the

$$\begin{bmatrix} 1 & -1 & -1 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} e \\ u \\ d \end{bmatrix} = 0. \tag{4.115}$$

Notice that the interconnection exhibits the following features.
(i) The interconnection is a regular feedback interconnection and thus it is compatible.
(ii) The controller can indeed be expressed only in terms of **u** and **d**.
(iii) In the "closed loop"behavior, perfect tracking $e(t) = u(t)$ is attained.

In the example above, the controller satisfies the compatibility constraint and accomplishes the task. However, this is still counter intuitive, and impossible to

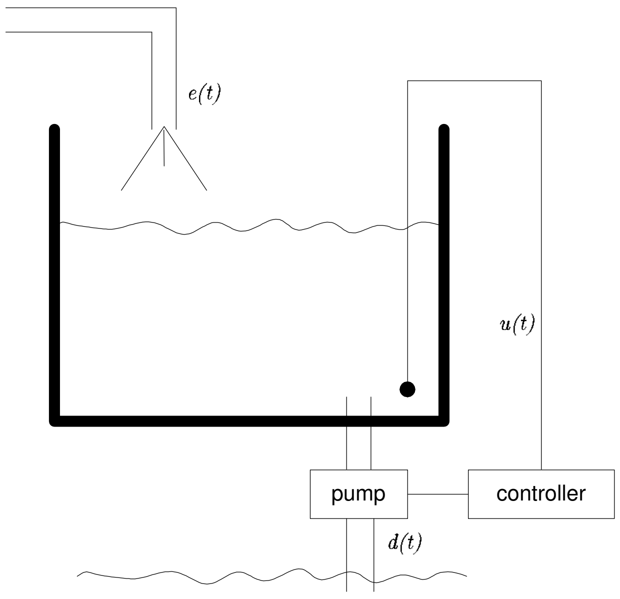


Figure 4.9: The water tank system in Example 4.61.

implement. The variable $\mathbf{d}$ is a measurement coming from a sensor, and yet we use it to control the system. Otherwise stated, we control the system by restricting the reading of a sensor.

Motivated by Example 4.60, we know that *we need to introduce a new kind of constraint*. Before we proceed to do that, consider the following example, which is a continuation of Example 4.60.

Example 4.61. Consider again the water tank system in Example 4.60. Let us swap the name of variables involved in the system as follows. We swap $\mathbf{d}$ and $\mathbf{u}$. The schematic of the system is now shown in Figure 4.9. Notice that the mathematical model of the system is still given by (4.113). Now take the controller $\mathcal{C}$ given by (4.114). Clearly, the features of the interconnection (4.115) are still there. What the controller now does is shut down the pump. This controller does not keep the water level constant. But, that is not the fact that we are interested in. The interesting observation is that now the interconnection does make sense.

These two examples suggest the following facts.

- The new constraint we are going to formulate later cannot be formulated based on the mathematical representation of the systems alone. The situations described in Example 4.60 and Example 4.61 share the same mathematical representation, yet in one situation the constraint is not satisfied, while in the other it is. This is in contrast with the compatibility constraint, where the constraint can actually be derived from the behaviors themselves.

- The new constraint is different from the compatibility constraint. Example 4.60 describes a situation where the compatibility constraint is met, while the new constraint that we are going to formulate is not.

As is indicated by the Example 4.60, the constraint is violated when the plant is restricted through a variable that is inherently an output of the system. That is, the variable is physically dictated to be an output of the system. The information that a variable is an output cannot be deduced from the mathematical description of the system, rather it has to be provided in addition to the description of the plant.

We say that the constraint is met *if the controller accepts the declared output of the plant as its input*. To say it differently, suppose that $\mathbf{y}$ is a (set of) variable(s) that is a part of the control variables. If $\mathbf{y}$ is declared as output because of some physical interpretation of the system, the constraint is met if we can partition the variables of the controller, such that $\mathbf{y}$ belongs to the input part. Input-output partitioning of the variables of a linear system has been introduced in Definition 4.46.

The control problem with input-output partitioning constraint for linear systems is then formally defined as follows.

Definition 4.62. Given a control problem, where the plant is

$$\mathcal{P} = \left\{(w,u,y) \mid R\left(\frac{d}{dt}\right)w + P\left(\frac{d}{dt}\right)u + Q\left(\frac{d}{dt}\right)y = 0\right\}. \tag{4.116}$$

The control variables are $\mathbf{u}$ and $\mathbf{y}$, where $\mathbf{y}$ is the predefined output variables of the plant. A controller $\mathcal{C}$ described as

$$\mathcal{C} = \left\{(u,y) \mid C_1\left(\frac{d}{dt}\right)u + C_2\left(\frac{d}{dt}\right)y = 0\right\}, \tag{4.117}$$

is said to satisfy the input-output constraint if the variables in $\mathcal{C}$ can be partitioned such that $\mathbf{y}$ belongs to the input part.

Lemma 4.63. Given a linear system

$$\mathcal{C} = \left\{(u,y) \mid C_1\left(\frac{d}{dt}\right)u + C_2\left(\frac{d}{dt}\right)y = 0\right\}. \tag{4.118}$$

Without loss of generality we assume that $[C_1\ C_2]$ is full row rank. The following statements are equivalent.
(i) The variables in $\mathcal{C}$ can be partitioned such that $\mathbf{y}$ belongs to the input part
(ii) C_1 is full row rank.
(iii) If $\mathcal{C}$ is the strong solution of the kernel representation, i.e. $\mathcal{C} \in \mathfrak{L}_{\mathrm{c}}^{\mathrm{u+y}}$, for any $y \in \mathfrak{C}^\infty(\mathbb{R},\mathbb{R}^{\mathrm{y}})$ there exists a $u \in \mathfrak{C}^\infty(\mathbb{R},\mathbb{R}^{\mathrm{u}})$ such that $(u,y) \in \mathcal{C}$.

Proof. (ii $\Rightarrow$ i) Suppose that C_1 is full row rank. If C_1 is a square matrix, then we already have an input-output partition with $\mathbf{u}$ as the output and $\mathbf{y}$ as the input. If C_1 is not square, then we can partition it into

$$C_1 = \left[\begin{array}{cc} C_{11} & C_{12} \end{array} \right], \tag{4.119}$$

possibly after rearranging the columns, such that C_{11} is a square matrix with full row rank. We can also partition $\mathbf{u}$ accordingly into $\mathbf{u}_1$ and $\mathbf{u}_2$. Now we have an input-output partition with $\mathbf{u}_1$ as the output and $\mathbf{u}_2$ and $\mathbf{y}$ as the input.

(i $\Rightarrow$ iii) Suppose that the variables in $\mathcal{C}$ can be partitioned such that $\mathbf{y}$ belongs to the input partition. This means we can partition $\mathbf{u}$ into $\mathbf{u}_1$ and $\mathbf{u}_2$, such that we have $\mathbf{u}_1$ as the output and $\mathbf{u}_2$ and $\mathbf{y}$ as the input. So we can partition C accordingly such that (4.119) holds. Following the elimination procedure in Subsection 3.2.2, we can eliminate $\mathbf{u}_1$ and find that the behavior in terms of $\mathbf{y}$ and $\mathbf{u}_2$ is $\mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^{\mathtt{y}+\mathtt{u}_2})$.

(iii $\Rightarrow$ ii) We shall prove it by contradiction. Suppose that C_1 is not full row rank. The matrix $[C_1\ C_2]$ can be transformed (by premultiplication with a suitable unimodular matrix) into

$$\left[\begin{array}{cc} C_1' & C_{21}' \\ 0 & C_{22}' \end{array} \right],$$

where C_1' and C_{22}' are full row rank. Following the elimination procedure in Subsection 3.2.2, we can eliminate $\mathbf{u}$ and find that the behavior in terms of $\mathbf{y}$ is the kernel of $C_{22}'(\frac{d}{dt})$. Hence, we cannot choose any $y \in \mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^{\mathtt{y}})$ as a trajectory of $\mathbf{y}$. $\square$

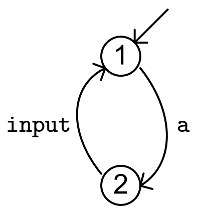
Statement (ii) of Lemma 4.63 gives us a way to test whether a given controller satisfies the input-output partitioning constraint. The results above can be applied to discrete time linear systems by replacing $\frac{d}{dt}$ with σ.

Remark 4.64. A remark on statement (iii) of Lemma 4.63. This statement is equivalent to the fact that the projection of $\mathcal{C}$ to the variable $\mathbf{y}$ (i.e. $\mathbf{u}$ is eliminated) yields $\mathfrak{C}^\infty(\mathbb{R}, \mathbb{R}^{\mathtt{y}})$. Suppose that we assume that $\mathcal{C}$ is in $\bar{\mathfrak{L}}_{\mathrm{c}}^{\mathtt{u}+\mathtt{y}}$ instead of $\mathfrak{L}_{\mathrm{c}}^{\mathtt{u}+\mathtt{y}}$, then this statement is equivalent to the fact that the closure of the projection of $\mathcal{C}$ to the variable $\mathbf{y}$ yields

The equivalent of the input-output partitioning of the variables of linear systems for discrete event systems is the partitioning of events that are used in the synchronization into input and output events [LT89]. An input event $\mathtt{e}$ is always enabled, meaning that in every state of the automaton there is always a transition with label $\mathtt{e}$. This means that the automaton cannot block the occurrence of event $\mathtt{e}$.

If we have an automaton as the plant of the control problem, and some events in a set E are specified as output. This means that we need to find a controller that accepts E as input. Equivalently, the controller cannot block the occurrences of the events in E. In the terminology of supervisory control of discrete event systems, these events are called uncontrollable [CL99].

The fact that a variable is an input to a linear system and the fact that a set of events is input to an automaton have something in common. We have seen in Lemma 4.63 that the fact that $\mathbf{u}$ is an input to a linear system means that we can choose any trajectory in the underlying function space to be the trajectory of $\mathbf{u}$. Since the fact that a set of events E is input to an automaton A means that A can execute any event in E from any state, this implies that the projection of $L(A)$, the language generated by A, to the set of events E yields E^*. Hence we can take any string over the alphabet E. However, the converse is not true. The fact that the projection of $L(A)$ to the set of events E yields E^* is only equivalent to the fact that the automaton can execute the events in E as many times as possible with any order. This does not imply that the the automaton A can execute any event in E from any state. The following automaton is a counterexample.



In this automaton, the projection of the generated language to {`input`} is `input`*, however from the initial state the automaton cannot execute `input`.

4.3.2 Achievability with compatibility and input-output partition constraint

We now return to linear systems and we shall formulate necessary and sufficient condition for solving a control problem with both compatibility constraint and input-output partitioning constraint. We shall work with the continuous time case. However, the treatment for the discrete time case follows immediately by replacing the differential operator $\frac{d}{dt}$ with σ. Let us first formulate the problem.

Problem 4.65. Given a control problem, where the plant is

$$\mathcal{P} = \left\{ (w,u,y) \mid R\left(\frac{d}{dt}\right) w + P\left(\frac{d}{dt}\right) u + Q\left(\frac{d}{dt}\right) y = 0 \right\}. \tag{4.120}$$

The control variables are $\mathbf{u}$ and $\mathbf{y}$, where $\mathbf{y}$ is the predefined output variables of the plant. The to-be-controlled variable is $\mathbf{w}$. The desired specification is given as

$$\mathcal{S} = \left\{ w \mid S\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.121}$$

Find a controller $\mathcal{C}$ in the form of

$$\mathcal{C} = \left\{ (u,y) \mid C_1\left(\frac{d}{dt}\right) u + C_2\left(\frac{d}{dt}\right) y = 0 \right\}, \tag{4.122}$$

such that
(i) $\pi_s \pi_c^{-1}(\pi_c \mathcal{P} \parallel \mathcal{C}) = \mathcal{S}$,
(ii) the interconnection $\pi_c \mathcal{P} \parallel \mathcal{C}$ is regular,
(iii) the controller $\mathcal{C}$ satisfies the input-output partitioning constraint as in Definition 4.62.

As usual, the symbol π_s and π_c denote the projection of the plant behavior by eliminating the control variables and the to-be-controlled variables respectively. Notice that here we require that the interconnection $\pi_c \mathcal{P} \parallel \mathcal{C}$ is regular instead of weakly compatible. This is because regular interconnections have a nicer algebraic characterization. Moreover, regular interconnections are always weakly compatible.

We shall now devise an algorithm to solve the Problem 4.65. The algorithm will also be equipped with a test that can determine whether or not the problem is solvable. The first step of the algorithm is to check whether $\mathcal{S}$ is regularly achievable. This can be done using the conditions given in Theorem 4.58. Obviously, if $\mathcal{S}$ is not regularly achievable, the problem doesn't have a solution. Let us now assume that $\mathcal{S}$ is regularly achievable and proceed.

Notation 4.66. Consider Problem 4.65. Assume that $\mathcal{S}$ is regularly achievable. We denote the class of regular controllers that achieves $\mathcal{S}$ as $\mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$.

Assuming that $\mathcal{S}$ is regularly achievable, it is straightforward to see that Problem 4.65 is equivalent to the following problem.

Problem 4.67. Given a control problem, where the plant is

$$\mathcal{P} = \left\{ (w,u,y) \mid R\left(\frac{d}{dt}\right) w + P\left(\frac{d}{dt}\right) u + Q\left(\frac{d}{dt}\right) y = 0 \right\}. \tag{4.123}$$

The control variables are $\mathbf{u}$ and $\mathbf{y}$, where $\mathbf{y}$ is the predefined output variables of the plant. The to-be-controlled variable is $\mathbf{w}$. The desired specification is given as

$$\mathcal{S} = \left\{ w \mid S\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.124}$$

Assume that $\mathcal{S}$ is regularly achievable. Find a controller $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$ in the form of

$$\mathcal{C} = \left\{ (u,y) \mid C_1\left(\frac{d}{dt}\right) u + C_2\left(\frac{d}{dt}\right) y = 0 \right\}, \tag{4.125}$$

such that the controller $\mathcal{C}$ satisfies the input-output partitioning constraint as in Definition 4.62.

By Lemma 4.63 we can conclude that in order to solve Problem 4.67 we need to find a controller $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$ in the form of (4.125) such that C_1 is full row rank. We shall use the following result.

Lemma 4.68. Let X be a subset of $\mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ such that for any $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ there exists a $\mathcal{C}' \in X$ such that $\mathcal{C} \subseteq \mathcal{C}'$. Then there exists a $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ satisfying the input-output partitioning constraint if and only if there exists a $\mathcal{C}' \in X$ that satisfies the constraint.

Proof. (if) Trivial, since $X \subset \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$.

(only if) Suppose that $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ satisfies the constraint. We shall show that any $\mathcal{C}' \in \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ such that $\mathcal{C} \subseteq \mathcal{C}'$ also satisfies the constraint. Let $\mathcal{C}$ be given as the kernel of $\begin{bmatrix} C_1 & C_2 \end{bmatrix}$ as in (4.125). We know that C_1 is full row rank. Since $\mathcal{C} \subseteq \mathcal{C}'$, there must be a full row rank matrix F such that $\mathcal{C}'$ is the kernel of $\begin{bmatrix} FC_1 & FC_2 \end{bmatrix}$. We also know that FC_1 is full row rank. Therefore $\mathcal{C}'$ also satisfies the constraint. □

Lemma 4.68 tells us that if we can construct a subset of $\mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ with the property of X, we do not need to search for the candidate controller in the whole $\mathfrak{C}_{\mathcal{S}}^{\text{reg}}$. Rather, we can restrict our attention in X. Now we are going to construct a subset X of $\mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ with the property as in Lemma 4.68.

Since $\mathcal{S}$ is regularly achievable, the canonical controller achieves it. The canonical controller is not necessarily regular [WBJT03]. Denote the canonical controller as $\mathcal{C}$. Two important properties of the canonical controller that we shall use in this discussion are
(i) $\mathcal{C}_{\text{can}} \subset \pi_c\mathcal{P}$, thus $\mathcal{C}_{\text{can}} \parallel \pi_c\mathcal{P} = \mathcal{C}_{\text{can}}$,
(ii) It is the least restrictive controller in the sense that for any other controller $\mathcal{C}'$ that achieves $\mathcal{S}$,

$$(\mathcal{C}' \parallel \pi_c\mathcal{P}) \subset (\mathcal{C}_{\text{can}} \parallel \pi_c\mathcal{P}) = \mathcal{C}_{\text{can}}. \tag{4.126}$$

Lemma 4.69. Define X as the set of all regular controllers $\mathcal{C}$ satisfying $\mathcal{C} \parallel \pi_c\mathcal{P} = \mathcal{C}_{\text{can}}$. Then
(i) $X \subset \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$,
(ii) For all $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$, there exists a $\mathcal{C}' \in X$ such that $\mathcal{C} \subset \mathcal{C}'$.

Proof. Statement (i) follows as a consequence of the fact that any controller in X achieves the specification $\mathcal{S}$. We shall now prove statement (ii). Take any $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$. By definition of $\mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ we know that
(a) $\mathcal{C}$ is a regular controller.
(b) For all $w \in \mathcal{S}$, there exists a $c \in \mathcal{C}$ such that $(w, c) \in \mathcal{P}$.
(c) For all $c \in \mathcal{C}$, $(w, c) \in \mathcal{P}$ implies $w \in \mathcal{S}$.

We construct $\mathcal{C}' := \mathcal{C} + \mathcal{C}_{\text{can}}$. Clearly $\mathcal{C} \subset \mathcal{C}'$. We have to prove that $\mathcal{C}' \in X$. That is, we have to prove that
(a′) $\mathcal{C}'$ is regular.
(b′) $\mathcal{C}' \parallel \pi_c\mathcal{P} = \mathcal{C}_{\text{can}}$.
The statement (a′) follows from the fact that $\mathcal{C} \subset \mathcal{C}'$ and the regularity of $\mathcal{C}$. To prove (b′), first we show that $\mathcal{C}'$ implements $\mathcal{S}$. From here, (b′) follows from the fact that $\mathcal{C}_{\text{can}} \subset \mathcal{C}'$ and the property of $\mathcal{C}_{\text{can}}$ being the least restrictive controller. Showing that $\mathcal{C}'$ implements $\mathcal{S}$ means showing that
(a″) For all $w \in \mathcal{S}$, there exists a $c' \in \mathcal{C}'$ such that $(w, c') \in \mathcal{P}$.
(b″) For all $c' \in \mathcal{C}'$, $(w, c') \in \mathcal{P}$ implies $w \in \mathcal{S}$.

Statement (a″) follows immediately from (b). To show that (b″) holds, notice that any $c' \in \mathcal{C}'$ can be written as $c + c_{\text{can}}$ with $c \in \mathcal{C}$ and $c_{\text{can}} \in \mathcal{C}_{\text{can}}$. Also notice that for all $c_{\text{can}} \in \mathcal{C}_{\text{can}}$, there exists a $w_{\text{can}} \in \mathcal{S}$ such that $(w_{\text{can}}, c_{\text{can}}) \in \mathcal{P}$. Thus,

$$\begin{aligned}(w, c') \in \mathcal{P} &\Rightarrow (w - w_{\text{can}} + w_{\text{can}}, c + c_{\text{can}}) \in \mathcal{P} \\ &\overset{\text{linearity}}{\Rightarrow} ((w - w_{\text{can}}), c) \in \mathcal{P} \\ &\overset{\text{(c)}}{\Rightarrow} (w - w_{\text{can}}) \in \mathcal{S} \overset{\text{linearity}}{\Rightarrow} w \in \mathcal{S}.\end{aligned}$$

□

We shall now find a parametrization of the kernel representation of the elements of X as defined in Lemma 4.69. For that purpose, we use the following results.

Lemma 4.70. Consider Problem 4.65. Assume that $\mathcal{S}$ is regularly achievable. Denote the canonical controller corresponding to this problem as $\mathcal{C}_{\text{can}}$. There exists a regular controller $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ such that $(\mathcal{C} \parallel \pi_c \mathcal{P}) = \mathcal{C}_{\text{can}}$.

Proof. Denote $\mathcal{P}_c := \pi_c \mathcal{P}$. Since we know that $\mathcal{S}$ is regularly achievable, we can construct a regular controller $\mathcal{C}'$ that achieves it. Denote $\mathcal{K}_c := \mathcal{C}' \parallel \mathcal{P}_c$. Since $\mathcal{C}'$ is regular, we have that

$$\mathcal{K}_c + \mathcal{P}_c^{\text{ctr}} = \mathcal{P}_c, \tag{4.127}$$

where $\mathcal{P}_c^{\text{ctr}}$ is the controllable part of $\mathcal{P}_c$. By the property of the canonical controller, $\mathcal{K}_c \subset \mathcal{C}_{\text{can}}$, thus

$$\mathcal{C}_{\text{can}} + \mathcal{P}_c^{\text{ctr}} = \mathcal{P}_c. \tag{4.128}$$

Equation (4.128) and Theorem 4.58 imply that there exists a regular controller $\mathcal{C}$ such that $(\mathcal{C} \parallel \mathcal{P}_c) = \mathcal{C}_{\text{can}}$. The fact that $\mathcal{C}$ achieves the specification $\mathcal{S}$ (as $\mathcal{C}_{\text{can}}$ does) follows from the fact that

$$(\mathcal{C} \parallel \mathcal{P}_c) = (\mathcal{C}_{\text{can}} \parallel \mathcal{P}_c). \tag{4.129}$$

□

We shall use Lemma 4.70 in constructing a parametrization of the kernel representation of the elements of X as defined in Lemma 4.69. However, in order to do that we need the two lemmas below.

Lemma 4.71. Let a plant $\mathcal{P}$ be given as the kernel of a full row rank $R(\frac{d}{dt})$ and a regular controller $\mathcal{C}$ be given as the kernel of a full row rank $C(\frac{d}{dt})$. Denote the full interconnection

$$\mathcal{K} := \mathcal{P} \parallel \mathcal{C}.$$

Let $\mathcal{C}'$ be another regular controller such that $\mathcal{P} \parallel \mathcal{C}' = \mathcal{K}$. A minimal kernel representation of $\mathcal{C}'$ has exactly as many rows as $C(\frac{d}{dt})$.

Proof. The number of rows in the minimal kernel representation of a behavior equals the number of its output variables. Thus, this lemma is a direct consequence of the definition of regularity (Definition 4.47). □

Lemma 4.72. Let a plant $\mathcal{P}$ be given as the kernel of a full row rank $R(\frac{d}{dt})$ and a regular controller $\mathcal{C}$ be given as the kernel of a full row rank $C(\frac{d}{dt})$. Denote the full interconnection

$$\mathcal{K} := \mathcal{P} \parallel \mathcal{C}.$$

Let $\mathfrak{C}_{\mathcal{K}}$ denote the set of all controllers (not necessarily regular ones) that
(i) have at most as many rows in the minimal kernel representation as $\mathcal{C}$ and
(ii) also implement $\mathcal{K}$ when interconnected with $\mathcal{P}$.
A controller $\mathcal{C}' \in \mathfrak{C}_{\mathcal{K}}$ if and only if its kernel representation can be written as $VR+C$ for some matrix V. Moreover, every controller in $\mathcal{C}' \in \mathfrak{C}_{\mathcal{K}}$ has the following properties.
(a) $\mathcal{C}'$ is regular.
(b) Its minimal kernel representation has exactly as many rows as that of $\mathcal{C}$.

Proof. (if) Suppose that a controller $\mathcal{C}'$ has $(VR+C)$ as its kernel representation, then $\mathcal{P} \parallel \mathcal{C}'$ is given by the kernel of

$$\begin{bmatrix} R \\ VR+C \end{bmatrix} = \begin{bmatrix} I & 0 \\ V & I \end{bmatrix} \begin{bmatrix} R \\ C \end{bmatrix}. \tag{4.130}$$

This shows that $\mathcal{P} \parallel \mathcal{C}' = \mathcal{K}$. Moreover, since $\mathcal{C}$ is a regular controller, it follows that $(VR+C)$ is a minimal kernel representation of $\mathcal{C}'$. Thus, properties (a) and (b) are verified.

(only if) Suppose that a controller $\mathcal{C}'$ satisfies (i) and (ii) above. This controller can be written as the kernel of a matrix (not necessarily minimal) $C'(\frac{d}{dt})$ with as many rows as $C(\frac{d}{dt})$. We know that there is a unimodular matrix U such that

$$U \begin{bmatrix} R \\ C \end{bmatrix} = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix} \begin{bmatrix} R \\ C \end{bmatrix} = \begin{bmatrix} R \\ C' \end{bmatrix}. \tag{4.131}$$

We shall prove that we can assume U to be of the form

$$U = \begin{bmatrix} I & 0 \\ V & I \end{bmatrix}. \tag{4.132}$$

First, we find a unimodular matrix W such that

$$RW = \begin{bmatrix} D & 0 \end{bmatrix}, \tag{4.133}$$

where D is a square nonsingular matrix. We then use the following notation

$$\begin{bmatrix} R \\ C \end{bmatrix} W =: \begin{bmatrix} D & 0 \\ C_1 & C_2 \end{bmatrix}, \tag{4.134}$$

$$\begin{bmatrix} R \\ C' \end{bmatrix} W =: \begin{bmatrix} D & 0 \\ C_1' & C_2' \end{bmatrix}. \tag{4.135}$$

It follows that (4.131) can be rewritten as

$$U \begin{bmatrix} D & 0 \\ C_1 & C_2 \end{bmatrix} W^{-1} = \begin{bmatrix} D & 0 \\ C_1' & C_2' \end{bmatrix} W^{-1}, \tag{4.136}$$

and since W is unimodular,

$$U\begin{bmatrix} D & 0 \\ C_1 & C_2 \end{bmatrix} = \begin{bmatrix} U_{11} & U_{12} \\ U_{21} & U_{22} \end{bmatrix}\begin{bmatrix} D & 0 \\ C_1 & C_2 \end{bmatrix} = \begin{bmatrix} D & 0 \\ C_1' & C_2' \end{bmatrix}. \tag{4.137}$$

Consequently, we have the following equations

$$\begin{aligned} U_{11}D + U_{12}C_1 &= D, && (4.138a)\\ U_{12}C_2 &= 0, && (4.138b)\\ U_{21}D + U_{22}C_1 &= C_1', && (4.138c)\\ U_{22}C_2 &= C_2'. && (4.138d) \end{aligned}$$

Since the controller $\mathcal{C}$ is regular, C_2 must be full row rank. Now, (4.138b) implies that U_{12} is a left annihilator of C_2. Consequently

$$U_{12} = 0. \tag{4.139}$$

Substituting this to (4.138a) yields

$$U_{11} = I. \tag{4.140}$$

Since U is unimodular, this implies that U_{22} is unimodular. Thus, we can conclude that

$$U = \begin{bmatrix} I & 0 \\ U_{21} & U_{22} \end{bmatrix}, \tag{4.141}$$

with U_{22} unimodular. Furthermore, $C'' := U_{22}C'$ is also a kernel representation of $\mathcal{C}'$ so we can assume U_{22} to be the identity matrix without any loss of generality. □

We take the following steps to construct a parametrization of the kernel representation of the elements of X as defined in Lemma 4.69.

Step 1. Construct a regular controller $\mathcal{C}$ such that $(\mathcal{C} \parallel \pi_c\mathcal{P}) = \mathcal{C}_{\text{can}}$. Lemma 4.70 guarantees that this can be done. The controller $\mathcal{C}$ and $\pi_c\mathcal{P}$ can be represented in the form of

$$\mathcal{C} = \left\{(u,y) \mid C_1\left(\frac{d}{dt}\right)u + C_2\left(\frac{d}{dt}\right)y = 0\right\}, \tag{4.142}$$

$$\pi_c\mathcal{P} = \left\{(u,y) \mid M_1\left(\frac{d}{dt}\right)u + M_2\left(\frac{d}{dt}\right)y = 0\right\}. \tag{4.143}$$

Step 2. We apply Lemma 4.72 with $\mathcal{C}_{\text{can}}$ and $\pi_c\mathcal{P}$ playing the role of $\mathcal{K}$ and $\mathcal{P}$ respectively. Because of Lemma 4.71, we know that a controller $\mathcal{C}'$ represented by

$$\mathcal{C}' = \left\{(u,y) \mid C_1'\left(\frac{d}{dt}\right)u + C_2'\left(\frac{d}{dt}\right)y = 0\right\} \tag{4.144}$$

is an element of X as defined in Lemma 4.69 if and only if there is a polynomial matrix V such that

$$C_1' = C_1 + VM_1, \tag{4.145a}$$

$$C_2' = C_2 + VM_2. \tag{4.145b}$$

With these two steps we obtain (4.145) as the parametrization. Now, the final step to solve Problem 4.67 is to find a polynomial matrix V such that C_1' in (4.145a) is full row rank. The necessary and sufficient condition for the existence of such a matrix V is given in the following lemma.

Lemma 4.73. Given polynomial matrices $C \in \mathbb{R}^{\mathtt{c}\times\mathtt{q}}[\xi]$ and $M \in \mathbb{R}^{\mathtt{m}\times\mathtt{q}}[\xi]$. There exists a polynomial matrix $V \in \mathbb{R}^{\mathtt{c}\times\mathtt{m}}[\xi]$ such that $C + VM$ is full row rank if and only if

$$\operatorname{rank}\begin{bmatrix} M \\ C \end{bmatrix} \geq \mathtt{c}. \tag{4.146}$$

Proof. (only if) Consider the following relation

$$\begin{bmatrix} I & 0 \\ V & I \end{bmatrix}\begin{bmatrix} M \\ C \end{bmatrix} = \begin{bmatrix} M \\ C+VM \end{bmatrix}. \tag{4.147}$$

Suppose that $C + VM$ is full row rank. This means it has a rank of $\mathtt{c}$. From (4.147) we know that

$$\operatorname{rank}\begin{bmatrix} M \\ C \end{bmatrix} = \operatorname{rank}\begin{bmatrix} M \\ C+VM \end{bmatrix} \geq \mathtt{c}. \tag{4.148}$$

(if) Assume that (4.146) holds. If C or M is zero, we can obviously choose a V such that $C+VM$ full row rank. We exclude this trivial case and suppose that both M and C are nonzero. Since the rank of a matrix is not affected by multiplication with unimodular matrices, we can assume without any loss of generality that M has the form of

$$M = \begin{bmatrix} M_1 & 0 \\ 0 & 0 \end{bmatrix}, \tag{4.149}$$

where M_1 is a diagonal matrix with nonzero determinant. Furthermore, with some appropriate multiplication with unimodular matrix, we can transform C to the following form.

$$C = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & 0 \\ 0 & 0 \end{bmatrix}, \tag{4.150}$$

where C_{12} and C_{21} are full row rank. Denote the rank of M_1, C_{12}, and C_{21} as $\mathtt{m}'$, $\mathtt{c}'$, and $\mathtt{c}''$ respectively. We have the following relation

$$\operatorname{rank}\begin{bmatrix} M \\ C \end{bmatrix} = \operatorname{rank} C_{12} + \operatorname{rank}\begin{bmatrix} M_1 \\ C_{21} \end{bmatrix}, \tag{4.151}$$

$$= \mathtt{c}' + \mathtt{m}', \tag{4.152}$$

$$\geq \mathtt{c}. \tag{4.153}$$

Thus

$$\mathtt{m}' \geq \mathtt{c} - \mathtt{c}'. \tag{4.154}$$

We can partition V accordingly to form

$$V = \begin{bmatrix} V_{11} & V_{12} \\ V_{21} & V_{22} \\ V_{31} & V_{32} \end{bmatrix}. \tag{4.155}$$

We choose the value of V to be

$$V = \begin{bmatrix} 0 & 0 \\ 0 & 0 \\ V_{31} & 0 \end{bmatrix}, \tag{4.156}$$

where V_{31} is to be chosen later. Therefore

$$C + VM = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & 0 \\ V_{31}M_1 & 0 \end{bmatrix}. \tag{4.157}$$

Our goal is to make $\begin{bmatrix} C_{21} \\ V_{31}M_1 \end{bmatrix}$ a full row rank matrix. Since C_{21} is full row rank and has the rank of $\mathtt{c}''$, we can find $\mathtt{c}''$ columns of C_{21} that form a square matrix with nonzero determinant. Denote this selection as N. Obviously $N \subset \{1, 2, \cdots, \mathtt{m}'\}$. We construct $V_{31} \in \mathbb{R}^{(\mathtt{c}-\mathtt{c}'-\mathtt{c}'')\times\mathtt{m}'}[\xi]$ such that the entries on the $i-$th column of V_{31} are zero if $i \in N$. The remaining $(\mathtt{m}' - \mathtt{c}'')$ columns of V_{31} form a $(\mathtt{c} - \mathtt{c}' - \mathtt{c}'')$ by $(\mathtt{m}' - \mathtt{c}'')$ matrix. From (4.154) we know that it is a wide matrix. We choose the values of the entries of these columns such that this wide matrix is full row rank. It follows that $\begin{bmatrix} C_{21} \\ V_{31}M_1 \end{bmatrix}$ is a full row rank matrix and hence $C + VM$ is full row rank. □

To conclude, the following is the algorithm to solve Problem 4.65.

Algorithm 4.74. The following steps provide a solution to Problem 4.65 if and only if it is solvable.
1. Verify if the specification $\mathcal{S}$ is regularly achievable. If it is, go to step 2, otherwise Problem 4.65 is not solvable.
2. Construct the canonical controller for this problem, denote it as $\mathcal{C}_{\text{can}}$.
3. Construct a regular controller $\mathcal{C}$ such that $(\mathcal{C} \parallel \pi_c\mathcal{P}) = \mathcal{C}_{\text{can}}$. Lemma 4.70 guarantees that this can be done. The controller $\mathcal{C}$ and $\pi_c\mathcal{P}$ can be represented in the form of

$$\mathcal{C} = \left\{(u, y) \mid C_1\left(\frac{d}{dt}\right)u + C_2\left(\frac{d}{dt}\right)y = 0\right\}, \tag{4.158}$$

$$\pi_c\mathcal{P} = \left\{(u, y) \mid M_1\left(\frac{d}{dt}\right)u + M_2\left(\frac{d}{dt}\right)y = 0\right\}. \tag{4.159}$$

4. Verify if

$$\operatorname{rank}\begin{bmatrix} M_1 \\ C_1 \end{bmatrix} \geq \mathtt{p}(\mathcal{C}), \tag{4.160}$$

where $\mathtt{p}(\mathcal{C})$ denotes the number of output variables of $\mathcal{C}$. If (4.160)is satisfied, go to step 5, otherwise Problem 4.65 is not solvable.
5. Compute a V such that $C_1 + VM_1$ is full row rank. The existence of such V is guaranteed by Lemma 4.73. A controller that solves Problem 4.65 is given as

$$\mathcal{C}' = \left\{(u,y) \mid \begin{bmatrix} C_1 + VM_1 & C_2 + VM_2 \end{bmatrix}\left(\frac{d}{dt}\right)\begin{bmatrix} u \\ y \end{bmatrix} = 0\right\}. \tag{4.161}$$

4.4 Control problem with minimal interaction

In this section we shall discuss a control problem related to Proposition 4.17. The proposition states that given a control problem as in Problem 4.2, if the problem is solvable, then it is also solvable if the projection π_c is replaced with a bigger projection ϕ_c. By larger we mean $\phi_c \succeq \pi_c$.

This result induces a control problem that can be formulated as follows.

Control with minimal interaction Given a control problem as in Problem 4.2. Assume that the problem is solvable.
(i) Is it possible to replace the control projection π_c with ϕ_c such that $\phi_c \preceq \pi_c$ and retain the achievability of the specification?
(ii) Is there a unique minimal control projection by which the specification is achievable?

As we have discussed earlier, the role of π_c is to define the extent, to which the controller can interact with the plant. Minimizing π_c while retaining achievability of the specification can mean:
(i) Using as few control variables as possible in linear systems.
(ii) Using as few events as possible in the synchronization between the controller and the plant, in the control of discrete event systems.
(iii) A combination of both points above, in hybrid systems.

However, in this section we shall restrict our attention to the case of linear systems, as it turns out that the machinery that we build in the previous section can also be used to analyze this problem. In the discussion we shall treat continuous time systems. However, as usual, the results can be extended to discrete time linear systems by replacing the $\frac{d}{dt}$ operator with σ. We use the following definition to formalize the problem statement.

Definition 4.75. Let a behavior $\mathfrak{B}$ be given by the kernel representation

$$R_1\left(\frac{d}{dt}\right)w_1 + R_2\left(\frac{d}{dt}\right)w_2 = 0. \tag{4.162}$$

If R_1 is the zero matrix, then the variables in w_1 are said to be **irrelevant** to $\mathfrak{B}$.

The problem of control with minimal interaction for linear systems, together with the compatibility constraint can be cast as follows.

Problem 4.76. Given the plant $\mathcal{P}$, where

$$\mathcal{P} = \left\{ (w, c) \mid R\left(\frac{d}{dt}\right) w + M\left(\frac{d}{dt}\right) c = 0 \right\}, \tag{4.163}$$

and the specification $\mathcal{S}$, where

$$\mathcal{S} = \left\{ w \mid S\left(\frac{d}{dt}\right) w = 0 \right\}. \tag{4.164}$$

Suppose that $\mathcal{S}$ is regularly achievable. Find a regular controller $\mathcal{C}$ of the form

$$\mathcal{C} = \left\{ c \mid C\left(\frac{d}{dt}\right) c = 0 \right\} \tag{4.165}$$

that achieves $\mathcal{S}$ and has as many irrelevant variables as possible.

Since the number of variables is finite, clearly there is a maximal number of irrelevant variables that can be attained. However, generally there is no unique selection of variables to make up this maximal number.

To solve the problem we need to find a controller $\mathcal{C}$ that has as many irrelevant variables as possible. This controller must be an element of $\mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$, which is the class of regular controllers that achieves $\mathcal{S}$. We are going to use a result similar to Lemma 4.68 for this purpose.

Lemma 4.77. Let X be a subset of $\mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$ such that for any $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$ there exists a $\mathcal{C}' \in X$ such that $\mathcal{C} \subseteq \mathcal{C}'$. If $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$ is a controller that has the maximal number of irrelevant variables, then there is a $\mathcal{C}' \in X$ that has at least as many irrelevant variables as $\mathcal{C}$.

Proof. If a variable is irrelevant in $\mathcal{C}$, it is also irrelevant in any $\mathcal{C}' \supseteq \mathcal{C}$. Therefore if a $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$ has n irrelevant variables, there is a $\mathcal{C}' \in X$ that has at least n irrelevant variables. □

Lemma 4.77 tells us that if we can construct a subset X of $\mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$ with the property described in the premise statement, then it is sufficient to search for the controller in X instead of $\mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$. However, from the previous section, we know that such subset X exists (see Lemma 4.69) and is well parametrized (see Lemma 4.71 and 4.72). Therefore, the following algorithm can be used to solve the problem.

Algorithm 4.78. The following steps provide a solution to Problem 4.76.
1. Construct the canonical controller for this problem. Denote it as $\mathcal{C}_{\mathrm{can}}$.
2. Construct a regular controller $\mathcal{C}$ such that $(\mathcal{C} \parallel \pi_c\mathcal{P}) = \mathcal{C}_{\mathrm{can}}$. Lemma 4.70

guarantees that this can be done. The controller $\mathcal{C}$ and $\pi_c\mathcal{P}$ can be represented in the form of

$$\mathcal{C} = \left\{ c \mid C\left(\frac{d}{dt}\right) c = 0 \right\}, \tag{4.166}$$

$$\pi_c\mathcal{P} = \left\{ c \mid M'\left(\frac{d}{dt}\right) c = 0 \right\}. \tag{4.167}$$

3. Find a matrix V such that $C + VM'$ has as many zero columns as possible.

With this algorithm, we have transformed Problem 4.76 into an algebraic problem given in the third step of the algorithm. The solution to this algebraic problem has a combinatorial aspect in it, as we generally need to search for the answer by trying all possible subsets of the columns.

However, for a special case, where M' is in the Smith form, we can get a relatively straightforward answer. In that case, we generally can write M' as

$$M' = \begin{bmatrix} I & 0 & 0 \\ 0 & D & 0 \end{bmatrix}, \tag{4.168}$$

where D is a diagonal matrix whose diagonal entries are polynomials with degree at least one. If $\pi_c\mathcal{P}$ is controllable then the size of D is actually zero. Similarly, C is partitioned as

$$C = \begin{bmatrix} C_1 & C_2 & C_3 \end{bmatrix}. \tag{4.169}$$

The problem becomes to find $V = [V_1\ V_2]$ such that

$$\begin{aligned} &\begin{bmatrix} C_1 & C_2 & C_3 \end{bmatrix} + \begin{bmatrix} V_1 & V_2 \end{bmatrix} \begin{bmatrix} I & 0 & 0 \\ 0 & D & 0 \end{bmatrix} \\ &= \begin{bmatrix} C_1 + V_1 & C_2 + V_2 D & C_3 \end{bmatrix} \end{aligned}$$

has as many zero columns as possible. The solution is obvious. The columns in the left most partition can be nullified by choosing $V_1 = -C_1$. The columns in the right most partition cannot be nullified, except for those that happen to be zero columns. The i-th column of the middle partition can be nullified if and only if it is a multiple of the polynomial D_{ii}. Here D_{ii} denotes the i-th entry of the diagonal of D.

Although we cannot present a complete procedure to compute the controller with minimal interaction in the general case, we can present an upper bound for the number of irrelevant variables in the controller.

Lemma 4.79. A controller $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ can have at most $\mathtt{c} - \mathtt{p}(\mathcal{C})$ irrelevant variables. Here $\mathtt{c}$ denotes the number of all control variables (the cardinality of $\mathbf{c}$) and $\mathtt{p}(\mathcal{C})$ denotes the number of output variables in $\mathcal{C}$.

Proof. From Lemma 4.71 we know that all elements of $\mathfrak{C}_{\mathcal{S}}^{\text{reg}}$ have the same number of output, i.e., $\mathtt{p}(\mathcal{C})$. This is the number of rows in a minimal kernel representation of any controller in $\mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$. It is easily seen that the number of columns is $\mathtt{c}$. If a controller $\mathcal{C} \in \mathfrak{C}_{\mathcal{S}}^{\mathrm{reg}}$ has more than $\mathtt{c} - \mathtt{p}(\mathcal{C})$ irrelevant variables, then the nonzero entries of its kernel representation form a tall matrix[3], and thus cannot be minimal. This contradicts Lemma 4.71. □

We shall see later in this section that the upper bound given by Lemma 4.79 is tight. There are some situations where this upper bound is actually reached.

An alternative problem of interest can be formulated as follows. Recall that in the Problem 4.76, our goal is to use as few control variables as possible. These control variables are a part of the initial control variables. Now, suppose that instead of picking a part of the original control variables as the new control variables directly, we first allow for an isomorphic transformation of variables to take place. This means, we construct a new set of control variables $\mathbf{c}_{\mathrm{new}}$ from the old ones $\mathbf{c}$ by

$$c_{\mathrm{new}} = T\left(\frac{d}{dt}\right)c. \tag{4.170}$$

The matrix T is a unimodular matrix to be designed. Our goal is to design the transformation T such that we can use as few variables in the new control variables $\mathbf{c}_{\mathrm{new}}$ as possible. Of course, with this new selection of control variables, we have to maintain regular implementability of the specification.

This alternative description of the problem also can be interpreted as control with minimal information. This is because $\mathbf{c}_{\mathrm{new}}$ and $\mathbf{c}$ contain the same 'amount' of information, as they are related through an isomorphic transformation. It turns out that this problem has a simple solution.

It can be verified that with this new problem formulation, the problem changes from 'finding a V such that $C + VM'$ has as many zero columns as possible' to 'finding a V and a unimodular T such that $(C + VM')T^{-1}$ has as many zero columns as possible'. Obviously the new problem formulation is equivalent to 'finding a V such that $C + VM'$ has as small column rank as possible'. From Lemma 4.71 we know that $C + VM'$ always has the same row rank as C, thus it always has the same column rank as C. We can then take any matrix to be V. For simplicity, we take $V = 0$. We then compute a unimodular matrix U such that

$$CU = \left[\begin{array}{cc} \tilde{C} & 0 \end{array}\right], \tag{4.171}$$

where $\tilde{C}$ has full column rank. The transformation is then

$$c_{\mathrm{new}} = U^{-1}\left(\frac{d}{dt}\right)c, \tag{4.172}$$

and the new control variables that are relevant to the controller are the first $\mathtt{p}(\mathcal{C})$ components of $\mathbf{c}_{\mathrm{new}}$. Here the symbol $\mathtt{p}(\mathcal{C})$ indicates the number of output variables of $\mathcal{C}$. Therefore, if we are allowed to transform the control variables, we can

[3]A tall matrix is a matrix, in which there are more rows than there are columns.

obtain exactly $\mathtt{c} - \mathtt{p}(\mathcal{C})$ irrelevant variables, which is the upper bound stipulated by Lemma 4.79.

Now we apply the result in this section in a numerical example. Consider the control problem as in Problem 4.65, where

$$R(\xi) = \begin{bmatrix} \xi & -1 & 0 & 0 \\ 0 & \xi & -1 & 0 \\ 0 & 0 & \xi & -1 \\ 1 & 1 & 1 & \xi+1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, M(\xi) = \begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & -1 & 0 & 0 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 0 & -1 \end{bmatrix}, \tag{4.173}$$

$$S(\xi) = \begin{bmatrix} \xi+1 & -1 & 0 & 0 \\ 1 & 1 & 1 & \xi+2 \\ 0 & \xi & -1 & 0 \\ 0 & 0 & \xi & -1 \end{bmatrix}. \tag{4.174}$$

We can compute that the projected plant behavior $\pi_c \mathcal{P}$ is given as the kernel of $M'(\frac{d}{dt})$, where

$$M'(\xi) = \begin{bmatrix} \xi^2 & 0 & -\xi^3 & 1 \\ \xi^3+\xi^2+\xi+1 & 1 & -\xi^4-\xi^3-\xi^2-\xi-1 & 0 \end{bmatrix}. \tag{4.175}$$

We can also verify that the specification $\mathcal{S}$ is regularly implementable. In fact, it is implemented by a regular controller $\mathcal{C}'$, which is the kernel of $C'(\frac{d}{dt})$, where

$$C'(\xi) = \begin{bmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \end{bmatrix}. \tag{4.176}$$

We see that the controller $\mathcal{C}'$ does not have any irrelevant variables.

Recall that the canonical controller $\mathcal{C}_{\mathrm{can}}$ is constructed as the behavior obtained by eliminating $\mathbf{w}$ from the following behavior (see Definition 4.10),

$$\left\{ (w, c) \mid \begin{bmatrix} R & M \\ S & 0 \end{bmatrix} \left(\frac{d}{dt} \right) \begin{bmatrix} w \\ c \end{bmatrix} = 0 \right\}.$$

Computing this, we obtain a kernel representation $C_{\mathrm{can}}(\frac{d}{dt})$ of $\mathcal{C}_{\mathrm{can}}$, where

$$C_{\mathrm{can}}(\xi) = \begin{bmatrix} \xi^2 & 0 & -\xi^3 & 1 \\ 1 & 0 & 1 & 0 \\ \xi^4+2\xi^3+2\xi^2+2\xi+2 & 1 & 0 & 0 \\ \xi^4+3\xi^3+3\xi^2+2\xi+2 & 0 & 0 & 0 \end{bmatrix}. \tag{4.177}$$

We see the $\mathcal{C}_{\mathrm{can}}$ is not a regular controller. Nevertheless, we can construct a regular controller $\mathcal{C} \in X$ as the kernel of $C(\frac{d}{dt})$, where

$$C(\xi) = \begin{bmatrix} 1 & 0 & 1 & 0 \\ \xi^4+3\xi^3+3\xi^2+2\xi+2 & 0 & 0 & 0 \end{bmatrix}. \tag{4.178}$$

We can see that $C(\xi)$ already has two zero columns. This means the second and fourth control variables are irrelevant in $\mathcal{C}$. Following Lemma 4.79, we know that there cannot be more than two irrelevant variables in a regular controller that implements $\mathcal{S}$. Thus $\mathcal{C}$ is a controller with minimal interaction.

As a final remark, notice the fact that in this particular example, the controller with minimal interaction $\mathcal{C}$ has a McMillan degree of 4, while another regular controller $\mathcal{C}'$ has a McMillan degree of 0. This fact shows that while the controller with minimal interaction uses fewer control variables to interact with the controller, it can be more complex than a controller that uses more control variables.

4.5 Summary

In this chapter we discuss about control as interconnection in the behavioral framework. We start with formulating some control problems in a general setting. Essentially, control problem in the behavioral framework can be formulated as follows.

Control problem Given a system called the *plant*, . The problem is to find a behavior (called the *controller*), which when interconnected with the plant behavior in a certain manner yields some desired properties, usually given in terms of another behavior (called the *specification*).

The simplest problem is the full interconnection control problem, where the interconnection between the plant and the controller is defined as a full interconnection.

There are also other problems, namely when the plant and the controller are interconnected with partial interconnection. We give some necessary and sufficient conditions for the existence of the solution to these problems. We also put forward the idea of canonical controllers. Canonical controllers are mathematical constructs that solve the control problem if the problem is solvable at all.

We proceed to define some constraints that may arise when we want to design a controller. The first constraint is the compatibility constraint. To put it simply in a few words, the compatibility constraint dictates that the interconnection between the plant and the controller should affect them causally. If the interconnection is formed at a certain time instant in the time axis, then it should be the case that all possible past trajectories of the systems prior to this time instant can be accommodated by the interconnection. We introduce two notions of compatibility, namely compatibility and weak compatibility. Subsequently we show that these concepts are related to the already known concepts of regular and regular feedback interconnections of linear systems. While conditions for achievability of a specification with a compatible/regular feedback interconnection remains an open problem, we manage to present necessary and sufficient conditions for that with a weakly compatible interconnection.

The second constraint that we discuss is the input-output partitioning constraint. We argue that in modeling some systems, it can happen that some variables are bound to be the output of the system. The exact definition of input and output variables are defined. The analog of the situation where we have input and output variables is when we have input and output events in automata.

For linear systems, we derive a procedure that can compute a controller that respects both the compatibility constraint (in the form of regularity) and the input-output partitioning constraint. The procedure also acts as a necessary and sufficient condition for solvability of the control problem, as it produces a result if and only if the problem is solvable at all.

In the final section of the chapter, we discuss about controller with minimal interaction, particularly for linear systems. For linear systems, controller with minimal interaction is a controller that respects the compatibility constraint (in the form of regularity) and uses as few control variables as possible. We also present a procedure that can compute a controller with minimal interaction.

5

External equivalence of systems

"All animals are equal but some animals are more equal than others." - George Orwell

5.1 External behavior equivalence

In this chapter we shall discuss about external equivalence of behaviors. So far in this book, systems are identified with their behaviors, and therefore equivalence of systems is understood as equality of the behaviors. In this chapter we are going to look at this issue from different points of view.

Recall that a dynamical system Σ is formally expressed as a triple $(\mathbb{T}, \mathbb{W}, \mathfrak{B})$.

Definition 5.1. We factor the signal space $\mathbb{W}$ as

$$\mathbb{W} = \mathbb{V} \times \mathbb{D}. \tag{5.1}$$

The space $\mathbb{V}$ is called the **external signal space** and $\mathbb{D}$ the **internal signal space**.

As a consequence of the distinction between external and internal signal space, every trajectory $w \in \mathfrak{B} \subset \mathbb{W}^{\mathbb{T}}$ can be written as a tuple $(v, d) \in \mathbb{V}^{\mathbb{T}} \times \mathbb{D}^{\mathbb{T}}$. We then introduce the external and internal projections of $\mathfrak{B}$ as follows.

Definition 5.2. The **external projection** π_v of a behavior $\mathfrak{B}$ is defined as

$$\pi_v((v, d)) = v, \forall (v, d) \in \mathfrak{B}. \tag{5.2}$$

The image of $\mathfrak{B}$ under π_v is called the **external behavior** of $\mathfrak{B}$. Similarly the **internal projection** π_d is defined as

$$\pi_d((v, d)) = d, \forall (v, d) \in \mathfrak{B}. \tag{5.3}$$

We define the **internal behavior** of $\mathfrak{B}$ as the image of $\mathfrak{B}$ under π_d.

With the definitions above, we define equivalence of external behavior as follows.

Definition 5.3. Given two behaviors $\mathfrak{B}_1$ and $\mathfrak{B}_2$ of type $(\mathbb{T}, \mathbb{V}\times\mathbb{D}_1)$ and $(\mathbb{T}, \mathbb{V}\times\mathbb{D}_2)$ respectively. Thus the behaviors share the same time axis and the same external signal space. We say that $\mathfrak{B}_1$ and $\mathfrak{B}_2$ are **external behavior equivalent** if their external behaviors are the same. We denote this property by $\mathfrak{B}_1 \approx_{\text{ext}} \mathfrak{B}_2$.

Example 5.4. Consider the linear behaviors $\mathfrak{B}_1$ and $\mathfrak{B}_2$ of $\overrightarrow{\mathfrak{L}}^4_{\mathrm{c}}$ given as follows.

$$\mathfrak{B}_i = \left\{ (x,u,y) \mid \begin{bmatrix} \frac{d}{dt} - A_i & -B_i & 0 \\ I & 0 & -C_i \end{bmatrix} \begin{bmatrix} x \\ u \\ y \end{bmatrix} = 0 \right\}, i = 1, 2, \tag{5.4}$$

where

$$A_1 = \begin{bmatrix} 0 & 1 \\ -1 & -2 \end{bmatrix}; B_1 = \begin{bmatrix} 0 \\ 1 \end{bmatrix}; C_1 = \begin{bmatrix} 1 & 0 \end{bmatrix}, \tag{5.5}$$

$$A_2 = \begin{bmatrix} -2 & -1 \\ 1 & 0 \end{bmatrix}; B_2 = \begin{bmatrix} 1 \\ 0 \end{bmatrix}; C_2 = \begin{bmatrix} 0 & 1 \end{bmatrix}. \tag{5.6}$$

The variable $\mathbf{x}$ is an internal variable, while $\mathbf{u}$ and $\mathbf{y}$ are the external variables. Notice that $\mathfrak{B}_1$ and $\mathfrak{B}_2$ differ in the ordering of the vector $\mathbf{x}$. We therefore factor the signal space $\mathbb{W} = \mathbb{R}^4$ into $\mathbb{V} \times \mathbb{D} = \mathbb{R}^2 \times \mathbb{R}^2$. The external projection of the behaviors correspond to the elimination of the internal variable $\mathbf{x}$. Substituting the values of the matrices and eliminating $\mathbf{x}$, we obtain

$$\pi_v \mathfrak{B}_1 = \pi_v \mathfrak{B}_2 = \left\{ (u,y) \mid \begin{bmatrix} -1 & \left(\frac{d}{dt} + 1\right)^2 \end{bmatrix} \begin{bmatrix} u \\ y \end{bmatrix} = 0 \right\}. \tag{5.7}$$

Thus $\mathfrak{B}_1$ and $\mathfrak{B}_2$ are external behavior equivalent.

The significance of equivalence in the sense of $\approx_{\text{ext}}$ can be explained as follows. Take two behaviors $\mathfrak{B}_1$ and $\mathfrak{B}_2$ of type $(\mathbb{T}, \mathbb{V} \times \mathbb{D}_1)$ and $(\mathbb{T}, \mathbb{V} \times \mathbb{D}_2)$ respectively. Suppose that the external projection of each behavior specifies how the behavior can be interconnected with other behaviors in each environment. By environment we mean any system(s) that can interact with the behavior. Denote the behavior of the environment by $\mathcal{E}$. Generally we need a projection that projects the environment to a behavior of type $(\mathbb{T}, \mathbb{V})$, since the environment can generally be of any type. Denote this projection as π_e. Therefore, the interaction between the behaviors $\mathfrak{B}_1$ and $\mathfrak{B}_2$ and the environment can be modelled as $(\pi_v \mathfrak{B}_1 \parallel \pi_e \mathcal{E})$ and $(\pi_v \mathfrak{B}_2 \parallel \pi_e \mathcal{E})$ respectively. Formally we should denote the external projection of $\mathfrak{B}_1$ and $\mathfrak{B}_2$ differently. However, since it is not likely to raise any confusion, we suppress the notation by denoting both of them by π_v. The trajectories of the environment that are allowed after the interconnection with $\mathfrak{B}_1$ and $\mathfrak{B}_2$ are $\pi_e^{-1}(\pi_v \mathfrak{B}_1 \parallel \pi_e \mathcal{E})$ and $\pi_e^{-1}(\pi_v \mathfrak{B}_2 \parallel \pi_e \mathcal{E})$ respectively.

Lemma 5.5. We have that

$$\pi_e^{-1}(\pi_v \mathfrak{B}_1 \parallel \pi_e \mathcal{E}) = \pi_e^{-1}(\pi_v \mathfrak{B}_2 \parallel \pi_e \mathcal{E}) \tag{5.8}$$

for all possible environments $\mathcal{E}$ if and only if $\mathfrak{B}_1 \approx_{\text{ext}} \mathfrak{B}_2$.

Proof. (if) By definition $\mathfrak{B}_1 \approx_{\text{ext}} \mathfrak{B}_2$ means $\pi_v \mathfrak{B}_1 = \pi_v \mathfrak{B}_2$.

(only if) Suppose that $\mathfrak{B}_1$ and $\mathfrak{B}_2$ are not external behavior equivalent. This means that at least one of the following statements is true
(i) There exists a $v \in \pi_v \mathfrak{B}_1$ such that $v \notin \pi_v \mathfrak{B}_2$.
(ii) There exists a $v \in \pi_v \mathfrak{B}_2$ such that $v \notin \pi_v \mathfrak{B}_1$.
By symmetry, without loss of generality we assume that (i) is true. We shall construct an environment $\mathcal{E}$ such that (5.8) does not hold. Let $\mathcal{E}$ be of type $(\mathbb{T}, \mathbb{V})$ and π_e be the identity map. Moreover, we construct $\mathcal{E}$ such that $v \in \mathcal{E}$. Clearly, v is an element of the left hand side of (5.8) but not of the right hand side. □

Lemma 5.5 tells us that $\mathfrak{B}_1 \approx_{\text{ext}} \mathfrak{B}_2$ means that the environment cannot distinguish between the two. For linear systems, the external behavior is typically associated with the *input-output behavior* [PW98]. The reason behind it is that typically internal variables are the state variables of the system. Thus, eliminating the state variables leaves us the input and output variables, as exemplified in Example 5.4. However, as we shall see later in this chapter, internal variables can also be of a different nature. They can also appear, for example, in the form of an input to the system. In Subsection 5.2.2 we shall encounter this situation.

In the context of interconnection of linear systems, there is another notion of external equivalence of systems. The notion is *transfer function equivalence*. Two systems are said to be transfer function equivalent if they share the same transfer function. The notion of transfer function is briefly introduced in Definition 4.42 (as transfer matrix) and is discussed extensively in textbooks on linear systems and control, for example, [Oga90, Bro91]. In [PW98], it is proven that two systems are transfer function equivalent if and only if the controllable part of their input-output behaviors are equal. Thus, external behavior (input-output behavior) equivalence is stronger than transfer function equivalence.

The notions of external and internal signal space applies to discrete event systems in the following way. Since the signal space of the system is the alphabet (see Chapter 2), we factor the alphabet of the system into

$$E = \mathbb{V} \times \mathbb{D}. \tag{5.9}$$

Again, $\mathbb{V}$ denotes the external part and $\mathbb{D}$ the internal part. We consider the language generated by an automaton as the external behavior of the system. Therefore, two automata are external behavior equivalent if and only if they share the same generated language.

It is well known that in nondeterministic automata, the state is generally not observable from the language of the automata. That is, given a string of events s, the state that can be reached by executing this string is not unique. However, we can use the internal part to enrich the language with more information, such that the state is observable. Consider the example depicted in Figure 5.1. The automaton on the left is nondeterministic. However, if we assume the event a to have external and internal values, it is possible to assign different internal values to the events so that a on the left branch can be differentiated from a on the right branch. This is done in the automaton on the right. In this automaton we no

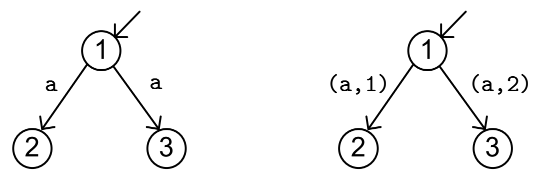


Figure 5.1: Nondeterministic and deterministic automata.

longer have nondeterminism and the state of the automaton is observable from the language.

We want that the state is observable from the trajectory because in the framework that we have built so far, state maps are by constructed on the trajectories of the system Thus, state maps are always observable from the trajectory. In the next section, we shall discuss another notion about external equivalence of systems, which is called *bisimulation*. Our aim is to analyze bisimulation with the framework that we have developed and see how some general results that can be applied to, for example, linear systems and discrete event systems can be obtained.

5.2 Bisimulation as external equivalence

5.2.1 Introduction

The concept of bisimulation originates from the field of discrete event systems / concurrent processes [Par81, Mil89, Arn94, BPS01]. Bisimulation for finite state automata can be defined as follows.

Definition 5.6. Let $A_1 = (X_1, E, T_1, X_{\mathrm{m}1}, x_{01})$ and $A_2 = (X_2, E, T_2, X_{\mathrm{m}2}, x_{02})$ be two finite state automata. A **bisimulation** $\mathcal{R}$ is a relation between X_1 and X_2 (thus $\mathcal{R} \subset X_1 \times X_2$) that satisfies of the following properties.
(1a) For any $(x_1, x_2) \in \mathcal{R}$ and $\mathtt{a} \in E$ such that $(x_1, \mathtt{a}, x_1') \in T_1$ for some $x_1' \in X_1$, there exists an $x_2' \in X_2$ such that $(x_2, \mathtt{a}, x_2') \in T_2$ and $(x_1', x_2') \in \mathcal{R}$.
(1b) For any $(x_1, x_2) \in \mathcal{R}$ and $\mathtt{a} \in E$ such that $(x_2, \mathtt{a}, x_2') \in T_2$ for some $x_2' \in X_2$, there exists an $x_1' \in X_1$ such that $(x_1, \mathtt{a}, x_1') \in T_1$ and $(x_1', x_2') \in \mathcal{R}$.
(2) $(x_{01}, x_{02}) \in \mathcal{R}$.

Definition 5.7. When two finite state automata A_1 and A_2 are such that there exists a bisimulation between them, as defined in Definition 5.6, we say that A_1 and A_2 are **bisimilar**. This property is denoted as $A_1 \approx_{\mathrm{bis}} A_2$.

Notice that bisimilarity relation $\approx_{\mathrm{bis}}$ can be regarded as a relation among finite state automata. Furthermore, we can prove the following results.

Lemma 5.8. (cf. Proposition 8.1 in [Arn94])For any finite state automata A_1, A_2, and A_3, the following statements hold.

(reflexivity) $A_1 \approx_{\text{bis}} A_1$.
(commutativity) If $A_1 \approx_{\text{bis}} A_2$ then $A_2 \approx_{\text{bis}} A_1$.
(transitivity) If $A_1 \approx_{\text{bis}} A_2$ and $A_2 \approx_{\text{bis}} A_3$ then $A_1 \approx_{\text{bis}} A_3$.

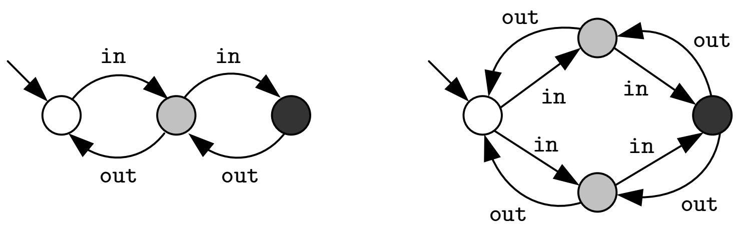
Proof. (reflexivity) Construct $\mathcal{R}$ as the identity relation in the set of states of A_1. (commutativity) Let $\mathcal{R}$ be the bisimulation relation that defines $A_1 \approx_{\text{bis}} A_2$. Due to the symmetric nature of Definition 5.6, it can be proven that $\mathcal{R}^{-1}$ is a bisimulation that defines $A_2 \approx_{\text{bis}} A_1$. (transitivity) Let $\mathcal{R}_{12}$ and $\mathcal{R}_{23}$ be the bisimulation relations that define $A_1 \approx_{\text{bis}} A_2$ and $A_2 \approx_{\text{bis}} A_3$ respectively. It can be readily proven that the composite relation

$$\mathcal{R}_{13} := \mathcal{R}_{12} \circ \mathcal{R}_{23}, \tag{5.10}$$
$$:= \{(\xi, \xi'') \mid \exists \xi' \text{ such that } (\xi, \xi') \in \mathcal{R}_{12} \text{ and } (\xi', \xi'') \in \mathcal{R}_{23}\} \tag{5.11}$$

is a bisimulation relation that defines $A_1 \approx_{\text{bis}} A_3$. □

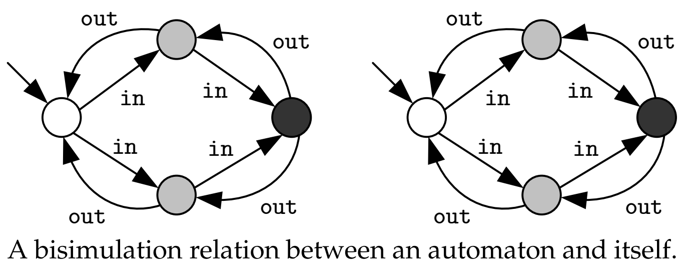
Lemma 5.8 shows that bisimulation can be regarded as an equivalence relation in the class of finite state automata.

Example 5.9. (adopted from [Her02]) The following two automata are models of a data buffer system. These two automata are bisimilar. The bisimulation relation is encoded in the shading of the states. States with similar shading are bisimilar. We can visually verify that Definition 5.6 is indeed satisfied.



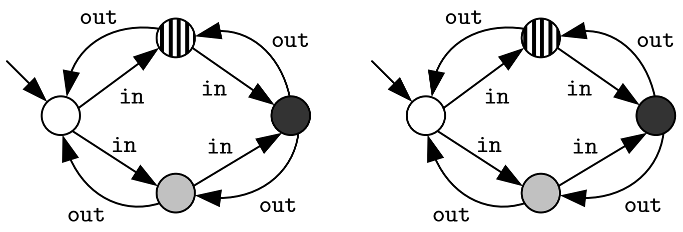
The two automata and the bisimulation relation. The bisimulation relation is expressed as the shading of the states.

Given the fact two automata are bisimilar, there can be more than just one bisimulation relation between them. Consider the right automaton in Example 5.9. The following illustration depicts a bisimulation relation between the automaton and itself.



A bisimulation relation between an automaton and itself.

However, we also know that the identity relation in the set of states also defines a bisimulation relation. This fact is shown as follows.



Another bisimulation between the automaton and itself.

The following result can be proven about the relation between all bisimulation relations between two automata.

Lemma 5.10. (cf. Proposition 8.1 in [Arn94]) For any finite state automata A_1 and A_2, let $(\mathcal{R}_i)_{i\in I}$ be a family of bisimulation relations between A_1 and A_2. The set I is an index set of the family. The relation

$$\mathcal{R} := \bigcup_{i\in I} \mathcal{R}_i \tag{5.12}$$

is also a bisimulation relation between A_1 and A_2.

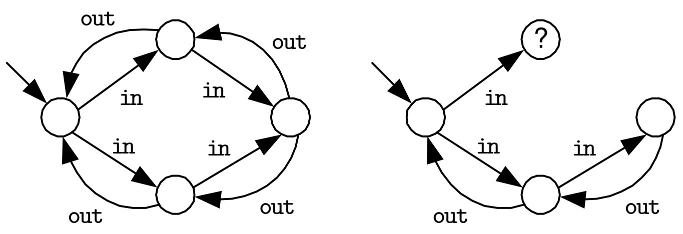
A consequence of Lemma 5.10 can be formulated as follows.

Corollary 5.11. For any bisimilar automata $A_1 \approx_{\text{bis}} A_2$, there exists a unique bisimulation $\mathcal{R}$ such that any other bisimulation relation is a subset of $\mathcal{R}$. This relation $\mathcal{R}$ is called the *maximal bisimulation* relation.

The relation between bisimilarity as a notion of equivalence between automata and generated language equivalence can be presented as follows.

Theorem 5.12. Given two finite state automata A_1 and A_2, if $A_1 \approx_{\text{bis}} A_2$ then $L(A_1) = L(A_2)$. The converse is generally not true.

We do not present a proof of this theorem, as it is a well known result. However, we present the following counterexample, in which the converse of the theorem is not true.



A counterexample showing two nonbisimilar automata with equal language.

To see that these two automata are not bisimilar, simply look at the state marked by '?'. This state cannot be related to any state in the other automaton as it does not have outgoing transition from it.

In the discussion so far, we have seen that bisimulation has a symmetry property with respect to the automata involved. A simulation can be thought of as a unidirectional version of bisimulation.

Definition 5.13. Let $A_1 = (X_1, E, T_1, X_{m1}, x_{01})$ and $A_2 = (X_2, E, T_2, X_{m2}, x_{02})$ be two finite state automata. A **simulation** of A_2 by A_1 is a relation $\mathcal{R} \subset X_1 \times X_2$ that satisfies all of the following statements.
(1) For any $(x_1, x_2) \in \mathcal{R}$ and $\mathtt{a} \in E$ such that $(x_2, \mathtt{a}, x_2') \in T_2$ for some $x_2' \in X_2$, there exists an $x_1' \in X_1$ such that $(x_1, \mathtt{a}, x_1') \in T_1$ and $(x_1', x_2') \in \mathcal{R}$.
(2) $(x_{01}, x_{02}) \in \mathcal{R}$.

When there exists a simulation of A_2 by A_1, we say that A_1 simulates A_2. This property is denoted by $A_1 \succ_{\text{sim}} A_2$.

As an analog of Lemma 5.8, we can present the following result.

Lemma 5.14. For any finite state automata A_1, A_2, and A_3, the following statements hold.
(reflexivity) $A_1 \succ_{\text{sim}} A_1$.
(transitivity) If $A_1 \succ_{\text{sim}} A_2$ and $A_2 \succ_{\text{sim}} A_3$ then $A_1 \succ_{\text{sim}} A_3$.

Notation 5.15. If $A_1 \succ_{\text{sim}} A_2$ and $A_2 \succ_{\text{sim}} A_1$, we denote this fact by $A_1 \approx_{\text{sim}} A_2$. The equivalence relation $\approx_{\text{sim}}$ is called **mutual simulation**.

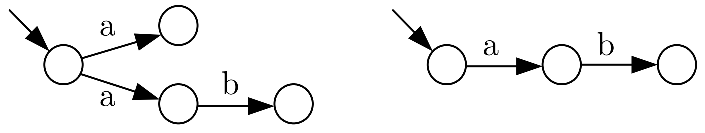
Lemma 5.14 tells us that simulation can be regarded as a partial ordering in the class of finite state automata. It is natural to conjecture that the equivalence relation induced by this partial ordering coincides with the bisimilarity equivalence relation. However, this is not the case. The relation between simulation and bisimulation can be expressed as follows.

Theorem 5.16. Given two finite state automata A_1 and A_2, if $A_1 \approx_{\text{bis}} A_2$ then

$$A_1 \succ_{\text{sim}} A_2, \tag{5.13}$$

$$A_2 \succ_{\text{sim}} A_1. \tag{5.14}$$

The converse of the theorem is generally not true. This is demonstrated by the following counterexample. These two automata mutually simulate each other, but they are not bisimilar.

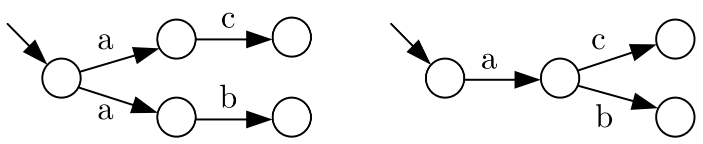


A counterexample showing two automata that mutually simulate each other, but are not bisimilar.

The relation between simulation and the generated language of the automata is presented in the following lemma.

Lemma 5.17. Given two finite state automata A_1 and A_2, if $A_1 \succ_{\mathrm{sim}} A_2$ then $L(A_1) \supseteq L(A_2)$.

The converse of this lemma is generally not true. A counterexample that proves it is presented below. The two automata generate the same language. However, the automaton on the left does not simulate the automaton on the right.



A counterexample showing two automata that share the same language, but do not mutually simulate each other.

So far we have discussed three notions of equivalence of finite state automata. They are bisimilarity ($\approx_{\mathrm{bis}}$), mutual simulation ($\approx_{\mathrm{sim}}$), and generated language equivalence ($\approx_{\mathrm{ext}}$). The relation between these notions can be summed up as follows.

$$\approx_{\mathrm{bis}} \subseteq \approx_{\mathrm{sim}} \subseteq \approx_{\mathrm{ext}} . \tag{5.15}$$

Remark 5.18. In terms of complexity, bisimulation of two automata is easier to check than language equality [KS90, ABGS91, HS96].

5.2.2 Bisimulation of other types of systems

Following recent increased interest in hybrid systems, some efforts have been made to extend known theories in both discrete event systems and continuous dynamical systems to the field of hybrid systems. Similarly, there also has been traffic of cross applications of theories between discrete event systems and continuous dynamical systems. Bisimulation is one of them.

The concept of bisimulation has been extended and applied to hybrid systems, for example in [LPS98, AHLP00, Sch04a, Sch04b, PvD04]. In addition, it has also been applied to continuous dynamical systems, for example in [Pap03, Sch03b, Sch04a] as well as discrete time dynamical systems [Pap03, Tab04].

Bisimulation is used as a notion of external equivalence of systems because it enables *consistent abstraction* of systems. That is, with bisimulation the system can be reduced to a quotient system with smaller state space, that preserves some properties of interest [Pap03]. In particular, bisimulation is known to preserve properties that can be expressed in temporal logics, such as, linear temporal logic (LTL) and computational tree logic (CTL) [AHLP00, DN00].

The approach taken in [Pap03] is to create a transition system corresponding to systems of the form

$$\frac{dx}{dt} = Ax + Bu, \tag{5.16}$$

$$y = Cx. \tag{5.17}$$

or its discrete time counterpart

$$x(k+1) = Ax(k) + Bu(k), \tag{5.18}$$

$$y(k) = Cx(k). \tag{5.19}$$

The transition system is formed as an abstraction of the dynamical system under study. The abstraction can be done at different levels, for example, time can be present or it can be abstracted as well. The study of bisimulation of such systems is a study about partitioning the state space into classes compatible with the linear output map and requiring that the partitions are bisimilar.

The approach taken in [Sch03b] is more direct, in the sense that it does not explicitly define transition systems corresponding to the systems under study. The dynamical systems studied there is in the following form.

$$\frac{dx}{dt} = Ax + Bu + Gd, \tag{5.20a}$$

$$y = Cx. \tag{5.20b}$$

This is a standard state-space representation, except for the variable $\mathbf{d}$. This variable represents the so called *disturbance* that generates nondeterminism in the system.

Bisimulation for systems in the form of (5.20) is defined as follows.

Definition 5.19. [Sch03b] Consider two dynamical systems of the form

$$\Sigma_i : \begin{array}{ll} \frac{dx_i}{dt} = A_i x_i + B_i u_i + G_i d_i, & x_i \in \mathcal{X}_i, u_i \in \mathcal{U}, d_i \in \mathcal{D}_i \\ y_i = C_i x_i, & y_i \in \mathcal{Y} \; i = 1, 2 \end{array} \tag{5.21}$$

with $\mathcal{X}_i, \mathcal{D}_i, \mathcal{U}, \mathcal{Y}$ finite dimensional linear spaces.
A **bisimulation** $\mathcal{R}$ between Σ_1 and Σ_2 is a linear subspace $\mathcal{R} \subset \mathcal{X}_1 \times \mathcal{X}_2$ with the following property.

$$\forall x_1 \in \mathcal{X}_1, \exists x_2 \in \mathcal{X}_2 \text{ such that } (x_1, x_2) \in \mathcal{R}, \tag{5.22}$$

$$\forall x_2 \in \mathcal{X}_2, \exists x_1 \in \mathcal{X}_1 \text{ such that } (x_1, x_2) \in \mathcal{R}. \tag{5.23}$$

Furthermore, if we take any $(x_{10}, x_{20}) \in \mathcal{R}$ and any joint input function $u_1(\cdot) = u_2(\cdot)$. Then for every disturbance function $d_1(\cdot)$ there should exists a disturbance function $d_2(\cdot)$ such that the resulting state solution trajectories $x_1(\cdot)$, with $x_1(0) = x_{10}$, and $x_2(\cdot)$, with $x_2(0) = x_{20}$, satisfy
(i) $(x_1(t), x_2(t)) \in \mathcal{R}$, for all $t \geq 0$

(ii) $C_1x_1(t) = C_2x_2(t)$, for all $t \geq 0$
Conversely, for every disturbance function $d_2(\cdot)$ there should exists a disturbance function $d_1(\cdot)$ such that the resulting state trajectories satisfy (i) and (ii).

Remark 5.20. Definition 5.19 slightly differs from the definition in [Sch03b]. The difference is that in the original definition in [Sch03b], a bisimulation relation is not required to satisfy the requirement (5.22) and (5.23). However, two systems are said to be bisimilar, if there exists a bisimulation that satisfies these additional requirements. Thus, we just move the requirements from the definition of bisimilarity to the definition of the bisimulation relation. The definition of bisimilar systems is therefore the same.

Two systems, between which a bisimulation relation exists, are said to be bisimilar. Among the important results presented in [Sch03b] are:

- Bisimilarity among systems is an equivalence relation.
- An algorithm to compute the largest bisimulation relation is derived.
- Bisimilarity implies equality of external behaviors.
- Simulation between systems of the form (5.20) can be defined in a way analogous to Definition 5.19. In the paper, it is proven that mutual simulation coincides with bisimilarity.

5.3 Bisimulation in the behavioral setting

In this section we shall discuss bisimulation in the general behavioral setting. Studies of bisimulation in the general systems framework, encompassing more than one class of systems, have also been done before. The reader is referred to, for example, the work by Haghverdi *et.al.* in the category theoretical framework [HTP02]; and to the more recent work by the same authors [HTP03].

A comparison between [HTP02, HTP03] and what we are going to do is that in the former the analysis is geared towards general bisimulation for general systems, while in the latter a particular (non abstract) bisimulation is considered for general systems.

As what we have discussed so far, bisimulation is defined as a notion of external equivalence of systems. However, we need to emphasize the following idea.

At the beginning of this book, we introduced dynamical systems as a triple $\Sigma = (\mathbb{T}, \mathbb{W}, \mathfrak{B})$. In this point of view, dynamical systems are identified by their behaviors. What we are going to discuss in this section is a notion of (external) equivalence of systems. However, the equivalence is not between dynamical systems as defined by their behaviors. Rather, we are going to define equivalence between systems equipped with a state map. Furthermore, we are also going to assume that the signal space of the systems is factored into the external and internal signal space as in Section 5.1.

Definition 5.21. Given two dynamical systems $\Sigma_1 = (\mathbb{T}, \mathbb{V} \times \mathbb{D}_1, \mathfrak{B}_1)$ and $\Sigma_2 = (\mathbb{T}, \mathbb{V} \times \mathbb{D}_2, \mathfrak{B}_2)$, with state maps x_1 and x_2 respectively. We denote the codomain of the state maps as $\mathcal{X}_1$ and $\mathcal{X}_2$ respectively. A **bisimulation** relation $\mathcal{R} \subset \mathcal{X}_1 \times \mathcal{X}_2$ is a relation with the following property.

$$\forall \xi_1 \in \mathcal{X}_1, \exists \xi_2 \in \mathcal{X}_2 \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}, \tag{5.24}$$
$$\forall \xi_2 \in \mathcal{X}_2, \exists \xi_1 \in \mathcal{X}_1 \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}. \tag{5.25}$$

Furthermore, if we take any $(\xi_1, \xi_2) \in \mathcal{R}$. Then, given any $w_1 := (v_1, d_1) \in \mathfrak{B}_1$ and $t_1 \in \mathbb{T}$ such that $x_1(w_1, t_1) = \xi_1$, the following holds. If $t_2 \in \mathbb{T}$ is such that there exists a $w' := (v', d') \in \mathfrak{B}_2$ such that $x_2(w', t_2) = \xi_2$, then there exists a $d_2 \in \pi_d \mathfrak{B}_2$ such that if we define

$$v_2 := v' \wedge_{t_1}^{t_2} v_1, \tag{5.26}$$
$$w_2 := (v_2, d_2), \tag{5.27}$$

we have that

$$w_2 \in \mathfrak{B}_2, \tag{5.28}$$
$$x_2(w_2, t_2) = \xi_2, \tag{5.29}$$
$$d_2(\tau) = d'(\tau), \forall \tau \leq t_2, \tag{5.30}$$

and for all $\tau > t_2$,

$$(x_1(w_1, \tau - t_2 + t_1), x_2(w_2, \tau)) \in \mathcal{R}. \tag{5.31}$$

Conversely, given any $w_2 := (v_2, d_2) \in \mathfrak{B}_2$ and $t_2 \in \mathbb{T}$ such that $x_2(w_2, t_2) = \xi_2$, the following holds. If $t_1 \in \mathbb{T}$ is such that there exists a $w' \in \mathfrak{B}_1$ such that $x_1(w', t_1) = \xi_1$, then there exists a $d_1 \in \pi_d \mathfrak{B}_1$ such that if we define

$$v_1 := v' \wedge_{t_2}^{t_1} v_2, \tag{5.32}$$
$$w_1 := (v_1, d_1), \tag{5.33}$$

we have that

$$w_1 \in \mathfrak{B}_1, \tag{5.34}$$
$$x_1(w_1, t_1) = \xi_1, \tag{5.35}$$
$$d_1(\tau) = d'(\tau), \forall \tau \leq t_1, \tag{5.36}$$

and for all $\tau > t_2$, (5.31) holds.

In words, the bisimulation requires that from any two bisimilar states[1] it is possible to proceed with equal external trajectories while visiting states that are bisimilar. This formulation can be regarded as a generalized version of Definition 5.19, which is used in [Sch03b]. To obtain Definition 5.19 from Definition 5.21, we can use the following comparison.

[1]Two states are bisimilar if they are related by the bisimulation relation.

Definition 5.21	Definition 5.19
x_1, x_2	x_1, x_2
$\mathbb{V}$	$\mathcal{U} \times \mathcal{Y}$
$\mathbb{D}_1, \mathbb{D}_2$	$\mathcal{D}_1, \mathcal{D}_2$

In addition to that, we need to assume that the state space representation in Definition 5.19 is observable, since state maps are observable by default. However, later we shall show that we do not lose any generality by requiring that the state space representation is observable. This is because we can always factor out the state space into the observable and non-observable part, work out the bisimulation for the observable part and then extend it to cover the non-observable part as well.

Definition 5.22. Two dynamical systems Σ_1 and Σ_2 equipped with the state maps x_1 and x_2 are said to be **bisimilar** if there exists a bisimulation relation between them.

Notation 5.23. We shall denote the dynamical system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathfrak{B})$ equipped with the state map x by the pair (Σ, x). The fact that (Σ_1, x_1) and (Σ_2, x_2) are bisimilar is denoted by $(\Sigma_1, x_1) \approx_{\text{bis}} (\Sigma_2, x_2)$.

5.3.1 Bisimilarity as an equivalence relation

Since the definition of bisimulation in Definition 5.19 induces the bisimilarity relation $\approx_{\text{bis}}$, which is an equivalence relation, we expect that the bisimulation relation in the sense of Definition 5.21 also induces an equivalence relation among pairs (Σ, x). The following propositions establish the fact that $\approx_{\text{bis}}$ is symmetric and transitive.

Proposition 5.24. For any pairs (Σ_i, x_i), $i = 1, 2$, if $(\Sigma_1, x_1) \approx_{\text{bis}} (\Sigma_2, x_2)$ then $(\Sigma_2, x_2) \approx_{\text{bis}} (\Sigma_1, x_1)$.

Proof. Suppose that $\mathcal{R} \subset \mathcal{X}_1 \times \mathcal{X}_2$ is the bisimulation relation that defines $(\Sigma_1, x_1) \approx_{\text{bis}} (\Sigma_2, x_2)$. The bisimulation relation that defines $(\Sigma_2, x_2) \approx_{\text{bis}} (\Sigma_1, x_1)$ can be constructed as $\mathcal{R}^{-1}$. That is,

$$(\xi, \zeta) \in \mathcal{R}^{-1} \Leftrightarrow (\zeta, \xi) \in \mathcal{R}. \tag{5.37}$$

By the symmetry of the Definition 5.21 we can verify that $\mathcal{R}^{-1}$ is indeed a bisimulation relation between (Σ_2, x_2) and (Σ_1, x_1). □

Proposition 5.25. For any pairs (Σ_i, x_i), $i = 1, 2, 3$. If $(\Sigma_1, x_1) \approx_{\text{bis}} (\Sigma_2, x_2)$ and $(\Sigma_2, x_2) \approx_{\text{bis}} (\Sigma_3, x_3)$, then $(\Sigma_1, x_1) \approx_{\text{bis}} (\Sigma_3, x_3)$.

Proof. Let $\mathcal{R}_{12}$ and $\mathcal{R}_{23}$ be the bisimulation relations that define the bisimilarities $(\Sigma_1, x_1) \approx_{\text{bis}} (\Sigma_2, x_2)$ and $(\Sigma_2, x_2) \approx_{\text{bis}} (\Sigma_3, x_3)$ respectively. We construct a candidate bisimulation relation between (Σ_1, x_1) and (Σ_3, x_3) as follows.

$$\mathcal{R}_{13} := \mathcal{R}_{12} \circ \mathcal{R}_{23}. \tag{5.38}$$

Now we are going to verify that $\mathcal{R}_{13}$ is indeed a bisimulation relation. The fact that the following relations hold is obvious.

$$\forall \xi_1 \in \mathcal{X}_1, \exists \xi_3 \in \mathcal{X}_3 \text{ such that } (\xi_1, \xi_3) \in \mathcal{R}_{13}, \tag{5.39}$$

$$\forall \xi_3 \in \mathcal{X}_3, \exists \xi_1 \in \mathcal{X}_1 \text{ such that } (\xi_1, \xi_3) \in \mathcal{R}_{13}. \tag{5.40}$$

Further, if we take any $(\xi_1, \xi_3) \in \mathcal{R}_{13}$. Then, given any $w_1 := (v_1, d_1) \in \mathfrak{B}_1$ and $t_1 \in \mathbb{T}$ such that $x_1(w_1, t_1) = \xi_1$, the following holds. If $t_3 \in \mathbb{T}$ is such that there exists a $w' := (v', d') \in \mathfrak{B}_3$ such that $x_3(w', t_3) = \xi_3$, then we can show that there exists a $d_3 \in \pi_d \mathfrak{B}_3$ such that if we define

$$v_3 := v' \wedge_{t_1}^{t_3} v_1, \tag{5.41}$$

$$w_3 := (v_3, d_3), \tag{5.42}$$

we have that

$$w_3 \in \mathfrak{B}_3, \tag{5.43}$$

$$x_3(w_3, t_3) = \xi_3, \tag{5.44}$$

$$d_3(\tau) = d'(\tau), \forall \tau \leq t_3, \tag{5.45}$$

and for all $\tau > t_3$,

$$(x_1(w_1, \tau - t_3 + t_1), x_3(w_3, \tau)) \in \mathcal{R}_{13}. \tag{5.46}$$

From (5.38) we can conclude that there exists a $\xi_2 \in \mathcal{X}_2$ such that $(\xi_1, \xi_2) \in \mathcal{R}_{12}$ and $(\xi_2, \xi_3) \in \mathcal{R}_{23}$. Let $\tilde{w} := (\tilde{v}, \tilde{d}) \in \mathfrak{B}_2$ and $t_2 \in \mathbb{T}$ be such that $x_2(\tilde{w}, t_2) = \xi_2$. Since $(\xi_1, \xi_2) \in \mathcal{R}_{12}$, this implies the existence of a $d_2 \in \pi_d \mathfrak{B}_2$ such that if we define

$$v_2 := \tilde{v} \wedge_{t_1}^{t_2} v_1, \tag{5.47}$$

$$w_2 := (v_2, d_2), \tag{5.48}$$

we have that

$$w_2 \in \mathfrak{B}_2, \tag{5.49}$$

$$x_2(w_2, t_2) = \xi_2, \tag{5.50}$$

$$d_2(\tau) = \tilde{d}(\tau), \forall \tau \leq t_2, \tag{5.51}$$

and for all $\tau > t_2$,

$$(x_1(w_1, \tau - t_2 + t_1), x_2(w_2, \tau)) \in \mathcal{R}_{12}. \tag{5.52}$$

Moreover, since $(\xi_2, \xi_3) \in \mathcal{R}_{23}$, there exists a $\tilde{d}_3 \in \pi_d \mathfrak{B}_3$ such that if we define

$$\tilde{v}_3 := v' \wedge_{t_2}^{t_3} v_2, \tag{5.53}$$

$$\tilde{w}_3 := (\tilde{v}_3, \tilde{d}_3), \tag{5.54}$$

we have that

$$\tilde{w}_3 \in \mathfrak{B}_3, \tag{5.55}$$
$$x_3(\tilde{w}_3, t_3) = \xi_3, \tag{5.56}$$
$$\tilde{d}_3(\tau) = d'(\tau), \forall \tau \leq t_3, \tag{5.57}$$

and for all $\tau > t_3$,

$$(x_2(w_2, \tau - t_3 + t_2), x_3(\tilde{w}_3, \tau)) \in \mathcal{R}_{23}. \tag{5.58}$$

Now, notice that $\tilde{v}_3 = v_3$, thus if we take $d_3 := \tilde{d}_3$ then (5.43) - (5.46) are satisfied. So far we have proven that $\mathcal{R}_{13}$ is a simulation relation of (Σ_1, x_1) by (Σ_3, x_3). The converse can be proven analogously to the proof above. □

To prove that $\approx_{\text{bis}}$ is also reflexive, we need to construct a bisimulation between any pair (Σ, x) with itself. The usual candidate for this purpose is the identity relation. However, as we see in the following proposition, we need more assumptions to be able to use the identity relation as the bisimulation relation.

Proposition 5.26. For any pair (Σ, x), if x is Markovian and past induced then $(\Sigma, x) \approx_{\text{bis}} (\Sigma, x)$.

Proof. We have to construct a bisimulation relation (in the sense of Definition 5.21) between (Σ, x) and itself. We use the usual bisimulation to prove bisimilarity between a system and itself, namely, the identity relation

$$\mathcal{R} \subset \mathcal{X} \times \mathcal{X}, \tag{5.59}$$
$$(\xi, \zeta) \in \mathcal{R} \text{ if and only if } \xi = \zeta. \tag{5.60}$$

Clearly, for all $\xi \in \mathcal{X}$ there exists an $\xi' \in \mathcal{X}$ such that $(\xi, \xi') \in \mathcal{R}$. Furthermore, if we take any $(\xi, \xi) \in \mathcal{R}$. Then, given any $w_1 := (v_1, d_1) \in \mathfrak{B}$ and $t_1 \in \mathbb{T}$ such that $x(w_1, t) = \xi$, the following holds. If $t_2 \in \mathbb{T}$ is such that there exists a $w' := (v', d') \in \mathfrak{B}$ such that $x(w', t_2) = \xi$, then we can show that there exists a $d_2 \in \pi_d \mathfrak{B}$ such that if we define

$$v_2 := v' \wedge_{t_1}^{t_2} v_1, \tag{5.61}$$
$$w_2 := (v_2, d_2), \tag{5.62}$$

we have that

$$w_2 \in \mathfrak{B}, \tag{5.63a}$$
$$x(w_2, t_2) = \xi, \tag{5.63b}$$
$$d_2(\tau) = d'(\tau), \forall \tau \leq t_2, \tag{5.63c}$$

and for all $\tau > t_2$,

$$(x(w_1, \tau - t_2 + t_1), x(w_2, \tau)) \in \mathcal{R}. \tag{5.64}$$

Since x is a state map, we can construct $w_2 := w' \wedge_{t_1}^{t_2} w_1$ and have $w_2 \in \mathfrak{B}$. Thus, (5.63a) and (5.63c) are satisfied. To show that (5.63b) is also satisfied, we use the fact that x is past induced. With this fact, we can infer that

$$x(w_2, t_2) = x(w', t_2) = \xi_2. \tag{5.65}$$

From here, (5.64) follows from the fact that x is Markovian. □

In the proof of Proposition 5.26, we see that the Markovian and past inducedness properties are sufficient conditions for the identity relation to be a bisimulation relation. Therefore, hereinafter we are going to restrict our attention to past induced Markovian state maps. Before we proceed, it is worthwhile to notice that the state construction of linear systems and deterministic finite state automata are Markovian and past induced.

Summing up the discussion about $\approx_{\text{bis}}$, we can present the following theorem.

Theorem 5.27. The bisimilarity relation $\approx_{\text{bis}}$ is an equivalence relation for the class of pairs (Σ, x), where x is past induced and Markovian.

Proof. This is a direct consequence of Propositions 5.24, 5.25, and 5.26. □

5.3.2 The maximal bisimulation relation

We present a result analogous to Lemma 5.10.

Lemma 5.28. Given two systems $\Sigma_i := (\mathbb{T}, \mathbb{V} \times \mathbb{D}_i, \mathfrak{B}_i)$, $i = 1, 2$, and their respective state maps x_1 and x_2. Let (Σ_1, x_1) and (Σ_2, x_2) be bisimilar and let $(\mathcal{R}_i)_{i \in I}$ be a family of bisimulation relations between (Σ_1, x_1) and (Σ_2, x_2). The set I is an index set of the family. The relation

$$\mathcal{R} := \bigcup_{i \in I} \mathcal{R}_i \tag{5.66}$$

is also a bisimulation relation.

Proof. Obviously we have that

$$\forall \xi_1 \in \mathcal{X}_1, \exists \xi_2 \in \mathcal{X}_2 \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}, \tag{5.67}$$
$$\forall \xi_2 \in \mathcal{X}_2, \exists \xi_1 \in \mathcal{X}_1 \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}. \tag{5.68}$$

Furthermore, if we take any $(\xi_1, \xi_2) \in \mathcal{R}$. Then, given any $w_1 := (v_1, d_1) \in \mathfrak{B}_1$ and $t_1 \in \mathbb{T}$ such that $x_1(w_1, t_1) = \xi_1$, the following holds. If $t_2 \in \mathbb{T}$ is such that there exists a $w' := (v', d') \in \mathfrak{B}_2$ such that $x_2(w', t_2) = \xi_2$, then we can show that there exists a $d_2 \in \pi_d \mathfrak{B}_2$ such that if we define

$$v_2 := v' \wedge_{t_1}^{t_2} v_1, \tag{5.69}$$
$$w_2 := (v_2, d_2), \tag{5.70}$$

we have that

$$w_2 \in \mathfrak{B}_2, \tag{5.71}$$

$$x_2(w_2, t_2) = \xi_2, \tag{5.72}$$

$$d_2(\tau) = d'(\tau), \forall \tau \leq t_2, \tag{5.73}$$

and for all $\tau > t_2$,

$$(x_1(w_1, \tau - t_2 + t_1), x_2(w_2, \tau)) \in \mathcal{R}. \tag{5.74}$$

Since $(\xi_1, \xi_2) \in \mathcal{R}$, then $(\xi_1, \xi_2) \in \mathcal{R}_i$ for some $i \in I$. The relation $\mathcal{R}_i$ is a bisimulation relation, therefore the existence of such d_2 is guaranteed. So far we have proven that $\mathcal{R}$ is a simulation relation of (Σ_1, x_1) by (Σ_2, x_2). The converse can be proven analogously to the proof above. □

Lemma 5.28 implies the existence of a *maximal bisimulation* relation between two bisimilar systems. In addition to that, we can assume that a bisimulation relation $\mathcal{R}$ as in Definition 5.21 possesses a kind of *homogeneity* property, such that for all $\xi_i, \eta_i \in \mathcal{X}_i$, $i = 1, 2$,

$$\{(\xi_1, \xi_2) \in \mathcal{R} \wedge (\xi_1, \eta_2) \in \mathcal{R} \wedge (\eta_1, \xi_2) \in \mathcal{R}\} \Rightarrow \{(\eta_1, \eta_2) \in \mathcal{R}\}. \tag{5.75}$$

This is due to the fact that we can prove that the *homogenized bisimulation relation* $\mathcal{R}_{\text{hom}}$, defined as

$$\mathcal{R}_{\text{hom}} := \mathcal{R} \circ \overline{(\mathcal{R}^{-1} \circ \mathcal{R})}, \tag{5.76}$$

is a bisimulation relation, if $\mathcal{R}$ is a bisimulation relation. The overbar in (5.76) indicates *equivalence closure*. The equivalence closure of a relation $\mathcal{R}$ is the smallest equivalence relation containing $\mathcal{R}$.

Proposition 5.29. Given two systems $\Sigma_i := (\mathbb{T}, \mathbb{V} \times \mathbb{D}_i, \mathfrak{B}_i)$, $i = 1, 2$, and their respective state maps x_1 and x_2. If a relation $\mathcal{R}$ is a bisimulation between (Σ_1, x_1) and (Σ_2, x_2), then the homogenized relation $\mathcal{R}_{\text{hom}}$ is a bisimulation relation as well.

Proof. Take any $(\xi_1, \xi_2) \in \mathcal{R}_{\text{hom}}$. Let $w_1 := (v_1, d_1) \in \mathfrak{B}_1$ and $t_1 \in \mathbb{T}$ such that $x_1(w_1, t_1) = \xi_1$. If $t_2 \in \mathbb{T}$ is such that there exists a $w' := (v', d') \in \mathfrak{B}_2$ such that $x_2(w', t_2) = \xi_2$, then we can show that there exists a $d_2 \in \pi_d \mathfrak{B}_2$ such that if we define

$$v_2 := v' \wedge_{t_1}^{t_2} v_1, \tag{5.77}$$

$$w_2 := (v_2, d_2), \tag{5.78}$$

we have that

$$w_2 \in \mathfrak{B}_2, \tag{5.79}$$

$$x_2(w_2, t_2) = \xi_2, \tag{5.80}$$

$$d_2(\tau) = d'(\tau), \forall \tau \leq t_2, \tag{5.81}$$

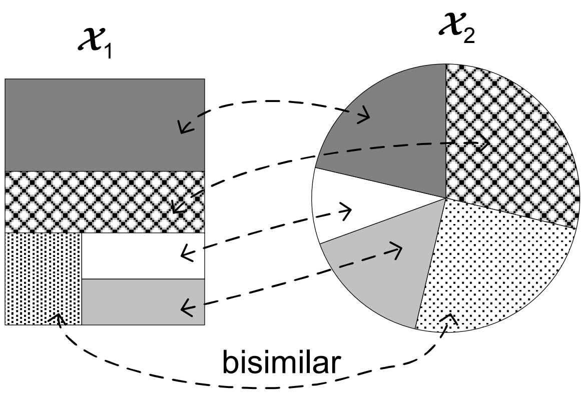


Figure 5.2: An illustration of the equivalence classes in $\mathcal{X}_1$ and $\mathcal{X}_2$ induced by the homogenized bisimulation relation. Classes with the same shading are bisimilar (related through $\mathcal{R}_{\text{hom}}$).

and for all $\tau > t_2$,

$$(x_1(w_1, \tau - t_2 + t_1), x_2(w_2, \tau)) \in \mathcal{R}_{\text{hom}}. \tag{5.82}$$

By construction (5.76), there exists a finite sequence $\eta_k|_{1 \le k \le 2n}$, $n \ge 0$, such that

$$\begin{gathered}\eta_1 = \xi_1 \text{ and } \eta_{2n} = \xi_2,\\ (\eta_i, \eta_{i+1}) \in \mathcal{R}, \text{ if } i \text{ is odd and}\\ (\eta_{i+1}, \eta_i) \in \mathcal{R}, \text{ if } i \text{ is even.}\end{gathered}$$

By following this chain of bisimilar states, we can infer the existence of such a d_2. So far we have proven that $\mathcal{R}_{\text{hom}}$ is a simulation relation of (Σ_1, x_1) by (Σ_2, x_2). The converse can be proven analogously to the proof above. □

The homogenized bisimulation relation has a special property that it generates partitions in $\mathcal{X}_1$ and $\mathcal{X}_2$ and these partitions are related by isomorphic bisimulation relation. This situation is depicted in Figure 5.2. The partitions in $\mathcal{X}_1$ are induced by the equivalence relation $\mathcal{R} \circ \mathcal{R}^{-1}$, while the partitions in $\mathcal{X}_2$ are induced by the equivalence relation $\mathcal{R}^{-1} \circ \mathcal{R}$.

Remark 5.30. The fact that the maximal bisimulation exists and that bisimulation generates a partition in the state space implies that there is a maximal lumping of states that are related by the bisimulation. By maximal lumping we mean making the partition as coarse as possible.

5.3.3 Bisimilarity and state reduction

As is mentioned before, one of the main purposes of constructing a bisimulation relation between bisimilar systems is to facilitate consistent state reduction. The following proposition reveals the relation between bisimulation and the partial ordering of the state maps.

Proposition 5.31. Given pairs (Σ, x_1) and (Σ, x_2), where Σ is a dynamical system, and x_1 and x_2 are state maps of it. If $x_1 \succcurlyeq x_2$ and they are both Markovian and past induced, then

$$(\Sigma, x_1) \approx_{\text{bis}} (\Sigma, x_2). \tag{5.83}$$

Proof. Let $\mathcal{X}_1$ and $\mathcal{X}_2$ be the state space of x_1 and x_2 respectively. Since $x_1 \succcurlyeq x_2$, we know that there is a surjective map $\phi : \mathcal{X}_1 \to \mathcal{X}_2$ such that for any $w \in \mathfrak{B}$ and $t \in \mathbb{T}$,

$$x_2(w, t) = \phi(x_1(w, t)). \tag{5.84}$$

A candidate for the bisimulation relation between (Σ, x_1) and (Σ, x_2) can be constructed as follows.

$$\mathcal{R} \subset \mathcal{X}_1 \times \mathcal{X}_2, \tag{5.85}$$

$$(\xi_1, \xi_2) \in \mathcal{R} \Leftrightarrow \xi_2 = \phi(\xi_1). \tag{5.86}$$

The fact that the following relations hold is obvious.

$$\forall \xi_1 \in \mathcal{X}_1, \exists \xi_2 \in \mathcal{X}_2 \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}, \tag{5.87}$$

$$\forall \xi_2 \in \mathcal{X}_2, \exists \xi_1 \in \mathcal{X}_1 \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}. \tag{5.88}$$

Further, if we take any $(\xi_1, \xi_2) \in \mathcal{R}$. Then, given any $w_1 := (v_1, d_1) \in \mathfrak{B}$ and $t_1 \in \mathbb{T}$ such that $x_1(w_1, t_1) = \xi_1$, the following holds. If $t_2 \in \mathbb{T}$ is such that there exists a $w' := (v', d') \in \mathfrak{B}$ such that $x_2(w', t_2) = \xi_2$, then we can show that there exists a $d_2 \in \pi_d\mathfrak{B}$ such that if we define

$$v_2 := v' \wedge_{t_1}^{t_2} v_1, \tag{5.89}$$

$$w_2 := (v_2, d_2), \tag{5.90}$$

we have that

$$w_2 \in \mathfrak{B}, \tag{5.91a}$$

$$x_2(w_2, t_2) = \xi_2, \tag{5.91b}$$

$$d_2(\tau) = d'(\tau), \forall \tau \le t_2, \tag{5.91c}$$

and for all $\tau > t_2$,

$$(x_1(w_1, \tau - t_2 + t_1), x_2(w_2, \tau)) \in \mathcal{R}. \tag{5.92}$$

Since $(\xi_1, \xi_2) \in \mathcal{R}$, we know that $x_2(w_1, t_1) = \xi_2$. We construct $w_2 := w' \wedge_{t_1}^{t_2} w_1$. Since x_2 is a state map, $w_2 \in \mathfrak{B}$. Thus, (5.91a) and (5.91c) are satisfied. To show that (5.91b) is also satisfied, we use the fact that x_2 is past induced. With this fact, we can infer that

$$x_2(w_2, t_2) = x_2(w', t_2) = \xi_2. \tag{5.93}$$

The fact that x_2 is Markovian implies that for all $\tau \geq \tau_2$,

$$x_2(w_1, \tau - t_2 + t_1) = x_2(w_2, \tau). \tag{5.94}$$

Equation (5.92) follows from (5.94) and the definition of $\mathcal{R}$. So far we have proven that $\mathcal{R}$ is a simulation relation of (Σ, x_1) by (Σ, x_2). The converse can be proven as follows. Take any $(\xi_1, \xi_2) \in \mathcal{R}$. Then, given any $w_2 := (v_2, d_2) \in \mathfrak{B}$ and $t_2 \in \mathbb{T}$ such that $x_2(w_2, t_2) = \xi_2$, the following holds. If $t_1 \in \mathbb{T}$ is such that there exists a $w' := (v', d') \in \mathfrak{B}$ such that $x_1(w', t_1) = \xi_1$, then we can show that there exists a $d_1 \in \pi_d \mathfrak{B}$ such that if we define

$$v_1 := v' \wedge_{t_2}^{t_1} v_2, \tag{5.95}$$
$$w_1 := (v_1, d_1), \tag{5.96}$$

we have that

$$w_1 \in \mathfrak{B}, \tag{5.97a}$$
$$x_1(w_1, t_1) = \xi_1, \tag{5.97b}$$
$$d_1(\tau) = d'(\tau), \forall \tau \leq t_1, \tag{5.97c}$$

and for all $\tau > t_2$,

$$(x_1(w_1, \tau - t_2 + t_1), x_2(w_2, \tau)) \in \mathcal{R}. \tag{5.98}$$

Since $(\xi_1, \xi_2) \in \mathcal{R}$, $x_2(w', t_1) = \xi_2$. We construct $w_1 := w' \wedge_{t_2}^{t_1} w_2$. Because of the state property of x_2, we know that (5.97a) and (5.97c) are satisfied. Furthermore, (5.97b) follows from the fact that x_1 is past-induced. Notice that $x_2(w_1, t_1) = \xi_2$. This fact, combined with the definition of $\mathcal{R}$ implies (5.98). □

Proposition 5.31 tells us that a dynamical system equipped with a past induced Markovian state map (Σ, x) is bisimilar to the pair (Σ, x'), where x' is also a past induced Markovian state map and $x \succcurlyeq x'$. Therefore, as far as the equivalence relation $\approx_{\text{bis}}$ is concerned, we can replace (Σ, x) with (Σ, x'). The potential benefit of doing so is to get a smaller state space. There exists a surjective mapping going from the state space of x to that of x'. Therefore the cardinality of the latter should be lesser than or equal to the former.

A corollary of Proposition 5.31 is that two pairs of a dynamical system equipped with two equivalent past induced Markovian state maps are bisimilar.

Corollary 5.32. Given pairs (Σ, x_1) and (Σ, x_2), where $x_1 \approx x_2$ and they are both Markovian and past induced. Then, $(\Sigma, x_1) \approx_{\text{bis}} (\Sigma, x_2)$.

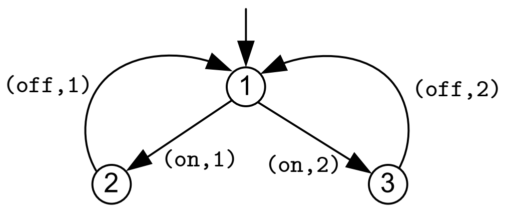
This result confirms the similar result in finite state automata, that two automata whose states are related by an isomorphism are bisimilar. Similar result is also presented in [Sch03b] for linear systems.

Literature about bisimulation in discrete event systems, linear systems and hybrid systems also explain about how to obtain the maximal state reduction using bisimulation. The maximal reduction is obtained by computing the maximal bisimulation relation between the pair (Σ, x) and itself. Algorithms for computing the maximal bisimulation relation are given in, for example, [LPS98, AHLP00, Pap03, Sch03b] for various types of systems. We shall not discuss about the algorithms, rather we focus on the system theoretic aspect of the reduction.

The maximal bisimulation relation generates partitions in $\mathcal{X}$, the state space of x, as illustrated in Figure 5.2. States in each partition are lumped and considered as a new state. The fact that the bisimulation relation is maximal implies there cannot be any coarser partitioning. Thus, maximal state reduction is achieved.

A system theoretic related question that arises is as follows. The lumping of the states explained in the previous paragraph gives us a dynamic map x', such that $x' \preccurlyeq x$. Can we guarantee that x' is also a state map? The answer to this question is no. Consider the following examples.

Example 5.33. A room has two lamps on its ceiling. Each lamp is connected to a switch that can turn the lamp on and off. However, at every time there can be at most one lamp turned on. The automaton A shown below models this situation.



The lamps are numbered 1 and 2. This automaton is obviously deterministic. We denote the alphabet of A as

$$E_A = \{(\mathtt{on}, 1), (\mathtt{on}, 2), (\mathtt{off}, 1), (\mathtt{off}, 2)\}. \tag{5.99}$$

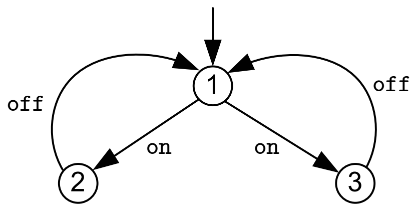
Thus, each event is labelled with a pair. The language generated by the automaton is a behavior of type $(\mathbb{Z}_+, E_A)$. The automaton is thus a dynamical system $\Sigma = (\mathbb{Z}_+, E_A, L(A))$. The states of the automaton can be thought of as a Markovian past-induced state map x. We can write the alphabet as

$$E_A = \mathbb{V} \times \mathbb{D}, \tag{5.100}$$

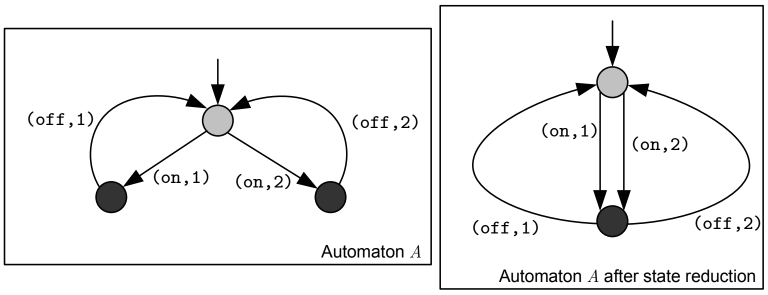
where

$$\mathbb{V} = \{\mathtt{on}, \mathtt{off}\},\ \mathbb{D} = \{1, 2\}. \tag{5.101}$$

Thus, we consider the number of the switch as being internal and the function of the switch, i.e. on and off, as being external. We can define the projection that projects the labels to $\mathbb{V}$ and obtain the following automaton, which we call A'.



The automaton A' is nondeterministic. If we compute the maximal bisimulation relation between (Σ, x) and itself, we get the following relation. The shading of the states is such that bisimilar states are given the same shading.



The partition induced by the bisimulation relation gives a dynamic map. Let us denote it by x'. We can verify that this dynamic map is not a state map of Σ. If it was a state map of Σ, the strings $(\texttt{on}, 1)(\texttt{off}, 2)$ and $(\texttt{on}, 2)(\texttt{off}, 1)$ would be in $L(A)$.

Example 5.34. Consider the following continuous time dynamical system $\Sigma = (\mathbb{R}, \mathbb{R}^3, \mathfrak{B})$. The behavior $\mathfrak{B}$ is given as all left continuous solutions of the following differential equation.

$$\frac{dy}{dt} = u, \tag{5.102a}$$

$$\frac{d}{dt}d = 0. \tag{5.102b}$$

$$\mathfrak{B} = \{(u, y, d) \mid \text{(5.102) is satisfied}\}. \tag{5.103}$$

For this dynamical system we can define a state map $x : (\mathfrak{B} \times \mathbb{R}) \to \mathbb{R}^2$ as follows.

$$x(u, y, d, t) := \begin{bmatrix} y(t) \\ d(t) \end{bmatrix}. \tag{5.104}$$

The state space $\mathcal{X}$ is thus $\mathbb{R}^2$. This state map is obviously Markovian and past induced. We define **u** and **y** as the external variables and **d** as the internal variable. Thus, $\mathbb{V} = \mathbb{R}^2$ and $\mathbb{D} = \mathbb{R}$.
The following relation $\mathcal{R} \subset \mathcal{X} \times \mathcal{X}$, defined as

$$\left(\begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \begin{bmatrix} y_1 \\ y_2 \end{bmatrix} \right) \in \mathcal{R} \Leftrightarrow (x_1 = y_1), \tag{5.105}$$

is a bisimulation relation between (Σ, x) and itself. The partition in the state space induced by $\mathcal{R}$ generates a dynamical map $x' : (\mathfrak{B} \times \mathbb{R}) \to \mathbb{R}^2$, where

$$x'(u, y, d, t) := y(t). \tag{5.106}$$

Obviously, x' is not a state map. If it was a state map, the behavior $\mathfrak{B}$ would accept any piecewise constant function as a trajectory of $\mathbf{d}$ instead of just the constant ones.

In the examples above we see that the reduced dynamic map obtained from the bisimulation relation is generally not a state map. However, it can be taken as a state map, at the expense of the fidelity of the system description with respect to the internal dynamics. That is, the reduced dynamic map can be considered as a state map of a dynamical system Σ', whose behavior is larger than that of Σ. The behavior of Σ' is larger because it also contains the trajectories resulting from concatenation that are not in the behavior of Σ.

In Example 5.33, adopting the reduced state map as a state map, we end up with a dynamical system Σ', whose behavior is larger than that of Σ. To be precise, Σ' differs from Σ in the sense that in the automaton corresponding to Σ', we do not care about the numbering of the lamps and the switches. Thus, in a sense, Σ' and Σ differ in the internal part.

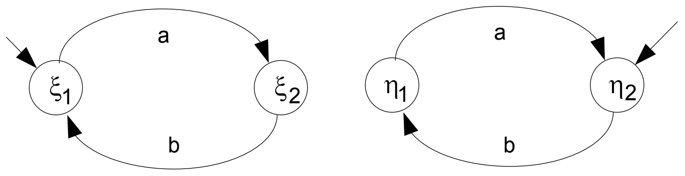
In Example 5.34, if we adopt x' as a state map, we also end up with a dynamical system Σ', whose behavior is larger than that of Σ. In Σ', we allow the trajectory of $\mathbf{d}$ to be piecewise constant, instead of just constant. Again, here we have that Σ' and Σ differ in the internal part.

5.3.4 Bisimilarity and the external behavior

We have seen that bisimilar automata generate the same language. Similarly, Van der Schaft [Sch03b] has proven that bisimilar linear systems of the form (5.20) have the same external behavior. We are going to derive a similar result for the general setting we have been discussing.

Generally, given two bisimilar pairs (Σ_1, x_1) and (Σ_2, x_2) according to the definition of bisimilarity so far, it is not necessarily true that the external behaviors of Σ_1 and Σ_2 are equal. Consider the following examples.

Example 5.35. Consider the two automata depicted below.



The alphabet $\{\mathtt{a}, \mathtt{b}\}$ is considered as the external signal space and the states of the automata represent Markovian past-induced state maps. We can easily verify that $\mathcal{R} := \{(\xi_1, \eta_1), (\xi_2, \eta_2)\}$ is a bisimulation relation, and that the two automata bisimilar. However, the external behaviors (i.e. the language generated by the automata) are not equal. In terms of regular expressions, the language generated by the automaton on the left and on the right are $(\texttt{ab})^*$ and $(\texttt{ba})^*$ respectively.

Example 5.36. Consider a dynamical system $\Sigma_1 = (\mathbb{T}, \mathbb{V} \times \mathbb{D}, \mathfrak{B}_1)$, where $\mathbb{T} = \mathbb{V} = \mathbb{D} = \mathbb{R}$ and the behavior $\mathfrak{B}_1$ is given as

$$\mathfrak{B}_1 = \{(v_1, d_1)\}, \tag{5.107}$$
$$v_1(t) = t, \forall t \in \mathbb{R}, \tag{5.108}$$
$$d_1(t) = 0, \forall t \in \mathbb{R}. \tag{5.109}$$

We can verify that $x_1 : \mathfrak{B}_1 \times \mathbb{R} \to \mathbb{R}$, where

$$x_1(v_1, d_1, t) := v_1(t), \forall t \in \mathbb{R} \tag{5.110}$$

is a state map of $\mathfrak{B}_1$.
Now, take another dynamical system $\Sigma_2 = (\mathbb{T}, \mathbb{V} \times \mathbb{D}, \mathfrak{B}_2)$, whose behavior $\mathfrak{B}_2$ is given as

$$\mathfrak{B}_2 = \{(v_2, d_2)\}, \tag{5.111}$$
$$v_2(t) = t - 1, \forall t \in \mathbb{R}, \tag{5.112}$$
$$d_2(t) = 0, \forall t \in \mathbb{R}. \tag{5.113}$$

We can verify that $x_2 : \mathfrak{B}_2 \times \mathbb{R} \to \mathbb{R}$, where

$$x_2(v_2, d_2, t) := v_2(t), \forall t \in \mathbb{R} \tag{5.114}$$

is a state map of $\mathfrak{B}_2$.
Furthermore, we can verify that the identity relation in $\mathbb{R} \times \mathbb{R}$ is a bisimulation relation between (Σ_1, x_1) and (Σ_2, x_2). Thus the two pairs are bisimilar. However, as we have seen, the external behaviors are not equal, due to time shifting.

In the examples above, we see bisimilar systems that do not share the same external behavior. Rather, the trajectories of one external behaviors are time shifted version of the other's. The reason behind this fact is that the bisimulation that we consider is time abstract. If we require that states reached at the same time instant are bisimilar, we can get external behavior equivalence under some additional technical conditions.

Definition 5.37. Given a dynamical system $\Sigma = (\mathbb{T}, \mathbb{W}, \mathfrak{B})$ and a state map x. Denote the state space of x by $\mathcal{X}$. We define the **time-indexed state space** $\mathcal{X}_t$, $t \in \mathbb{T}$ as

$$\mathcal{X}_t := \{\xi \in \mathcal{X} \mid \exists w \in \mathfrak{B} \text{ such that } x(w, t) = \xi\}. \tag{5.115}$$

Thus, the time indexed state space consists of all states that are reached at a certain time instant. With this definition we can define *synchronized bisimulation* as follows.

Definition 5.38. Given bisimilar pairs (Σ_1, x_1) and (Σ_2, x_2) with a bisimulation relation $\mathcal{R} \subset \mathcal{X}_1 \times \mathcal{X}_2$. The relation $\mathcal{R}$ is a **synchronized bisimulation** if the following relations hold for every $t \in \mathbb{T}$.

$$\forall \xi_1 \in \mathcal{X}_{1t}, \exists \xi_2 \in \mathcal{X}_{2t} \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}, \tag{5.116}$$

$$\forall \xi_2 \in \mathcal{X}_{2t}, \exists \xi_1 \in \mathcal{X}_{1t} \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}. \tag{5.117}$$

If the pairs (Σ_1, x_1) and (Σ_2, x_2) are such that there exists a synchronized bisimulation between them, then they are said to be *synchronously bisimilar*. Notice that two bisimilar pairs (Σ_1, x_1) and (Σ_2, x_2) are synchronously bisimilar if and only if the maximal bisimulation between them is a synchronized bisimulation.

Remark 5.39. The bisimulation relations given in Examples 5.35 and 5.36 are not synchronized bisimulation.

Remark 5.40. For finite state automata, it is quite easy to see that a bisimulation relation is a synchronized bisimulation if and only if the initial states are related by the bisimulation. For linear systems with observable state space representation, any bisimulation is a synchronized bisimulation. This is due to the fact that the time-indexed state space at any time instant is the state space itself.

The following result can be proven about synchronously bisimilar systems.

Theorem 5.41. Given two dynamical systems $\Sigma_1 = (\mathbb{T}, \mathbb{V} \times \mathbb{D}_1, \mathfrak{B}_1)$ and $\Sigma_2 = (\mathbb{T}, \mathbb{V} \times \mathbb{D}_2, \mathfrak{B}_2)$ equipped with state maps x_1 and x_2 respectively. If (Σ_1, x_1) and (Σ_2, x_2) are synchronously bisimilar, then the following statements hold.
(i) For any $v_1 \in \pi_v \mathfrak{B}_1$ and $\tau \in \mathbb{T}$, there exists a $v_2 \in \pi_v \mathfrak{B}_2$ such that

$$v_1(t) = v_2(t), \forall t > \tau. \tag{5.118}$$

(ii) For any $v_2 \in \pi_v \mathfrak{B}_2$ and $\tau \in \mathbb{T}$, there exists a $v_1 \in \pi_v \mathfrak{B}_1$ such that (5.118) holds.

Proof. The proof is straightforward from the definition of bisimulation (Definition 5.21). □

Theorem 5.41 tells us that synchronously bisimilar systems have almost equal external behaviors. To be precise, they share the same suffix behavior (see Definition 4.32). If the time axis $\mathbb{T}$ has a minimal element t_0, then the suffix of the external behaviors after time t_0 are equal. For finite state automata, this implies that the generated languages are equal.

If the time axis $\mathbb{T}$ does not have a minimal element and the external behaviors are *complete*, then bisimilarity also implies that the external behaviors are equal. This is the case, for example, for linear systems. A behavior $\mathfrak{B}$ of type $(\mathbb{T}, \mathbb{W})$ is complete if for any trajectory $w \in \mathbb{W}^{\mathbb{T}}$, which is not in $\mathfrak{B}$, there exists a finite time interval $\Delta := [\tau, \tau'] \subset \mathbb{T}$ such that

$$w(t)|_{t\in\Delta} \notin \mathfrak{B}|_{t\in\Delta}. \tag{5.119}$$

This means, if for any finite time interval we cannot distinguish a trajectory w from trajectories in $\mathfrak{B}$, then w must be in $\mathfrak{B}$.

Example 5.42. Define a family of functions $p(t;\tau)$, parameterized by τ as follows.

$$p(\cdot;\tau) : \mathbb{R} \to \mathbb{R}_+, \tag{5.120}$$

$$p(t;\tau) := \left\{ \begin{array}{ll} e^{-t}, & t > \tau \\ e^{-\tau}, & t \leq \tau \end{array} \right. . \tag{5.121}$$

Consider two behaviors $\mathfrak{B}_1$ and $\mathfrak{B}_2$ defined as

$$\mathfrak{B}_1 := \{p(t;\tau) \mid \tau \leq 0\}, \tag{5.122}$$

$$\mathfrak{B}_2 := \mathfrak{B}_1 \cup \{e^{-t}\}. \tag{5.123}$$

Notice that $\mathfrak{B}_1$ is not complete. Therefore, although $\mathfrak{B}_1$ and $\mathfrak{B}_2$ share the same suffix behavior, yet $\mathfrak{B}_1 \neq \mathfrak{B}_2$.

In the preceding discussion we have learned that bisimilar linear systems in state space representation, where bisimulation is in the sense of Definition 5.21 and the state space representation is observable, have equivalent external behavior. We also have learned that, generally, systems with equivalent external behaviors are not bisimilar. This is a well known fact, for example, in the case of finite state automata or hybrid systems [LPS98].

We close this section by proving that for linear systems of the form (5.20), two systems are bisimilar if and only if their external behaviors are equal.

Consider the following equation

$$\frac{dx}{dt} = Ax + Bu + Gd, \tag{5.124a}$$

$$y = Cx, \tag{5.124b}$$

with (A, C) observable. We can write a kernel representation corresponding to this equation as follows.

$$\begin{bmatrix} \frac{d}{dt}I - A & -B & 0 & -G \\ C & 0 & -I & 0 \end{bmatrix} \begin{bmatrix} x \\ u \\ y \\ d \end{bmatrix} = 0. \tag{5.125}$$

Since the states are observable, we can always transform the representation into the following form.

$$\begin{bmatrix} 0 & P & Q & R \\ I & T_1 & T_2 & T_3 \end{bmatrix} \left(\frac{d}{dt}\right) \begin{bmatrix} x \\ u \\ y \\ d \end{bmatrix} = 0, \tag{5.126}$$

where $\begin{bmatrix} P & Q & R \end{bmatrix}$ is a full row rank matrix.

We define the following behavior

$$\mathfrak{B}_{\text{full}} \in \overrightarrow{\mathfrak{L}}_{\mathtt{c}}^{\mathtt{x+u+y+d}}, \tag{5.127}$$

$$\mathfrak{B}_{\text{full}} := \{(x, u, y, d) \mid \text{(5.126) is satisfied}\}. \tag{5.128}$$

The behavior obtained by eliminating the state $\mathbf{x}$ from $\mathfrak{B}_{\text{full}}$ can be represented as follows.

$$\mathfrak{B} \in \overrightarrow{\mathfrak{L}}_{\mathtt{c}}^{\mathtt{u+y+d}}, \tag{5.129a}$$

$$\mathfrak{B} = \left\{ (u, y, d) \mid \begin{bmatrix} P & Q & R \end{bmatrix} \left(\frac{d}{dt}\right) \begin{bmatrix} u \\ y \\ d \end{bmatrix} = 0 \right\}. \tag{5.129b}$$

The elimination of the state variables is an exact elimination (see Chapter 3). The state variables $\mathbf{x}$ can be regarded as the canonical minimal state map of $\mathfrak{B}$.

$$x(u, y, d, t) = T_1 \left(\frac{d}{dt}\right) u + T_2 \left(\frac{d}{dt}\right) y + T_3 \left(\frac{d}{dt}\right) d. \tag{5.130}$$

We can see that the state map is both Markovian and past induced.

To prove that behaviors of the form (5.129) equipped with state map (5.130) are bisimilar if their external behaviors are equal, we need the following lemma.

Lemma 5.43. Given two dynamical systems $\Sigma_i = (\mathbb{R}, \mathbb{R}^{\mathtt{u+y+d}_i}, \mathfrak{B}_i)$, $i = 1, 2$. The behavior $\mathfrak{B}_i$ is defined as

$$\mathfrak{B}_i \in \overrightarrow{\mathfrak{L}}_{\mathtt{c}}^{\mathtt{u+y+d}_i}, \tag{5.131a}$$

$$\mathfrak{B}_i = \left\{ (u, y, d) \mid \begin{bmatrix} P_i & Q_i & R_i \end{bmatrix} \left(\frac{d}{dt}\right) \begin{bmatrix} u \\ y \\ d \end{bmatrix} = 0 \right\}, \tag{5.131b}$$

$i = 1, 2$. The variables $\mathbf{u}$ and $\mathbf{y}$ are considered the external variables, and $\mathbf{d}$ the internal variable. Define α_i, $i = 1, 2$, as the Nerode state construction of $\mathfrak{B}_1$ and $\mathfrak{B}_2$ respectively. Also, define π_v as the projection that eliminates the variable $\mathbf{d}_i$ from $\mathfrak{B}_i$, $i = 1, 2$. If $\pi_v \mathfrak{B}_1 = \pi_v \mathfrak{B}_2$, then $(\Sigma_1, \alpha_1) \approx_{\text{bis}} (\Sigma_2, \alpha_2)$.

Proof. Recall that the Nerode state construction α of a behavior $\mathfrak{B}$ of type $(\mathbb{W}, \mathbb{T})$ is defined as

$$\alpha(w, \tau) = w(t)|_{t \leq \tau}. \tag{5.132}$$

Thus, each state is an element of a function space. Denote the state space of α_1 and α_2 by $\mathcal{X}_1$ and $\mathcal{X}_2$ respectively. We need to construct a bisimulation relation $\mathcal{R}$ between (Σ_1, α_1) and (Σ_2, α_2). We construct $\mathcal{R}$ as follows.

$$\mathcal{R} \subset \mathcal{X}_1 \times \mathcal{X}_2, \tag{5.133a}$$

$$(\xi_1, \xi_2) \in \mathcal{R} :\Leftrightarrow (\pi_v \xi_1 = \pi_v \xi_2). \tag{5.133b}$$

The right hand side of (5.133b) is an abuse of notation. By $(\pi_v \xi_1 = \pi_v \xi_2)$ we mean that the external part of the states are equal. Since the external behaviors of the systems are equal, we know that

$$\forall \xi_1 \in \mathcal{X}_1, \exists \xi_2 \in \mathcal{X}_2 \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}, \tag{5.134}$$

$$\forall \xi_2 \in \mathcal{X}_2, \exists \xi_1 \in \mathcal{X}_1 \text{ such that } (\xi_1, \xi_2) \in \mathcal{R}. \tag{5.135}$$

Further, if we take any $(\xi_1, \xi_2) \in \mathcal{R}$. Then, given any $w_1 := (u_1, y_1, d_1) \in \mathfrak{B}_1$ and $t_1 \in \mathbb{R}$ such that $\alpha_1(w_1, t_1) = \xi_1$, the following holds. If $t_2 \in \mathbb{R}$ is such that there exists a $w' := (u', y', d') \in \mathfrak{B}_2$ such that $\alpha_2(w', t_2) = \xi_2$, then we can show that there exists a $d_2 \in \pi_d \mathfrak{B}_2$ such that if we define

$$u_2 := u' \wedge_{t_1}^{t_2} u_1, \tag{5.136a}$$

$$y_2 := y' \wedge_{t_1}^{t_2} y_1, \tag{5.136b}$$

$$w_2 := (v_2, d_2), \tag{5.136c}$$

we have that

$$w_2 \in \mathfrak{B}_2, \tag{5.137a}$$

$$\alpha_2(w_2, t_2) = \xi_2, \tag{5.137b}$$

$$d_2(\tau) = d'(\tau), \forall \tau \leq t_2, \tag{5.137c}$$

and for all $\tau > t_2$,

$$(\alpha_1(w_1, \tau - t_2 + t_1), \alpha_2(w_2, \tau)) \in \mathcal{R}. \tag{5.138}$$

First of all, we see the (5.137b) is automatically satisfied by the construction (5.136). The fact that $(\xi_1, \xi_2) \in \mathcal{R}$ implies that

$$t_1 = t_2, \tag{5.139a}$$

$$u_1(t) = u'(t), \forall t \leq t_1, \tag{5.139b}$$

$$y_1(t) = y'(t), \forall t \leq t_1. \tag{5.139c}$$

Therefore, we also have that

$$u_2 = u_1, \tag{5.140}$$

$$y_2 = y_1. \tag{5.141}$$

Hence, (5.138) is also satisfied. Let us define

$$\mathfrak{D} := \{d \mid (u_1, y_1, d) \in \mathfrak{B}_2\}. \tag{5.142}$$

Since the external behaviors of the two systems are equal, we know that $\mathfrak{D}$ is not empty. Furthermore, because of (5.139b) and (5.139c), we also know that there exists a $\tilde{d}_2 \in \mathfrak{D}$ such that

$$\tilde{d}_2(\tau) = d'(\tau), \forall \tau \leq t_2. \tag{5.143}$$

Therefore we can use $d_2 := \tilde{d}_2$ and satisfy (5.137a) and (5.137c). So far we have demonstrated that the relation $\mathcal{R}$ is a simulation of (Σ_1, α_1) by (Σ_2, α_2). However, showing that the converse also holds is analogous to the proof above. $\square$

Using Lemma 5.43 we can prove the following theorem.

Theorem 5.44. Given two dynamical systems $\Sigma_i = (\mathbb{R}, \mathbb{R}^{\mathtt{u}+\mathtt{y}+\mathtt{d}_i}, \mathfrak{B}_i)$, $i = 1, 2$. The behavior $\mathfrak{B}_i$ is defined as

$$\mathfrak{B}_i \in \overrightarrow{\mathfrak{L}}_{\mathrm{c}}^{\mathtt{u}+\mathtt{y}+\mathtt{d}_i}, \tag{5.144a}$$

$$\mathfrak{B}_i = \left\{ (u, y, d) \mid \left[\begin{array}{ccc} P_i & Q_i & R_i \end{array} \right] \left(\frac{d}{dt} \right) \left[\begin{array}{c} u \\ y \\ d \end{array} \right] = 0 \right\}, \tag{5.144b}$$

$i = 1, 2$. The variables $\mathbf{u}$ and $\mathbf{y}$ are considered the external variables, and $\mathbf{d}$ the internal variable. We equip the systems with Markovian past induced state maps x_i, where

$$x_i(w_i, t) := T_i \left(\frac{d}{dt} \right) w_i, \forall w_i \in \mathfrak{B}_i, t \in \mathbb{R}, \tag{5.145}$$

for some polynomial matrix T_i, $i = 1, 2$. Also, define π_v as the projection that eliminates the variable $\mathbf{d}_i$ from $\mathfrak{B}_i$, $i = 1, 2$. If $\pi_v \mathfrak{B}_1 = \pi_v \mathfrak{B}_2$, then $(\Sigma_1, x_1) \approx_{\mathrm{bis}} (\Sigma_2, x_2)$.

Proof. From Lemma 5.43 we know that the dynamical systems equipped with their Nerode state construction α_i are bisimilar. Further, since $\alpha_i \succcurlyeq x_i$, for $i = 1, 2$, we can use Proposition 5.31 to infer that

$$(\Sigma_i, \alpha_i) \approx_{\mathrm{bis}} (\Sigma_i, x_i), i = 1, 2. \tag{5.146}$$

Finally, by using the transitivity property of $\approx_{\mathrm{bis}}$ (Proposition 5.25), we conclude that

$$(\Sigma_1, x_1) \approx_{\mathrm{bis}} (\Sigma_2, x_2). \tag{5.147}$$

□

One consequence of Theorem 5.44 is that for linear systems of the form (5.124) with observable state space, two systems are bisimilar if and only if their external behaviors are equal. Here, bisimulation is in the sense of Definition 5.21. That is, there exists a relation $\mathcal{R}$ between the state space of the systems that satisfies the conditions in Definition 5.21. Since Definition 5.21 is a generalization of Definition 5.19, the relation $\mathcal{R}$ also satisfies the conditions in Definition 5.19, except for the fact that we do not explicitly require that $\mathcal{R}$ is a linear subspace.

To extend the result to the case where the state space representation is not necessarily observable, we do the following. We use a result in linear systems theory that any unobservable state space representation of the form (5.124) can be brought to the form shown in (5.148), by means of invertible transformation of the states. See for example, Corollary 5.3.14 in [PW98].

$$\dot{x}^{\mathrm{obs}} = \tilde{A}_{11} x^{\mathrm{obs}} + \tilde{B}_1 u + \tilde{G}_1 d, \tag{5.148a}$$

$$\dot{x}^{\mathrm{non}} = \tilde{A}_{21} x^{\mathrm{obs}} + \tilde{A}_{22} x^{\mathrm{non}} + \tilde{B}_2 u + \tilde{G}_2 d, \tag{5.148b}$$

$$y = \tilde{C} x^{\mathrm{obs}}, \tag{5.148c}$$

with $(\tilde{A}_{11}, \tilde{C})$ an observable pair. Here we can see that the transformation split the states into the observable and unobservable parts.

It can be verified that eliminating x^{non} can be done by removing (5.148b) from (5.148). That is, (5.148a) and (5.148c) give us a representation of the behavior with respect to u, y, d, and x^{obs}. Notice that by doing so we obtain an observable state space representation of the same behavior. Therefore x^{obs} is a state map of the behavior. We denote the observable subspace of the state space as $\mathcal{X}^{\mathrm{obs}}$.

From Theorem 5.44 we already know that if two linear systems of the form (5.124) have equal external behaviors, there exists a bisimulation relation between the observable part of the state spaces $\mathcal{X}_1^{\mathrm{obs}}$ and $\mathcal{X}_2^{\mathrm{obs}}$. If we denote this relation by $\mathcal{R}$, we can formulate the extension of $\mathcal{R}$ to cover the whole state space as follows. Denote the extended relation by $\tilde{\mathcal{R}}$, then for any $\xi_i \in \mathcal{X}_i$, $i = 1, 2$,

$$(\xi_1, \xi_2) \in \tilde{\mathcal{R}} :\Leftrightarrow (\xi_1^{\mathrm{obs}}, \xi_2^{\mathrm{obs}}) \in \mathcal{R}. \tag{5.149}$$

The symbols ξ_1^{obs} and ξ_2^{obs} denote the projection of ξ_1 and ξ_2 to $\mathcal{X}_1^{\mathrm{obs}}$ and $\mathcal{X}_2^{\mathrm{obs}}$ respectively.

In [Sch03b] there is an example to demonstrate that external behavior equivalence of systems of type 5.124 does not imply bisimilarity.

Example 5.45. (taken from [Sch03b]) Consider the following two systems

$$\Sigma_1 : \begin{array}{ll} \frac{dx_1}{dt} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x_1 + \begin{bmatrix} 0 \\ 1 \end{bmatrix} d_1, & x_1 \in \mathbb{R}^2, d_1 \in \mathbb{R} \\ y_1 = \begin{bmatrix} 1 & 0 \end{bmatrix} x_1, & y_1 \in \mathbb{R} \end{array}, \tag{5.150}$$

$$\Sigma_2 : \begin{array}{ll} \frac{dx_2}{dt} = d_2, & x_2 \in \mathbb{R}, d_2 \in \mathbb{R} \\ y_2 = x_2, & y_2 \in \mathbb{R} \end{array}. \tag{5.151}$$

In the example, the behavior of the systems are assumed to be the strong solution to (5.124). The external variable is denoted by $\mathbf{y}$ and the internal variable $\mathbf{d}$. Notice that the first system is a double integrator, with $\mathbf{d}$ as the input and $\mathbf{y}$ the output. The second system is a (single) integrator. These systems have equal external variables. However, the algorithm presented in [Sch03b] shows that they are not bisimilar.

The result that we have proven here does not contradict the example. The explanation is that the state construction in (5.150) and (5.151) are not state maps for the strong behaviors.

The discrete time counterpart of Theorem 5.44 generally does not hold. This is because the discrete time counterpart of Lemma 5.43, which is used in constructing the proof of Theorem 5.44, generally does not hold. In fact, we can pinpoint where the proof fails in the discrete time part. In the proof of Lemma 5.43 we define a set $\mathfrak{D}$ and assert the existence of a $\tilde{d}_2 \in \mathfrak{D}$ such that (5.143) holds. This is generally not true for discrete time linear systems. As a consequence, we can construct an example of discrete time linear systems with equal external behaviors, which are not bisimilar. The example is the discrete time version of Example 5.45.

Example 5.46. Consider the following two systems

$$\Sigma_1 : \begin{array}{ll} x_1(k+1) = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} x_1(k) + \begin{bmatrix} 0 \\ 1 \end{bmatrix} d_1(k+1), & x_1 \in \mathbb{R}^2, d_1 \in \mathbb{R} \\ y_1(k+1) = \begin{bmatrix} 1 & 0 \end{bmatrix} x_1(k), & y_1 \in \mathbb{R} \end{array}, \tag{5.152}$$

$$\Sigma_2 : \begin{array}{ll} x_2(k+1) = d_2(k+1), & x_2 \in \mathbb{R}, d_2 \in \mathbb{R} \\ y_2(k+1) = x_2(k), & y_2 \in \mathbb{R} \end{array}. \tag{5.153}$$

The external variable is denoted by $\mathbf{y}$ and the internal variable $\mathbf{d}$. These systems have equal external variables. However, we can prove that they are not bisimilar. We shall do it by contradiction. Suppose that a bisimulation relation $\mathcal{R}$ exists between $\mathcal{X}_1$ and $\mathcal{X}_2$. Consequently the set $\{\xi \in \mathcal{X}_1 \mid (\xi, 0) \in \mathcal{R}\}$ is not empty. Let $(a,b) \in \mathcal{X}_1$ be such that $((a,b),0) \in \mathcal{R}$. Necessarily, $a = 0$. However, for any $b \in \mathbb{R}$, $((0,b),0)$ cannot be in $\mathcal{R}$, since departing from $(0,b) \in \mathcal{X}_1$, the external trajectory of the first system is necessarily 0 and then b at the first two time instances. Meanwhile, the trajectory of the second system departing from $0 \in \mathcal{X}_2$ is free after the first time instance. Thus the first system cannot simulate the second.

5.4 Summary

In this chapter we discuss about the notion of external equivalence of systems. We start with the notion of external behavior equivalence. An external behavior of a system is obtained by factoring the signal space $\mathbb{W}$ as $\mathbb{V} \times \mathbb{D}$. The latter are the external and internal signal spaces respectively. The behavior obtained by projecting the behavior of the system to the space $\mathbb{V}$ is called the external behavior.

Another notion of external behavior equivalence is bisimulation. The concept of bisimulation originates in the field of theoretical computer science. A bisimulation is a relation with certain properties, defined between the state space of two systems. Two systems are called bisimilar if there exists a bisimulation relation between them. We review bisimulation in the sense of finite state automata. Bisimulation for other class of systems that has appeared in the literature is reviewed as well. A well known result about the relation between bisimilarity and the external behavior equivalence, as external equivalence of systems, is that the former is stronger than the latter.

We adopt the notion of bisimulation and define it for the class of dynamical systems equipped with a state map. That is, given dynamical systems $\Sigma_i = (\mathbb{T}, \mathbb{V} \times \mathbb{D}_i, \mathfrak{B}_i)$, $i = 1, 2$, and state maps x_1 and x_2 of the systems respectively, we define the notion of bisimulation between (Σ_1, x_1) and (Σ_2, x_2). We prove that for Markovian past induced state maps, bisimilarity is an equivalence relation. We also discuss the idea of state reduction in this general setting.

In the last section, we discuss about the relation between bisimilarity and external behavior equivalence. It is proven that with some assumption, bisimilarity implies external behavior equivalence. Another interesting result that is derived is that for a class of continuous time linear systems, which has been treated in the literature before [Pap03, Sch03b], bisimilarity is equivalent to external behavior equivalence. However, for the discrete time counterpart of this class of systems, the result does not hold.

6 Conclusions and recommendations

"Whereof one cannot speak, thereof one must be silent." - Ludwig Wittgenstein

6.1 Conclusions

In this book we discuss a unified systems theoretic framework for modeling and analyzing dynamical systems. The framework is based on the behavioral approach to systems theory, pioneered by Jan Willems [Wil86a, Wil86b, Wil87]. The paradigm of this approach is to identify systems with their behaviors. The behavior of a system is the collection of trajectories that the system can exhibit / undertake. This is discussed in Chapter 2. We also discuss a few classes of dynamical systems, namely linear systems, discrete event systems and hybrid systems from the behavioral point of view.

One of the main themes of the book, namely *interconnection*, is discussed in Chapter 3. We explain how interconnection or synchronization of systems mentioned in the previous paragraph can be cast as behavior interconnection. We start with the most basic kind of interconnection, that is, the *full interconnection*. The full interconnection is formally defined as interconnection of behaviors of equal types. Loosely speaking, in full interconnection, full information about the behaviors involved is shared. This is evident when full interconnection is interpreted, for example, for linear systems and discrete event systems. In these cases, full interconnection means that the behaviors synchronize on all variables and events respectively.

Another kind of interconnection that we discuss is *partial interconnection*. In a partial interconnection, the behaviors involved do not share full information. The fact that some information is hidden in the interconnection is represented by a behavioral *projection*. A projection is a surjective map that maps a behavior to another one, possibly of a different type. After a projection, some information about the behavior can be lost, since it is possible that multiple trajectories are mapped to the same trajectory in the domain.

By defining different projections, we can decide what information to hide and what information to retain. Thus, different projections can retain different information. Projections that retain the same information are said to be equivalent. We also define a partial ordering for projections. Loosely speaking, a projection ϕ is less than another projection γ if the information in ϕ can be inferred from that retained in γ. When this is the case, we also say that ϕ is *observable* from γ.

Another important concept that we introduce is that of *dynamic maps*. Dynamic maps are surjective maps that maps the Cartesian product of a behavior $\mathfrak{B}$ and its time axis $\mathbb{T}$ to a domain. Dynamic maps can be regarded as projections. We also define a partial ordering for dynamic maps. Furthermore, we show that the class of dynamic maps defined on a behavior has a *lattice* structure.

We define several subclasses of dynamic maps, based on some certain properties they have. The most important subclass is the class of so called *state maps*. State maps are dynamic maps that possess the so called *state property*.

In Chapter 4 we discuss *control problems* seen as interconnections. The basic idea is that a control problem can be seen as follows. Given a system that we call the *plant*. The problem is to find another system, called the *controller*, that when interconnected with the plant yields a system with a desired property. We also specify a projection that determines how the controller is interconnected with the plant.

An important issue related to every control problem is that of *achievability*. Given a control problem, the specification is said to be achievable if there exists a controller that yields it. That is, the interconnection between the controller and the plant achieves the specification. We presented a construction of the so called *canonical controllers*. There are two kinds of canonical controllers. The first canonical controller is defined as a set theoretic construction. It achieves the desired specification if and only if it is achievable at all. The second canonical controller is defined as an interconnection. For plant with homogeneity property, this controller achieves the desired specification if and only if it is achievable at all.

In some cases, on top of achieving the desired specification, the controller also has to satisfy some additional constraints. We discuss two kinds of constraints. The first one is the so called *compatibility constraint*. The idea behind this constraint is the interpretation that interconnection is seen as something realized on previously existing systems. With this interpretation, it is necessary that all possible pasts of the systems involved in the interconnection can be continued. By past we mean the portion of the trajectories prior to the realization of the interconnection.

We define two notions of compatibility, strong and weak. Further, we show that for linear systems, the notion of compatibility and weak compatibility are related to the notion of *regular* interconnection and *regular feedback* interconnections [PW98]. In fact, we show that achievability with a weakly compatible controller is equivalent to achievability with a regular controller. Conditions for achievability with a regular controller are published in [BT02].

The other constraint that we discuss is *input-output partitioning constraint*. This constraint concerns linear systems. The basic idea is as follows. In some situation, there are some variables that are used in the interconnection between the plant and the controller, which are inherently outputs of the plant. By this we mean, for example, variables that are related to measurement or sensor information. To be realistic, the controller should not impose any constraint directly on these variables. This concept is elaborated mathematically in Section 4.3. We also derive an algorithm to construct a controller that achieves the desired specification, while respecting the compatibility constraint (in the form of regularity) and the input-output partitioning constraint. The algorithm yields a solution, if and only if such a controller exists at all.

Using the same mathematical results as the ones used to construct the controller described in the previous paragraph, we also give an algebraic solution to the so called *control problem with minimal interaction* for linear systems. The problem is about finding a controller that achieves the desired specification using as few control variables as possible.

Chapter 5 is devoted to the issue of external equivalence of systems. We introduce the notion of *external behavior equivalence*. First, we factor the signal space of the behavior into an internal and external part. The external behavior of a system is simply its behavior projected to the external part of the signal space.

We also discuss the concept of bisimulation as a notion of external equivalence. Bisimulation is a concept that originates from the field of discrete event systems and concurrent processes. We also review the concept of bisimulation particularly for discrete event systems and continuous time linear systems.

It is generally known that bisimilarity is a stronger notion of systems equivalence, than external behavior equivalence. However, in this thesis it is proven that for continuous time linear systems, the two notions coincide. This result strengthens a result by Van der Schaft [Sch03b, Sch04a], that for that class of systems, bisimilarity is equivalent to mutual simulation. Mutual simulation is another notion of equivalence, whose strength is between bisimulation and external behavior equivalence.

6.2 Contributions of the thesis

The contributions of the thesis can be summarized as follows.

- In this thesis we formulate a unified systems theoretic framework for modeling and analyzing dynamical systems. The framework is based on the behavioral approach to systems theory and is generalized to include systems such as linear systems, discrete event systems and hybrid systems.

- In the framework, interconnection of systems is handled as interconnection of behaviors. We also introduce the concept of behavioral projection to enable handling partial interconnections in a general setting. Furthermore, a partial ordering of projections is defined, and its relation to the concept of observability is presented.

- Another tool of systems analysis developed in this book is dynamic maps. We introduce the concept of dynamic maps as well as some subclasses of dynamic maps. One of the subclasses is the class of state maps, which is a general concept that captures the notion of state in the usual sense.
- We also introduce a partial ordering and lattice structure for dynamic maps and show how the properties of the subclasses of dynamic maps, which are discussed in the previous point, are related to this lattice.
- The concept of canonical controllers is analyzed in a general set-theoretic manner. It is then related with the issue of achievability of a specification in control problems.
- We present the notion of compatibility constraint and show how it is related to the already known concepts of regular and regular feedback interconnections.
- We also present the notion of input-output partitioning constraint for control problem of linear systems. An algorithm to synthesize a regular controller satisfying the constraint is given.
- Using the same mathematical tool as in the previous point, we treat the problem of control with minimal interaction for linear systems. The problem is about finding a controller that achieves the desired specification using as few control variables as possible. An algorithm is presented, with which the problem is transformed into a simple algebraic-combinatoric problem. We also derive a tight lower bound on the number of control variables.
- External equivalence of systems is considered, in the form of external behavior equivalence and bisimulation. After reviewing bisimulation in the sense of discrete event systems and linear systems, we present bisimulation in a general behavioral setting. Analysis is performed by exploiting the lattice structure of the dynamic maps. One important result that we obtain is that for continuous time linear systems, bisimilarity is equivalent to external behavior equivalence.

6.3 Recommendations for further research

There are several issues that can be recommended for further research. We shall review them one by one.

6.3.1 More common structure for general behaviors

Although we have seen the application of the theory developed in this thesis for several classes of dynamical systems, i.e., linear systems, discrete event systems and hybrid systems, it is apparent that these systems are structured to different extents. By this, we mean the following.

Linear systems are equipped with a realization theory that tells us whether a set of trajectories can be represented as a behavior in one of the classes that we discuss in Section 2.3 [Wil86b, PM99]. Analysis in the general framework is typically done in a set-theoretic fashion. It is therefore an important issue to know when a behavior that is defined as a set-theoretic construct actually falls into the class of interest. The discussion about proper eliminability of variables in Chapter 3 is an example of a discussion on this issue.

Behaviors of linear systems also admit a nice algebraic framework, namely that of polynomial matrices. In this framework analysis and synthesis of systems can be performed. This fact is exemplified in Chapter 4, where we present some algorithms for controller synthesis. Furthermore, the algebraic structure of the linear systems admits some canonical forms, for example, the minimal representation and the Smith form. The availability of such canonical forms enables us to translate systems theoretic properties into properties of the representation. For example, controllability and observability of linear systems can be tested by bringing the system to its Smith form [PW98].

Remark 6.1. Some algebraic-geometric structure for nonlinear dynamical systems, similar to that of linear systems also exists, for example in [Nv90, Sas99, Pom01a, Pom01b]. This has been exploited, for example, in the research about bisimulation of nonlinear dynamical systems [Sch03b], and other related research [PS02, TP03].

Discrete event systems, particularly finite state automata, also have a realization theory. It is manifested in the fact that regular languages are representable by finite state automata. There is also an algebraic framework for discrete event systems, namely *process algebra* [BPS01]. However, finite state automata do not yet possess an algebraic structure as extensive as that of linear systems. For example, there is not (yet) any translation of the systems theoretic properties, such as controllability and observability, in the sense discussed in this book, to the properties of the representation of the automata.

Hybrid systems are even less structured. For this large class, there is not yet any realization theory that relates the collection of trajectories to the representation as a hybrid behavioral automaton. There is, however, ongoing research to this direction, for example, [Pet04]. As to the algebraic structure for hybrid systems, there have been efforts to extend process algebra, which is a tool originating in the field of computer science, to cover hybrid systems [Cui04, BKU04, BKU05]. This approach offers an axiomatic system, in which notions such as equivalence of processes can be formally defined.

Based on the exposition above, it would be favorable to conduct research in the direction of establishing a realization theory for hybrid systems and a general algebraic framework that can be used to analyze all three classes of systems mentioned above. As a starting point for the realization theory, one might want to take a look at the literature, where a general and abstract realization theory is presented, for example [Wil79, Sch84, Son98]. For the algebraic framework, it is desired to have a structure that allows transformation of a given representation to equivalent representations (like unimodular matrices do), and admits some canonical forms, on which systems theoretical properties can be easily checked. To this end, hybrid process algebra can be used as a starting point.

6.3.2 Compatibility and nonblocking interconnection

The constrained control problems discussed in Chapter 4 are mainly about linear systems. Consider the compatibility constraint. The idea behind compatibility constraint is to avoid deadlock because of the interconnection. Deadlocks in the models of continuous time physical systems are typically undesirable. However, deadlocks for discrete event systems, in the sense that an execution does not continue indefinitely could be desirable. What is usually undesirable is if the execution deadlocks in a non-marked state. This is typically called *blocking* [CL99]. An interconnection of automata that does not have blocking phenomenon is called a *nonblocking interconnection*. Nonblockingness typically appears as a constraint in the supervisory control of discrete event systems.

We see that compatible interconnection is related to nonblocking interconnection. Nevertheless, they are different notions. It would be interesting to study the nonblocking issue in a general framework. A result in this direction might be useful, for example, in dealing with control problems involving hybrid systems. Perhaps, we can use the framework developed in Chapter 5, where we deal with systems equipped with state maps. This framework is already close to the transition systems/automata framework of the discrete event systems.

6.3.3 Input-output partitioning constraint and input enabledness

In Section 4.3 we touched upon some loose connection between the fact that a variable is an input to a linear system and the fact that an event is input enabled in a discrete event system. An event `inp` is input enabled means that from any state of the automaton there is always a transition labelled with `inp`. It would be interesting to study the following questions. Is it possible to treat input enabledness in the framework that we have developed in this book? If yes, what would be the analog of it for other classes of systems?

6.3.4 Regular feedback achievability of linear systems

In Chapter 4, we show that for linear systems, compatibility is equivalent to the fact that an interconnection is a linear feedback interconnection. We also prove that weakly compatible achievability is equivalent to regular achievability, whose necessary and sufficient conditions are known [BT02, Bel03]. However, necessary and sufficient conditions for a specification in a given control problem to be achievable with a regular feedback controller is still an open problem.

6.3.5 Further studies on bisimulation

The study of bisimulation in a general behavioral setting is done in Chapter 5. It would be interesting to conduct further studies in this direction. In doing so, we might be able to answer some interesting questions, for example:

- We know that for continuous time linear systems, bisimilarity is equivalent to external behavior equivalence (see Subsection 5.3.4). Is it possible to generalize this result?
- We also know that for discrete time linear systems, bisimilarity is not equivalent to external behavior equivalence (see Subsection 5.3.4, particularly Example 5.46). What extra conditions can we impose so that this fact is also true for discrete time linear systems?
- Can we treat weak bisimulation [Mil89, Her02] in a behavioral setting and how?
- How do we formulate and solve control problems where systems equivalence is understood as bisimulation?

Bibliography

[ABGS91] C. Alvarez, J. L. Balcazar, J. Gabarro, and M. Santha. Parallel complexity in the design and analysis of concurrent systems. In *PARLE91*, Lecture Notes in Computer Science. Springer-Verlag, 1991.

[ACHH93] R. Alur, C. Courcoubetis, T. A. Henzinger, and P. H. Ho. *Hybrid automata: an algorithmic approach to the specification and verification of hybrid systems*, volume 736 of *Lecture Notes in Computer Science*, pages 209–229. Springer, 1993.

[AHLP00] R. Alur, T. A. Henzinger, G. Lafferriere, and G. J. Pappas. Discrete abstractions of hybrid systems. *Proceedings of the IEEE*, 88:971–984, 2000.

[Alu95] R. Alur *et. al.* The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[Arn94] A. Arnold. *Finite transition systems*. Prentice Hall International series in computer science. Prentice Hall Int'l, Paris, 1994.

[BBK87] J. C. M Baeten, J. A. Bergstra, and J. W. Klop. Ready trace semantics for concrete process algebra with priority operator. *Computer Journal*, 30(6):498–506, 1987.

[BBM98] M. S. Branicky, V. S. Borkar, and S. K. Mitter. A unified framework for hybrid control: model and optimal control theory. *IEEE Trans. Automatic Control*, 43:31–45, 1998.

[Bel03] M. N. Belur. *Control in a behavioral context*. PhD thesis, University of Groningen, June 2003.

[BKO88] J. A. Bergstra, J. W. Klop, and E. R. Olderog. Readies and failures in the algebra of communicatinf processes. *SIAM Journal on Computation*, 17:1134–1177, 1988.

[BKU04] E. Brinksma, T. Krilavicius, and Y. S. Usenko. Behavioural hybrid process calculus. Technical report, submitted for publication., 2004.

[BKU05] E. Brinksma, T. Krilavicius, and Y. S. Usenko. Process algebraic approach to hybrid systems. submitted to the 16th IFAC World Congress, 2005.

[BPS01] J. A. Bergstra, A. Ponse, and S. A. Smolka, editors. *Handbook of process algebra*. Elsevier Science, 2001.

[Bro91] W. L. Brogan. *Modern control theory*. Prentice Hall International, New Jersey, 1991.

[BT02] M. N. Belur and H. L. Trentelman. Stabilization, pole placement and regular implementability. *IEEE Trans. Automatic Control*, 47:735–744, 2002.

[Büc89] J. R. Büchi. *Finite automata, their algebras and grammars*. Springer-Verlag, 1989.

[CL99] C. G. Cassandras and S. Lafortune. *Introduction to Discrete Event Systems*. Kluwer, 1999.

[CR03] P. J. L. Cuijpers and M. A. Reniers. Hybrid process algebra. Technical report, Technical University Eindhoven, The Netherlands, 2003.

[Cui04] P. J. L. Cuijpers. *Hybrid process algebra*. PhD thesis, Technical University Eindhoven, December 2004.

[DN00] J. M. Davoren and A. Nerode. Logics for hybrid systems. *Proc. of the IEEE*, 88:985 – 1010, July 2000.

[Hen96] T. A. Henzinger. The theory of hybrid automata. In *Proc. 11th Annual Symposium on Logic in Computer Science*, pages 278–292. IEEE Computer Society Press, 1996.

[Her02] Holger Hermanns. *Interactive Markov Chains*, volume 2428 of *LNCS*. Springer-Verlag, Berlin, 2002.

[HMU01] J. E. Hopcroft, R. Motwani, and J. D. Ullman. *Introduction to automata theory, languages, and computation*. Addison Wesley, 2nd edition, 2001.

[Hoa84] C. A. R. Hoare. *Communicating sequential processes*. Prentice Hall International, 1984.

[HS96] H Huttel and Sandeep Shukla. The complexity of deciding behavioural equivalences and preorders. Survey paper. Available on http://www.cs.aau.dk/~hans/Publications/pubs.html, 1996.

[HTP02] E. Haghverdi, P. Tabuada, and G. J. Pappas. Unifying bisimulation relations for discrete and continuous systems. In *Proc. Mathematical Theory of Networks and Systems*, 2002.

[HTP03] E. Haghverdi, P. Tabuada, and G. J. Pappas. Bisimulation relations for dynamical, control, and hybrid systems. submitted to *Theoretical Computer Science*, 2003.

[JPS05] A. A. Julius, J. W. Polderman, and A. J. van der Schaft. Controller with minimal interaction. Submitted to the 16th IFAC World Congress, 2005.

[JSS03] A. A. Julius, S. N. Strubbe, and A. J. van der Schaft. Control of hybrid behavioral automata by interconnection. In *Proc. IFAC Conf. Analysis and Design of Hybrid Systems*, pages 135–140, St. Malo, 2003. IFAC.

[JS03] A. A. Julius and A. J. van der Schaft. Compatibility of behavior interconnections. In *Proc. European Control Conference*, Cambridge, September 2003. IEE.

[JS04a] A. A. Julius and A. J. van der Schaft. A behavioral framework for compositionality: linear systems, discrete event systems and hybrid systems. In *Proc. of Mathematical Theory of Networks and Systems*, Leuven, July 2004.

[JS04b] A. A. Julius and A. J. van der Schaft. State maps of general behaviors, their lattice structure and bisimulation. In *Proc. Mathematical Theory of Networks and Systems*, Leuven, July 2004.

[JWBT04] A. A. Julius, J. C. Willems, M. N. Belur, and H. L. Trentelman. The canonical controllers and regular interconnection. admitted for publication in Systems and Control Letters, 2004.

[KS90] P. C. Kanellakis and S. A. Smolka. CCS expressions, finite state processes and three problems of equivalence. *Information and Computation*, 86:43–68, 1990.

[Lan92] R. Langerak. *Transformation and semantics of LOTOS*. PhD thesis, University of Twente, The Netherlands, November 1992.

[LPS98] G. Lafferriere, G. J. Pappas, and S. Sastry. Hybrid systems with finite bisimulations. In *Hybrid Systems V*, Lecture Notes in Computer Science. Springer, 1998.

[LSV01] N. A. Lynch, R. Segala, and F. W. Vaandrager. Hybrid I/O automata revisited. In *Proceedings 4th Int'l Workshop on Hybrid Systems : Computation and Control*, pages 403–417. Springer-Verlag, 2001.

[LSV03] N. A. Lynch, R. Segala, and F. W. Vandraager. Hybrid I/O automata. *Information and Computation*, 185:105–157, 2003.

[LT89] N. A. Lynch and M. R. Tuttle. An introduction to input/output automata. *CWI Quarterly*, 2:219–246, 1989.

[Mil89] R. Milner. *Communication and concurrency*. Prentice Hall International, 1989.

[MR99] T. Moor and J. Raisch. Supervisory control of hybrid systems within a behavioral framework. *Systems and Control Letters*, 38:157–166, 1999.

[Nv90] H. Nijmeijer and A. J. van der Schaft. *Nonlinear dynamical control systems*. Springer Verlag, New York, 1990.

[Oga90] K. Ogata. *Modern control engineering*. Prentice Hall International, New Jersey, 1990.

[Pap03] G. J. Pappas. Bisimilar linear systems. *Automatica*, 39:2035–2047, December 2003.

[Par81] D. Park. Concurrency and automata on infinite sequences. In *5th GI Conference on Theoretical Computer Science*, pages 167–183, Berlin, 1981. Springer-Verlag.

[Pet04] M. Petreczky. Realization theory for linear switched systems. In *Proceedings of the Mathematical Theory of Networks and Systems*, Leuven, July 2004.

[PM99] J. W. Polderman and I. Mareels. A behavioral approach to adaptive control. In J. W. Polderman and H. L. Trentelman, editors, *Mathematics of System and Control*, Festschrift on the occasion of the 60-th birthday of J.C. Willems, Groningen, 1999. 119-130.

[Pol97] J. W. Polderman. Proper elimination of latent variables. *Systems and Control Letters*, 32:261–269, 1997.

[Pom01a] J. F. Pommaret. *Partial differential control theory*, volume 1:Mathematical tools. Kluwer, Dordrecht, 2001.

[Pom01b] J. F. Pommaret. *Partial differential control theory*, volume 2:Control systems. Kluwer, Dordrecht, 2001.

[PP04] I. Pendharkar and H. K. Pillai. On a behavioral theory for nonlinear systems. In *Proceedings of the Mathematical Theory of Networks and Systems*, Leuven, July 2004.

[PS02] G. J. Pappas and S. Simic. Consistent abstractions of affine control systems. *IEEE Transactions on Automatic Control*, 47:745–756, May 2002.

[PvD04] G. Pola, A. J. van der Schaft, and M. D. Di Benedetto. Bisimulation theory for switching linear systems. To appear in the Proc. 43rd IEEE Conference on Decision and Control, 2004.

[PW98] J. W. Polderman and J. C. Willems. *Introduction to Mathematical Systems Theory: A Behavioral Approach*. Springer, New York, 1998.

[Ram87] P. J. Ramadge. Supervisory control of discrete event systems: a survey and some new results. In *Discrete event systems: models and applications*, Lecture Notes in Control and Information Sciences, pages 69–80. Springer-Verlag, 1987.

[Roc02] P. Rocha. Regular implementability of nD behaviors. In *Proceedings of Mathematical Theory of Networks and Systems*, Univ. Notre Dame, 2002.

[RW89] P. J. Ramadge and W. M. Wonham. The control of discrete event systems. *Proceedings of the IEEE*, 77:81–98, January 1989.

[RW97] P. Rapisarda and J. C. Willems. State maps for linear systems. *SIAM Journal of Contr. Optimization*, 35:1053–1091, 1997.

[RW01] P. Rocha and J. Wood. Trajectory control and interconnection of 1D and nD systems. *SIAM J. Control Optim.*, 40:107–134, 2001.

[Sas99] S. Sastry. *Nonlinear systems: analysis, stability, and control*. Springer Verlag, New York, 1999.

[Sme87] R Smedinga. Using trace theory to model discrete events. In *Discrete event systems: models and applications*, Lecture Notes in Control and Information Sciences, pages 81–99. Springer-Verlag, 1987.

[Sme89] R. Smedinga. *Control of Discrete Events*. PhD thesis, Rijksuniversiteit Groningen, 1989.

[Son98] E. D. Sontag. *Mathematical control theory: deterministic finite dimensional systems*. Texts in Applied Mathematics. Springer-Verlag, 1998.

[Tab04] P. Tabuada. Nonlinear flat systems admit finite bisimulations. In *Proc. of Mathematical Theory of Networks and Systems*, Leuven, July 2004.

[TP03] P. Tabuada and G. J. Pappas. Abstraction of hamiltonian control systems. *Automatica*, 39:2025–2033, December 2003.

[Tre99] H. L. Trentelman. A truly behavioral approch to the H_∞ control problem. In J. W. Polderman and H. L. Trentelman, editors, *Mathematics of System and Control*, Festschrift on the occasion of the 60-th birthday of J. C. Willems, pages 177–190, Groningen, 1999.

[Tre04] H. L. Trentelman. *Unsolved Problems in Mathematical Systems and Control Theory*, chapter Regular feedback implementability for linear differential behaviors, pages 44–48. Princeton University Press, 2004.

[TW99] H. L. Trentelman and J. C. Willems. H-infinity control in a behavioral context: The full information case. *IEEE Trans. Automatic Control*, 44:521–536, 1999.

[Sch84] A. J. van der Schaft. *System Theoretic Description of Physical Systems*. CWI Tract No.3. CWI, Amsterdam, 1984.

[Sch03a] A. J. van der Schaft. Achievable behavior of general systems. *Systems and Control Letters*, 49:141–149, 2003.

[Sch03b] A. J. van der Schaft. Equivalence of dynamical systems by bisimulation. submitted to IEEE Trans. Automatic Control, 2003.

[Sch04a] A. J. van der Schaft. Bisimulation of dynamical systems. In *Proceedings 7th Int'l Workshop on Hybrid Systems : Computation and Control*, Philadephia, 2004. Springer-Verlag.

[Sch04b] A. J. van der Schaft. Equivalence of hybrid dynamical systems. In *Proceedings of the Mathematical Theory of Networks and Systems*, Leuven, July 2004.

[SJ02] A. J. van der Schaft and A. A. Julius. Achievable behavior by composition. In *Proceedings 41st IEEE Conf. Decision and Control*, pages 7–12, Las Vegas, 2002. IEEE.

[SS00] A. J. van der Schaft and J. M. Schumacher. *An Introduction to Hybrid Dynamical Systems*. Springer, London, 2000.

[WBJT] J. C. Willems, M. N. Belur, A. A. Julius, and H. L. Trentelman. The canonical controller and its regularity. To appear in the Proc. IEEE Conf. Decision and Control 2003.

[WBJT03] J. C. Willems, M. N. Belur, A. A. Julius, and H. L. Trentelman. The canonical controller and its regularity. In *Proc. IEEE Conference on Decision and Control*, pages 1639–1644, Hawaii, December 2003.

[Wil79] J. C. Willems. System theoretic models for the analysis of physical systems. *Richerche di Informatica*, 10:71–106, 1979.

[Wil86a] J. C. Willems. From time series to linear systems - Part I. Finite dimensional linear time invariant systems. *Automatica*, 22:561–580, 1986.

[Wil86b] J. C. Willems. From time series to linear systems - Part II. Exact modeling. *Automatica*, 22:675–694, 1986.

[Wil87] J. C. Willems. From time series to linear systems - Part III. Approximate modeling. *Automatica*, 23:87–115, 1987.

[Wil91] J. C. Willems. Paradigms and puzzles in the theory of dynamical systems. *IEEE Trans. Automatic Control*, 36:259–294, 1991.

[Wil97] J. C. Willems. On interconnections, control and feedback. *IEEE Trans. Automatic Control*, 42:326–339, 1997.

[ZB99] A. Zavala-Rio and B. Brogliato. On the control of a one degree-of-freedom juggling robot. *Dynamics and control*, 9:67–91, January 1999.

Acknowledgements

I have been working on this thesis for about half a year now. It has not been completely easy and smooth. I am much indebted to my research supervisor, Arjan van der Schaft, as well as Jan Willem Polderman and Jan Willems, for reading carefully through the draft version of this thesis and giving me valuable comments. I particularly feel so, because I have been reading through the draft myself and I learned that it was anything but fun.

I also would like to thank the other members of my graduation committee, Hans Schumacher, Ed Brinksma, George Pappas, and Rom Langerak for agreeing to serve in the committee and reading the final draft of the thesis.

The research reported in this thesis was conducted under the auspices of the NWO through grant number 617.023.002. I would like to thank the NWO for funding me for the last four years.

Graduation and financial matters aside, I would also like to share a few words about how being a PhD student was for me. In the last four years I have been enjoying a very pleasant research atmosphere. My research supervisor, Arjan van der Schaft, is very supportive and gives his students a lot of freedom in doing their research. Most, if not all, of the work I have done was inspired by what he had been researching. Arjan, I was glad when you offered me this PhD position, and I'm still glad that I accepted the offer. I am also indebted to the other staff members of the group for sharing a friendly and collegial (and humorous) research atmosphere with me. In particular, I would like to thank Jan Willem Polderman, Gjerrit Meinsma, Hans Zwart and Michel Vellekoop for helping me when I had technical or TeX-nical problems. *Heren, bedankt*! I also want to thank the group secretary, Marja Langkamp, for taking care of the administrative matters and for the occasional discussions on nontechnical matters.

To Jan Willems, I would like to express my gratitude for the opportunity I had to work with him, for his interest in my work and also for the time we spent on long distance scientific discussions. Jan, it was such a privilege to work with you. I also thank Madhu Belur for being my *comrade-in-arms* in my research on behavioral systems theory. I hope our cooperation is as much useful to you as it is to me.

If I had to tell a reason why life as a PhD student (or AiO as we say it in Dutch) is fun, I would have to point to the other PhD students in the group. With my officemate, Stefan Strubbe, I have enjoyed a lot of discussion on technical and nontechnical matters, as well as a companionship. *Stefan, jij bent hartelijk bedankt*! Other students, ex-students, visitors, and ex-visitors, Agoes, Vishy, Ram, Javi, Ari, Emad, Hendra, Maria, Simon, and others are also 'guilty' for *the fun and comical research atmosphere* that I have enjoyed. (I wonder if this italized phrase is ever used before).

I am a member of a rather closed and secretive group called the miniCASH. Before you think that it is a malicious conspiracy to ruin the financial world with small coins (if it is possible at all), let me tell you that it is actually an informal group of 'young' researchers in hybrid systems, with whom I have shared many amiable lunches and scientific discussions. Rom, Tomas, Stefan, and Raj, thank you for everything. I enjoyed our shared time and I hope we can keep in touch.

Being an Indonesian living in a foreign country, I appreciate very much the friendship I share with the guys and girls of Indonesian students organization and other Indonesian people in Enschede. They have been my sports mates, chat mates, and keeping me well fed. *Terima kasih banyak buat semuanya*.

In a more personal tone, I would like to thank my Deppenbroek neighborhood *gang*, Agoes, Dadan, and Emad. Thank you guys for being there when I need it and for sharing the good dinners with me. I also would like to thank my friend Ilan, for the friendship we have and the good times we had.

To other friends I have made and met here in Enschede, I wish to express my thank and my apology for not being able to mention them personally here.

Last, but certainly not least, I thank my parents and my brothers for supporting me from the distance. I dedicate this thesis to them.

Index