QUEEN'S UNIVERSITY AT KINGSTON

DOCTORAL THESIS

Decentralized Problems of Discrete-Event Systems:

Epistemic Reasoning and Graph Representation

Author:
K. RITSUKA

Supervisor:
PROF. KAREN RUDIE

Thesis Committee:
PROF. JOSHUA MARSHALL
PROF. XIAODAN ZHU
PROF. CHRISTIAN MUISE
PROF. FENG LIN

A thesis submitted to the Graduate Program
in Electrical and Computer Engineering
in conformity with the requirements
for the degree of Doctor of Philosophy

July 2023

This document is written with Visual Studio Code, set in Bitstream Charter, compiled with pdfTeX and biber .

This document uses the LaTeX (both 2$_\varepsilon$ and 3) macro packages. The LaTeX document class is KOMA-Script. Other macro packages used are: `l3keys2e`, `etoolbox`, `fontenc`, `inputenc`, `microtype`, `geometry`, `blankpage`, `mathtools`, `amssymb`, `delarray`, `empheq`, `bm`, `xparse`, `mathalpha`, `extdash`, `biblatex`, `enumitem`, `tasks`, `colortbl`, `multirow`, `placeins`, `flafter`, `float`, `subcaption`, `hyperref`, `bookmark`, `xcolor`, `normalcolor`, `glossaries-extra`, `ntheorem`, `cleveref`, `graphicx`, `tikz`, `hologo`, `pdfpages`, `ctex`, and `mylatexformat`.

This document is compiled with `\ACtrue\Twosidetrue\Letterfalse\Darkfalse`.

To Mom and Dad

To Karen Rudie,
who is philo-sophy-ical

致 愛人
桥霜

Abstract

This thesis provides modelling formalisms for decentralized problems of discrete-event systems for better derivation of problem solvability and solution constructions and for comparison of one decentralized architecture with another.

The thesis establishes equivalence between three classes of problems — observation problems, diagnosis problems, and control problems — in terms of Turing reduction. Through the reduction, the thesis demonstrates that the solvability of the control problems is undecidable, alongside the similar result known for the other two classes of problems. Moreover, since the observation problems are formally simpler, the thesis advocates focusing research effort on these problems whenever suitable, i.e., when the results can transfer to the other two classes of problems via the reduction.

The thesis then puts into uniform frameworks solutions to decentralized problems, characterized by their *architectures*. Two such frameworks have been proposed, one formalized using epistemic logic, while the other in terms of graph theory. Both frameworks capture the essential indistinguishability relations.

The epistemic logic formalism is primarily suited for methodologically deriving problem solvability conditions and solution constructions under a given architecture. The methodology promotes *coupling* knowledge and action, so that a problem solvability condition directly expresses what knowledge an agent needs to perform its actions. This contrasts with the traditional case-by-case, *ad hoc* approach. The resulting epistemic expressions are closer to human reasoning than the traditionally-used predicate logic. We provide epistemic expressions for well-known problem solvability conditions. Being able to circumscribe a collection of such conditions in a uniform and concise modelling paradigm, we are able to refine the known hierarchy of such conditions in a more concise manner.

The graph theoretical formalism provides a direct approach to compare architectures. This contrasts with the traditional approach in which one first derives problem solvability conditions for the architectures to be compared, and then shows that one condition (strictly) implies another. The approach also gives a visually intuitive model for understanding why a given architecture is superior to another.

Acknowledgements

I would like to express my utmost gratitude to my supervisor, Prof. Karen Rudie. For four years, she has provided supervision and guidance while giving me the freedom to pursue my ideas. I fear that I have not been a quite docile student, yet she showed me much patience, and often persuaded rather than instructed. Her support and encouragement, both within my research work and within my life, ensured the final quality of this research.

I would also like to thank the members my thesis committee, Prof. Joshua Marshall, Prof. Xiaodan Zhu, Prof. Christian Muise, and Prof. Feng Lin, for their careful examination and critical comments of my dissertation.

The research described in this thesis was undertaken at Queen's University, which is situated on traditional Anishinaabe and Haudenosaunee territory. The research was inspired by and supported through an NSERC CRD-DND project with General Dynamics Land Systems–Canada and Defence Research and Development Canada.

I would like to appreciate those who provided me education, demonstration, and inspiration. To those at Queen's University, especially Prof. Naraig Manjikian, Prof. Karen Rudie, Prof. Kai Salomaa, Prof. Jana Dunfield. To Prof. William Farmer at McMaster University. To those at my high school, especially Mr. James Smith.

Appreciation also goes to Student Wellness Services of Queen's University, Queen's University Library, and City of Kingston.

To my friends, especially 申畅，刘亦桐，严博琛，林霄峰，周雯芯. To 王桦.

To my extended family, especially my parents, 里葆玲 and 晏刚, and my paternal grandmother, 马如明.

To my love, 桥霜.

Queen's University, Kingston
July 2023

K. Ritsuka
晏雲杉

Contents

List of Figures

List of Symbols

$\models$	sematics of epistemic formulae	12
$\sim$	accessibility relation that is a partial equivalence relation (except in Chapter 8)	14
$\simeq$	accessibility relation that is an equivalence relation	14
$\langle g_1, \ldots, g_n \rangle$	broadcasting application	111
$(g_1, \ldots, g_n)$	element-wise application	111
$\mathcal{CD}$	set of control decisions	7
Con	class of control problems	103
$\mathrm{C}(L, K, \Sigma_{i,o}, \Sigma_{i,c})$	an instance of control problem	103
d	σ can be disabled	45
$\underline{d}$	σ must be disabled	45
δ	transition function	5
$(\mathcal{D}, \sim)$	decision graph	113
$\mathcal{D}$	as an abbreviation of $(\mathcal{D}, \sim)$	113
Dx	class of diagnosis problems	102
$\mathrm{D}(L, \Sigma_{i,o}, \sigma_f, m)$	an instance of diagnosis problem	102
e	σ can be enabled	45
$\underline{e}$	σ must be enabled	45
ε	the empty string	5
$[\cdot]$	(generalized) equivalence class	14
$E\phi$	everyone knows ϕ	45
E	automaton specification of the legal language, a sub-automaton of G	9
f_i	the i-th agent in $\mathcal{N}$	6
$f_{\mathcal{N}}$	joint supervision/observation	7
$f_{\mathcal{N}}/G$	closed loop system	7
G	A finite-state automaton, usually the plant	5
G^{obs}	observer	13
G'	$G \times G_1^{obs} \times \cdots \times G_n^{obs}$	13
I	Kripke structure (frame)	11
K	legal language	100
$K_i\phi$	agent i knows ϕ	12

$K_i^0\sigma_d$	$K_i\sigma_d$	63
$K_i^0\sigma_e$	$K_i\sigma_e$	63
$K_i^1\sigma_d$	$K_i(\sigma_{e!} \Rightarrow O\sigma_e)$	63
$K_i^1\sigma_e$	$K_i(\sigma_{d!} \Rightarrow O\sigma_d)$	63
L	physically possible language (when occurring alongside K)	100
$L(\cdot)$	language recognized by given automaton	5
$(L, \sim)$	observation graph	112
L	as an abbreviation of $(L, \sim)$ (only in Chapter 8)	112
n	number of agents in $\mathcal{N}$	6
$\mathcal{N}_\sigma$	set of agents controlling σ	6
$\mathcal{N}$	set of agents	6
$O\phi$	someone (other than i, given by the context) knows ϕ	45
Obs	class of observation problems	101
$\mathrm{O}(L, K, \Sigma_{i,o})$	an instance of observation problem	101
$\mathscr{P}$	power set	13
P_i	i-th projection/observation function	6
$P_i(s) = P_i(s')$	s is indistinguishable from s' to agent i	
q	a typical automaton state in Q.	
q_0	initial state	5
Q	set of states	5
s	a typical string in Σ^*, used to represent a sequence of events	
σ	a typical letter in Σ, used to represent an event	
σ_E	σ is legal	14
σ_G	σ is physically possible	14
σ_d	σ can be disabled	62
σ_e	σ can be enabled	62
σ_*	$*$ is to be chosen from e and d	63
$\sigma_{d!}$	σ must be disabled	62
$\sigma_{e!}$	σ must be enabled	62
$S\phi$	someone knows ϕ	45
Σ	alphabet, typically stands for set of possible events	5
Σ_c	controllable events	6
$\Sigma_{i,c}$	i-th set of controllable events	6
$\Sigma_{i,o}$	i-th set of observable events	6
Σ_o	observable events	6

Σ^*	set of all strings over Σ	5
Σ_{uc}	uncontrollable events	6
Σ_{uo}	unobservabble events	6
w	a world in a Kripke structure	
w_e	the “environment” component of w	13

List of Abbreviations

C&P	Conjunctive and Permissive	18
D&A	Disjunctive and Anti-permissive	18
DES	discrete-event systems	1
DSCOP	Decentralized Supervisory Control and Observation Problem	9
FSA	finite state automaton	5

1 Introduction

This thesis establishes two formal modelling frameworks for studying the observation problems of discrete-event systems (DES).

The research is motivated by a research collaboration with General Dynamics Land Systems Canada (GDLS-C) and Defence Research and Development Canada (DRDC), where we were interested in maintaining secrecy in the operation of a group of autonomous agents. At that time, we inspected one of the methods to maintain secrecy: decentralized supervisory control. We noticed that, while the problem of decentralized supervisory control has been partially solved under numerous *architectures* [Cie+88; RW92; PKK97; YL02; RR00; YL04; RR07; KT05; KC18; CK08a; CK08b; CK11], there is no proof or even indication for any of these architectures to be the most general. Specifically, an architecture later in the forgoing list solves more problems than one earlier in the list, while the list is not known to stop from growing. An interesting incidence is that Yoo and Lafortune [YL02] called their architecture the "general architecture" [YL02], which subsumes the three architectures listed before it. However as we are seeing now, many architectures that are more "general" have come thereafter.

Moreover, as the list of architectures grows, the formalisms of the new architectures become more and more complicated. First, it is becoming more difficult to compare new architectures. In addition to the complex specifications of the architectures, the traditional approach is indirect: one has to derive problem solvability conditions for the architectures and compare the conditions instead. Then, all existing problem solvability conditions seem to be derived in a case-by-case, ad-hoc manner, with no uniform methodology. Consequently, it has become harder to verify such conditions for more complicated architectures, let alone to derive the conditions.

The thesis addresses the issues above by providing two unifying frameworks, one in terms of epistemic logic and the other based on graph theory.

Our epistemic logic formalism provides a concise and intuitive language compared to the currently used predicate logic language in describing multi-agent behaviour under partial observation. With this formalism, the thesis advocates a methodology of linking knowledge and action, which allows us to see easily what knowledge an agent must possess to achieve the desired control strategy. This methodology yielded more concise and intuitive expressions for problem solvability conditions of various existing architectures. The effort yielded a refined hierarchy of existing architectures, both in the sense that the conditions are put in a unified language, from which comparisons fall out, and that it allowed more existing architectures to

be positioned easily in the hierarchy.

Then, an alternative, graph-theoretical framework for decentralized problems is proposed. This framework circumvents the indirect approach to compare two architectures by providing a direct one.

Finally, it should be noted that while the discussion above is articulated in the context of decentralized supervisory control problems, they apply to decentralized observation problems and decentralized diagnosis problems as well. In fact, due to the equivalence between the three classes of problems established by this thesis, some of the results mentioned above were achieved by studying the formally simpler observation problems instead.

The thesis is organized as follows. The thesis begins by unifying architectures for control problems under the umbrella formalism of epistemic logic.

- Chapter 2 presents essential definitions.
- Chapter 3 puts some decentralized control architectures into a uniform framework of epistemic logic.
- Chapter 4 extends Chapter 3 by adding the conditional architecture into the framework. In the meantime, Chapter 4 demonstrates that with epistemic logic, problem solvability conditions and solutions can be derived systematically.
- Chapter 5 formally unifies the conditional architecture with the inference-based architecture.
- Chapter 6 provides a visual alternative to the epistemic logic formalism.

The thesis then proceeds to Chapter 7, where we demonstrate that observation problems are equivalent to control problems as well as to diagnosis problems. The result presented here indicates that results in Chapters 3, 4 and 6 can be easily adapted to the observation problems. Furthermore, the equivalence directly entails unsolvability of the control problems in the general case.

Chapter 8 proposes a graph-theoretic translation of the epistemic logic formalism of the unifying framework. The chapter promotes the graph-theoretic formalism primarily as a direct approach for comparing architectures

Finally, Chapter 9 summarizes the thesis.

Some of the chapters are published:

Chapter 3

K. Ritsuka and Karen Rudie. "Epistemic interpretations of decentralized discrete-event system problems". In: *Discrete Event Dynamic Systems* 32.3 (June 2022), pp. 359–398. DOI: 10.1007/s10626-022-00363-7

Chapter 4

K. Ritsuka and K. Rudie. *Do What You Know: Coupling Knowledge with Action in Discrete-Event Systems*. Submitted for publication. 2023

An old draft is available as arXiv:2108.02000. This preprint version differs substantially from the current version submitted for peer-review.

Chapter 6

K. Ritsuka and Karen Rudie. "A Visualization of Inference-Based Supervisory Control in Discrete-Event Systems". In: *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, Dec. 2021. DOI: 10.1109/cdc45484.2021.9683210

Chapter 7 An old, significantly different version was available on arXiv, which does not include discussion on diagnosis problems.
K. Ritsuka and Karen Rudie. *A correspondence between control and observation problems in decentralized discrete-event systems*. 2022. arXiv: 2204.10792 [eess.SY]

Chapter 8

K. Ritsuka and Karen Rudie. *A Uniform Treatment of Architectures and Fusion Rules in Decentralized Discrete-Event Systems*. 2022. arXiv: 2210.16511 [eess.SY]

2 Preliminaries

The work in this thesis uses discrete-event system models and also epistemic logic as a language to describe supervisory control. Therefore, in this chapter we provide a brief summary of the necessary concepts from each domain.

2A Discrete-Event Systems

A discrete-event system is a system with finite and discrete state space, which executes actions and changes its internal state according to only its current state. We call the occurrence of an action an *event*. We consider the system's behaviours to be all finite sequences of events the system can generate from a certain initial state.

Formally we define discrete-event systems following Wonham and Cai [WC18] and Cassandras and Lafortune [CL07].

Definition 2A.1
Denote a *plant* modelled as a finite state automaton (FSA) by

$$G = (\Sigma, Q, \delta, q_0)$$

where Σ is a finite set of *events*, Q a finite set of *states*, $\delta \subseteq Q \times \Sigma \times Q$ the *transition relation*, and $q_0 \in Q$ the unique *initial state*.

Without loss of generality, it is assumed that δ is univalent and can be seen as a partial function $\delta\colon Q \times \Sigma \nrightarrow Q$. We write $\delta(p, \sigma) = q$ for the unique q s.t. $(p, \sigma, q) \in \delta$ if such q exists. In this case we write $\delta(p, \sigma)!$ and say $\delta(p, \sigma)$ is defined when the particular value of q is not of interest.

The collection of all finite sequences over Σ is denoted as Σ^*. An element of Σ^* represents a sequence of event occurrences and is called a *string*. The empty string is denoted by ε.

The transition function δ can be inductively extended on its second argument so that $\delta\colon Q \times \Sigma^* \nrightarrow Q$.

In cases where confusion could arise, we superscript components of an automaton with the automaton's name. For example, we use Q^G to refer to the state set of G.

The *language generated by* G is defined as

$$L(G) = \{\, s \in \Sigma^* \mid \delta(q_0, s)! \,\}$$

A language L is said to be *prefix-closed* whenever for all strings $s\sigma \in L$, it is always the case that $s \in L$.

The language $L(G)$ is interpreted as the set of physically possible behaviours of G. By definition, $L(G)$ is always prefix-closed.

2A1 Decentralized Supervisory Control with Partial Observations

A plant's behaviours may not all be desirable. In such a case, we constrain its behaviours through supervisory control. In the problems we consider, we allow an arbitrary number of supervisors to jointly perform the control, where each supervisor observes and controls a subset of events.

Decentralized control has been examined by many DES researchers. For a more extensive discussion, see Cassandras and Lafortune [CL07, Chapter 3.8] on decentralized control.

With an event being controlled potentially by multiple supervisors, a mechanism to combine control decisions by these supervisors is necessary. Prosser, Kam, and Kwatny [PKK97] explicitly name such mechanisms *fusion rules*. Later work by Yoo and Lafortune [YL02] realized that fusion rules for each event can be chosen separately and independently.

Formally, we express the decentralized supervisory architecture as follows.

Definition 2A1.1
Let $\mathcal{N} = \{f_1, \ldots, f_n\}$ be a finite set of n supervisors for plant G. We write i instead of f_i when referring to the supervisor per se; this choice is determined by readability.

For each supervisor $i \in \mathcal{N}$, let $\Sigma_{i,c}, \Sigma_{i,o} \subseteq \Sigma$ be the sets of controllable and observable events for Supervisor i, resp. Denote the set of events controlled by some supervisors $\Sigma_c = \bigcup_{i\in\mathcal{N}} \Sigma_{i,c}$, and the set of events not controlled by any supervisor $\Sigma_{uc} = \bigcap_{i\in\mathcal{N}} \Sigma - \Sigma_{i,c}$. Hence we have $\Sigma_{uc} = \Sigma - \Sigma_c$. The sets Σ_o and Σ_{uo} are defined similarly. Let $\mathcal{N}_\sigma = \{\, i \in \mathcal{N} \mid \sigma \in \Sigma_{i,c} \,\}$ be the set of agents controlling σ.

For each $i \in \mathcal{N}$, define a function that represents a supervisor's observation. Define the projection function $P_i : \Sigma \to \Sigma_{i,o} \cup \{\, \varepsilon \,\}$ such that $P_i(\sigma) = \sigma$ if $\sigma \in \Sigma_{i,o}$ and

$P_i(\sigma) = \varepsilon$ otherwise. Informally, P_i erases unobservable events and preserves observable events in their original sequential order. Extend P_i from Σ to Σ^* inductively.

With a slight abuse of notation, we use $P_i(G)$ to denote the automaton constructed by replacing all transitions labelled by an unobservable event with ε and determinized, so that $P_i(G)$ recognizes the language $P_i(L(G))$.

Let $\mathcal{CD}$ be the set of supervisory control decisions. Now supervisors can be prescribed by $f_i : P_i(L(G)) \times \Sigma_{i,c} \to \mathcal{CD}$ for all $f_i \in \mathcal{N}$. Specifying supervisors taking arguments from $P_i(L(G))$ instead of $L(G)$ encodes requirements traditionally referred to as *feasibility* and *validity*, i.e., a supervisor should behave consistently for two strings s, s' such that $P_i(s) = P_i(s')$. We focus only on FSA-based supervisors. That is, a supervisor f_i can be realized as a Moore machine (S_i, f_i') such that $f_i(s, \sigma) = f_i'(\delta_i(q_{i,0}, s), \sigma)$, where S_i is an FSA $(\Sigma, Q_i, \delta_i, q_{i,0})$, and $f_i' : Q_i \times \Sigma_{i,c} \to \mathcal{CD}$. We will refer to f_i' simply as f_i when convenient.

For each controllable event σ, let $cd_{\mathcal{N}_\sigma}$ denote the collection of control decisions issued by supervisors $i \in \mathcal{N}_\sigma$, hence $cd_{\mathcal{N}_\sigma}$ has exactly $|\mathcal{N}_\sigma|$ elements. Let $\mathcal{CD}_{\mathcal{N}_\sigma}$ be the collection of all such $cd_{\mathcal{N}_\sigma}$'s. Let $\mathcal{FD} = \{$ **enable**, **disable** $\}$ be the set of fused decisions. Let $f_\sigma : \mathcal{CD}_{\mathcal{N}_\sigma} \to \mathcal{FD}$ be the fusion functions chosen separately for each $\sigma \in \Sigma_c$, and the joint supervision $f_\mathcal{N} : L(G) \times \Sigma_c \to \mathcal{FD}$ be defined as $f_\mathcal{N}(s, \sigma) = f_\sigma(\{ f_i(P_i(s), \sigma) \}_{i \in \mathcal{N}_\sigma})$. Consequently, only decisions issued by supervisors $i \in \mathcal{N}_\sigma$ are fused, and decisions of supervisors not controlling the event σ are ignored.

The closed-loop behaviour of the plant with the joint supervision imposed is denoted as defined $L(f_\mathcal{N}/G)$, and defined inductively as the smallest set such that:

- $\varepsilon \in L(f_\mathcal{N}/G)$
- $s \in L(f_\mathcal{N}/G) \wedge s\sigma \in L(G) \wedge \sigma \in \Sigma_{uc} \Rightarrow s\sigma \in L(f_\mathcal{N}/G)$
- $s \in L(f_\mathcal{N}/G) \wedge s\sigma \in L(G) \wedge \sigma \in \Sigma_c \wedge f_\mathcal{N}(s, \sigma) = \textbf{enable} \Rightarrow s\sigma \in L(f_\mathcal{N}/G)$

The second bullet point in the definition of closed-loop behaviour captures the requirement that a physically possible event that is not controllable by any supervisor must be allowed to occur under supervision. The third bullet point says that a physically possible event that is controllable and for which the fused decision is **enable** must be allowed to occur under supervision.

Fig. 2.1 illustrates the architecture defined above, which is adapted from Fig. 2 by Yoo and Lafortune [YL04] with entities labelled according to our symbolism.

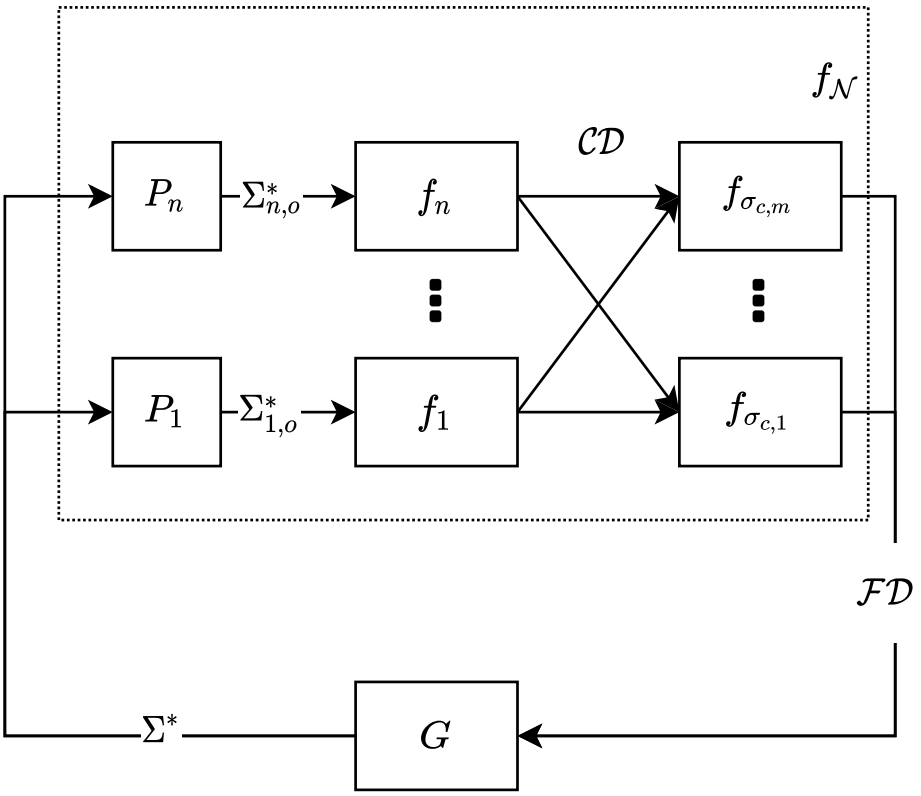

Figure 2.1: Architecture of Decentralized Control of DES. The controllable events are $\sigma_{c,1}, \ldots, \sigma_{c,m}$.

Whereas the fusion function f can be seen as an n-ary operation on supervisory control decisions $\mathcal{CD}$, there is no operation over the fused decision set $\mathcal{FD}$, since elements in this set are to be interpreted as fused decisions and should be regarded as final.

In the studies of decentralized problems, it is customary to identify a fusion rule as an architecture.

In particular, even though for the architectures with binary control decisions, it happens to be the case that $|\mathcal{CD}| = |\mathcal{FD}|$, and by choosing $\mathcal{CD} = \mathcal{FD} = \{\mathbf{0}, \mathbf{1}\}$ (the Boolean values), where $\mathbf{0}$ (resp., $\mathbf{1}$) stands for **disable** (resp., **enable**), the fusion rule can be conveniently described as Boolean conjunction/disjunction[1], as Rudie and Wonham [RW92] and Prosser, Kam, and Kwatny [PKK97] did, we refrain from doing so to avoid conflating the *de facto* different sets of binary control decisions. As we will reveal, with an epistemic approach, a control decision $\mathbf{0}$ issued in a conjunctive architecture has a different meaning from the same decision issued in a disjunctive architecture; and a similar observation can be made for the other control decision $\mathbf{1}$.

The sets $\mathcal{CD}$ and $\mathcal{FD}$ being disjoint also simplifies discussion: we can now refer to an element of either set without explicitly stating from which set it comes. We also refer to a certain element of either set simply as a decision when no confusion would arise.

Remark 2A1.2
Whereas the set $\mathcal{CD}$ determines the number of distinct control decisions available to the supervisors, what those decisions mean — their semantics — is given by the fusion rule f. Although the symbols we choose for control decisions may be formally meaningless, we will still choose them with the intended fusion rule in mind. ∎

Constructing multiple supervisors jointly restricting a plant's behaviours will be called the *Decentralized Supervisory Control and Observation Problem* (DSCOP). We will use the term "condition" (without qualification) to refer to the *necessary and sufficient* condition needed to solve DSCOP.

For the sake of comparison, we will use the following generic definition of DSCOP as a common ground for subsequent discussions.

Problem 2A1.3 (Decentralized Supervisory Control and Observation Problem, DSCOP)
Given an automaton G which specifies the plant behaviour as the prefix-closed

[1] Perhaps under the influence of Prosser, Kam, and Kwatny [PKK97], the work of Takai and Ushio [TU01], which precedes Takai, Kumar, and Ushio [TKU05], reverses the meaning of $\mathbf{0}$ and $\mathbf{1}$. Hence their OR (resp., AND) rule corresponds to our conjunctive (resp., disjunctive) rule. We follow the more common convention here.

language $L(G)$, an automaton E which specifies the legal behaviour as the prefix-closed language $L(E)$, n pairs of controllable/observable event sets, choose an appropriate set of control decisions $\mathcal{CD}$, and a fusion rule f, and synthesize a set $\mathcal{N}$ of supervisors, such that $L(f_{\mathcal{N}}/G) = L(E)$.

We arrange E to be a subautomaton of G as we will need this assumption to construct the structure relevant to the interpretation of our epistemic expressions

We usually study the condition for a class of DSCOP for $\mathcal{CD}$ and f that are fixed *a priori*. See also Rmk. 2A1.2. In particular, $\mathcal{CD}$ and f should be independent of any specific G and E. We have to emphasize that in practice one is certainly free to choose whatever $\mathcal{CD}$ and f necessary to solve the problem at hand. Fixing $\mathcal{CD}$ and f allows us to classify pairs of G and E according to the $\mathcal{CD}$ and f sufficient for the decentralized control problem to be solvable, and thus allows comparison among pairs of $\mathcal{CD}$ and f.

The problem solvability condition of an architecture is usually called the co-observability condition of that architecture. Sometimes more specific names have been given when multiples architectures are being studied together.

2B Epistemic Logic

Ricker and Rudie [RR00; RR07] observed that reasoning about the decision-making of decentralized supervisors could be facilitated using formal reasoning about knowledge, via epistemic logic. Although formal conditions for solving DSCOP can be described using conditions on strings in languages, and hence do not require a formal logic description, epistemic logic provides a natural modelling paradigm that parallels natural languages, thus giving better intuition into the reasoning behind the decisions that supervisors make. Specifically, the epistemic operator in the language expresses concepts such as "agent i *knows* a certain event must be disabled". Discussing a supervisor with such anthropomorphic phrases puts oneself in the perspective of the supervisor and reveals what "knowledge" the agent must have to make a decision.

Epistemic logic as used in distributed computing problems was first presented by Halpern and Moses [HM90]. See Fagin et al. [Fag+04] for more details. We provide in the remainder of this section the concepts from epistemic logic needed to understand our work.

Definition 2B.1
For a fixed set $\mathcal{V}$ of variables, where v denotes some element of $\mathcal{V}$, and a fixed finite

set $\mathcal{N}$ of agents, where i denotes some element of $\mathcal{N}$, the set of epistemic modal formulae is defined inductively by the following grammar:

$$\begin{array}{rll} S,T & ::= & (v) \quad \text{propositional variable } v \\ & | & (\neg S) \quad \text{negation of } S \\ & | & (S \wedge T) \quad \text{conjunction of } S, T \\ & | & (K_i S) \quad \text{agent } i \text{ knows } S \end{array}$$

Definition 2B.2
It is conventional to define other connectives from the primitive ones above:

- $(\alpha \vee \beta) =_{df} \neg(\neg\alpha \wedge \neg\beta)$,
- $(\alpha \Rightarrow \beta) =_{df} (\neg\alpha \vee \beta)$

Where convenient, we use the connectives defined above to express ideas, but when reasoning about epistemic formulae, we assume that defined connectives of Defn. 2B.2 have all been syntactically expanded, so we only have to deal with primitive ones of Defn. 2B.1.

We omit parentheses according to the following precedence convention: unary operators $\neg, K_i$ bind tightest, then $\wedge, \vee, \Rightarrow$.

When an expression S is short enough, we sometimes write $\neg S$ as $\overline{S}$ for compactness.

The semantics of epistemic formulae are given through the use of a structure called a Kripke structure.

Definition 2B.3
For some $\mathcal{V}$ and $\mathcal{N}$, a Kripke structure, or simply a frame I is

$$(W, \pi, \{ \sim_i \}_{i \in \mathcal{N}})$$

where

- W is a finite set of possible worlds, or states [2]
- $\pi : W \times \mathcal{V} \to \{ \textbf{true}, \textbf{false} \}$ evaluates each propositional variable in $\mathcal{V}$ at each possible world in W to either **true**, or **false**.

[2]The term "states" should cause no confusion in this context, since the worlds in the frames we construct in this work happen to be states of some FSA.

- For each $i \in \mathcal{N}$, $\sim_i \subseteq W \times W$ is the accessibility relation over possible worlds, and we say world w' is considered by agent i as an epistemic alternative if $w' \sim_i w$.

Whereas the accessibility relations are commonly required to be equivalence relations over W, a formal construction we will present uses relations that are not reflexive, and are thus partial equivalence relations. Hence we denote accessibility relations as $\sim$, and reserve $\simeq$ for discussions in which the relations are indeed equivalence relations. Note as Ricker and Rudie [RR07] do not distinguish these cases, they used $\sim$ for the latter.

While, as we have already signified, the language of epistemic logic gives intuitive understanding of how agents reason about uncertainty and choose control decisions accordingly, we make no philosophical claim over what knowledge is. That is, we use epistemic logic purely as a formal instrument, to encapsulate complexity of expressions otherwise given rise to by using predicate logic, which is commonly used in traditional approaches toward supervisory control of DES, such as Rudie and Wonham [RW92] and Yoo and Lafortune [YL04].

In our formalism, the propositional connectives (the second and third items in Defn. 2B.4 below) are to be understood as usual. The semantics of the epistemic operator (the last item in Defn. 2B.4) reflect that, upon observing a sequence of events generated by the plant, a supervisor can only *know* something (i.e., be certain that it is true), if it is always true after any sequence (generated by the plant) that looks the same to the supervisor as the sequence of events it has observed.

To reflect the discussion above, we thus adopt the following formal definition of the semantics of epistemic formulae as the relation $\models$ of pairs of Kripke structures and worlds, and epistemic modal formulae, given inductively over the structure of the formulae.

Definition 2B.4

- $(I, w) \models v$ iff $\pi(w, v) =$ **true**
- $(I, w) \models \neg S$ iff it is not the case that $(I, w) \models S$
- $(I, w) \models S \wedge T$ iff $(I, w) \models S$ and $(I, w) \models T$
- $(I, w) \models K_i S$ iff for all $w' \in W$ such that $w' \sim_i w$, $(I, w') \models S$.

By construction, either $(I, w) \models S$ or not, so we have either $(I, w) \models S$ or $(I, w) \models \neg S$, i.e., one and only one of a formula and its negation is satisfied. Moreover, it is decidable. Hence we can inductively extend π from propositional variables to epistemic formulae, subject to the interpretation I, and regard the relation $\models$

as the relation satisfying $(I, w) \models S \Leftrightarrow \pi_I(S, w) =$ **true**. Henceforth when talking about the semantic of formulae against interpretations, we also call formulae as expressions, because they evaluate.

Often, throughout a discussion, all epistemic expressions are evaluated against the same pair of I, w. In those cases, to avoid repeatedly writing $(I, w) \models \cdot$, we tend to simply say S in place of $(I, w) \models S$.

2B1 Epistemic Logic in DSCOP

In this subsection, we discuss how Kripke structures are constructed to express DSCOP problems. These Kripke structures will be used to interpret a number of epistemic expressions, where an expression being interpreted as truth corresponds to a co-observability condition holding. The epistemic expressions will then give denotation to the control decisions and fusion rules. This approach is adapted from Ricker and Rudie [RR07] with some necessary modifications.

Consider a plant G, a subautomaton E of G specifying the legal behaviour, n pairs of sets of controllable/observable events. Construct $G_i^{obs} = P_i(G)$ for each i, where it can also be interpreted $Q_i^{obs} = \{\, \Sigma_{i,uo}\text{-closure of } q \mid q \in Q \,\}$. This is agent i's perception of the plant under partial observation, i.e., the agent cannot distinguish G and G_i^{obs} by only observing sequences of events generated by these two FSA.

Next we construct a composite structure that will allow us to keep track of plant behaviour and each supervisor's view of the corresponding plant behaviour. We do this through the construction $G' = G \times G_1^{obs} \times \cdots \times G_n^{obs} = (\Sigma, Q', \delta', q_0')$, where $Q' \subseteq Q \times Q_1^{obs} \times \cdots \times Q_n^{obs} \subseteq Q \times \mathcal{P}Q \times \cdots \times \mathcal{P}Q$ and $\mathcal{P}Q$ is the powerset of Q, δ' is component-wise application of δ^G and δ_i^{obs} for $i \in \mathcal{N}$, $q_0' = (q_0, q_{0,1}^{obs}, \ldots, q_{0,n}^{obs})$ where $q_{0,i}^{obs} \in Q_i^{obs}$ and thus $q_{0,i}^{obs} \subseteq Q$ for $i \in \mathcal{N}$.

Our composite structure G' generates the same language as G does, however the Cartesian product of states forming Q' allows us to track more information than that is available by simply tracking the sequence of states in Q visited by some sequence of events in the plant language. Namely, $(q, q_1^{obs}, \ldots, q_n^{obs}) \in Q'$ records not only the current state q of G, but also each agent's best estimation q_i^{obs} of the set of states the plant could possibly be at based on agent i's observation, for each $i \in \mathcal{N}$, which is to be expected since the actual plant state should always be a state that any observer *thinks* the plant could be in.

Now we are ready to construct the Kripke structure against which the expressions of the various co-observability conditions are interpreted.

We start by letting $W = Q'$. To avoid multiple arguments with both subscripts and superscripts, we will write $(w_e, w_1, \ldots, w_n)$ for an element of W instead of $(q, q_1^{obs}, \ldots, q_n^{obs})$,

Each of the two Kripke structures we are about to discuss is constructed with one of the following two kinds of accessibility relations.

The accessibility relations $\simeq_i$ is constructed such that $w \simeq_i w'$ whenever $w_i = w'_i$. The accessibility relations $\simeq_i$ are clearly equivalence relations. Hence denote $\{\, w' \in W \mid w' \simeq_i w \,\}$ as $[w]_{\simeq_i}$, or simply $[w]_i$. This coincides with the construction by Ricker and Rudie [RR07].

The other kind of accessibility relations $\sim_i$ is constructed such that $w \sim_i w'$ whenever $w_e \in Q^E \wedge w'_e \in Q^E \wedge w_i = w'_i$. Particularly note that $\sim_i$ is an equivalence relation on $\{\, w \in W \mid w_e \in Q^E \,\}$, and for all w such that $w_e \notin Q^E$, w has no referent nor relatum (participating $\sim_i$). Hence, the relations $\sim_i$ are *partial equivalence relations*. It is reasonable to consider the equivalence class $[w]_{\sim_i}$, or simply, $[w]_i$, whenever $w_e \in Q^E$. Only with an abuse of notation, let $[w]_i = \varnothing$ for $w_e \notin Q^E$. Informally, one may interpret $[w]_i$ as containing exactly the worlds that are epistemic alternatives to w as perceived by agent i.

The collection of all equivalence classes induced by $\simeq_i$ ($\sim_i$) is denoted as $\ker \simeq_i$ ($\ker \sim_i$).

We need the following atomic propositions, taken from [RR07], to capture the presence/absence of transitions of an event σ. Hence we let $\mathcal{V} = \bigcup_{\sigma \in \Sigma} \{\, \sigma_G, \sigma_E \,\}$, and define

$$\pi(w, \sigma_G) = \begin{cases} \textbf{true} & \delta^G(w, \sigma)! \\ \textbf{false} & \text{otherwise} \end{cases}$$

$$\pi(w, \sigma_E) = \begin{cases} \textbf{true} & \delta^E(w, \sigma)! \\ \textbf{false} & \text{otherwise} \end{cases}$$

The intended meaning of $\pi(w, \sigma_G) = \textbf{true}$ is that σ can physically occur at state w, as specified by G; whereas $\pi(w, \sigma_E) = \textbf{true}$ indicates that σ is legal and should be allowed to happen. It follows that $\pi(w, \sigma_E) = \textbf{true} \Rightarrow \pi(w, \sigma_G) = \textbf{true}$, which reflects the fact that E is a subautomaton of G.

Finally, let the Kripke structures be $\overline{I} = (W, \pi, \{\, \simeq_i \,\}_{i \in \mathcal{N}})$ and $I = (W, \pi, \{\, \sim_i \,\}_{i \in \mathcal{N}})$.

Technically, the constructed Kripke structures $\overline{I}$ and I are parameterized over certain G, $\{\, P_i \,\}_{i \in \mathcal{N}}$, and E. However, our discussion will not simultaneously concern multiple sets of these entities, but assume an indefinite one, hence we write simply

$\overline{I}$ and I, rather than, say, $\overline{I}(G, P_1, \cdots, P_n, E)$, for the Kripke structure parameterized over that indefinite, but specific set of arguments.

The difference between the two kinds of Kripke structures is as follows. Since by construction we have $\sim_i \subseteq \simeq_i$, hence the condition $\overline{I} \models K_i(\phi)$ is strictly stronger than the condition $I \models K_i(\phi)$.

References

[CL07] Christos G. Cassandras and Stéphane Lafortune. *Introduction to Discrete Event Systems*. Second. Springer-Verlag GmbH, 2007. 772 pp. [cit. on p. 5. 6].

[Fag+04] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Vardi. *Reasoning About Knowledge*. The MIT Press, 2004. DOI: 10.7551/mitpress/5803.001.0001. [cit. on p. 10].

[HM90] Joseph Y. Halpern and Yoram Moses. "Knowledge and common knowledge in a distributed environment". In: *Journal of the ACM* 37.3 (July 1990), pp. 549–587. DOI: 10.1145/79147.79161. [cit. on p. 10].

[PKK97] J. H. Prosser, M. Kam, and H. G. Kwatny. "Decision fusion and supervisor synthesis in decentralized discrete-event systems". In: *Proceedings of the American Control Conference*. IEEE, 1997. DOI: 10.1109/ACC.1997.608978. [cit. on p. 6. 9].

[RR00] S. L. Ricker and K. Rudie. "Know means no: Incorporating knowledge into discrete-event control systems". In: *IEEE Transactions on Automatic Control* 45.9 (2000), pp. 1656–1668. DOI: 10.1109/9.880616. [cit. on p. 10].

[RR07] S. L. Ricker and K. Rudie. "Knowledge Is a Terrible Thing to Waste: Using Inference in Discrete-Event Control Problems". In: *IEEE Transactions on Automatic Control* 52.3 (Mar. 2007), pp. 428–441. DOI: 10.1109/TAC.2007.892371. [cit. on p. 10. 12. 13. 14].

[RW92] K. Rudie and W. M. Wonham. "Think globally, act locally: decentralized supervisory control". In: *IEEE Transactions on Automatic Control* 37.11 (1992), pp. 1692–1708. DOI: 10.1109/9.173140. [cit. on p. 9. 12].

[TKU05] S. Takai, R. Kumar, and T. Ushio. "Characterization of co-observable languages and formulas for their super/sublanguages". In: *IEEE Transactions on Automatic Control* 50.4 (Apr. 2005), pp. 434–447. DOI: 10.1109/tac.2005.844724. [cit. on p. 9].

[TU01] S. Takai and T. Ushio. “Strong co-observability conditions for decentralized supervisory control of discrete event systems”. In: *Proceedings of the 40th IEEE Conference on Decision and Control*. IEEE, 2001. DOI: 10.1109/cdc.2001.980821. [cit. on p. 9].

[WC18] W. Murray Wonham and Kai Cai. *Supervisory Control of Discrete-Event Systems*. Springer-Verlag GmbH, 2018. 487 pp. [cit. on p. 5].

[YL02] T.-S. Yoo and Stéphane Lafortune. “A General Architecture for Decentralized Supervisory Control of Discrete-Event Systems”. In: *Discrete Event Dynamic Systems* 12.3 (2002), pp. 335–377. DOI: 10.1023/a:1015625600613. [cit. on p. 6].

[YL04] T.-S. Yoo and S. Lafortune. “Decentralized Supervisory Control With Conditional Decisions: Supervisor Existence”. In: *IEEE Transactions on Automatic Control* 49.11 (Nov. 2004), pp. 1886–1904. DOI: 10.1109/tac.2004.837595. [cit. on p. 7. 12].

3 Epistemic Interpretations of Decentralized Discrete-Event System Problems

This chapter presents epistemic characterizations to co-observability conditions in decentralized supervisory control of discrete-event systems. The logical characterizations provide more intuitive interpretations of the various co-observability conditions, and make immediately apparent the relations between the conditions. Closures under set union of some of the conditions are also discussed.

3A Introduction

Different architectures have been explored in the literature for decentralized supervisory control. Since the inference architecture subsumes other architectures — conjunctive architecture [RW92], disjunctive architecture [PKK97],and other variations [TKU05] — we claimed in the earlier work [RR23] that, by simply removing certain lines in our epistemic expression of inference-observability, which corresponds to removing certain control decisions, the conditions for each of these subsumed architectures can also be interpreted epistemically, due to the line-by-line coupling in our expressions. In this chapter we provide such results. Consequently, although relations between the various co-observability conditions were known before (e.g., Takai, Kumar, and Ushio [TKU05]), proving them is not only cumbersome, but also had to be done for each implication individually. In contrast, once each condition is cast as a logic expression, the relationships are immediately apparent.

Moreover, perhaps not satisfied by the absence of desirable algebraic properties — closure under set union/intersection — of the various co-observability conditions, Takai, Kumar, and Ushio [TKU05] provided for each a stronger version to restore the desirable properties. Since these stronger versions are obtained with an algebraic approach, it is not intuitive how much "stronger" these strong versions are relative to the original ones. This chapter provides insight with an epistemic perspective.

3B Co-observability Conditions and Their Strong Versions

The earliest work by Rudie and Wonham [RW92] (and similarly by Cieslak et al. [Cie+88]) on decentralized control problems considered the architecture where the binary set of control decisions are expressed as Boolean values and the fusion rule is taken to be the Boolean conjunction. The condition for the class of problems to be solvable is originally called by Rudie and Wonham [RW92] simply *co-observability*. Later when multiple architectures were being considered simultaneously, Yoo and Lafortune [YL02] called such architecture *Conjunctive and Permissive (C&P) architecture* and changed the name of its condition to *C&P co-observability*.

Prosser, Kam, and Kwatny [PKK97] showed that, when the fusion rule is taken to be the Boolean disjunction, the condition of the problem differs from that of C&P co-observability. Yoo and Lafortune [YL02] called this architecture *Disjunctive and Anti-permissive (D&A) architecture*, and also called the condition of the *D&A architecture*, given by Prosser, Kam, and Kwatny [PKK97] as *D&A co-observability*.

Further, Yoo and Lafortune [YL02] noticed that the fusion rules for each event can be chosen separately and independently. This gives the architecture they originally called *general architecture*, and the condition of which is, potentially confusingly, called *co-observability*. The names of the architecture and its condition are later renamed to *C&P∨D&A architecture* and *C&P∨D&A co-observability*, respectively, by Takai, Kumar, and Ushio [TKU05]. While the control decisions used in the C&P∨D&A architecture can be encoded as Boolean values, as we will reveal, semantically there are three kinds of decisions, two of which will never be issued simultaneously when the language is C&P∨D&A co-observable.

While Takai, Kumar, and Ushio [TKU05] has defined a notion of *C&P∧D&A co-observability* as the conjunction of *C&P co-observability* and *D&A co-observability*, this notion has received less discussion compared to the other co-observability conditions. Particularly, Takai, Kumar, and Ushio [TKU05] did not provide a corresponding *C&P∧D&A architecture* and how a DSCOP problem is solved under such an architecture. More specifically, we might ask "what would be the fusion rule of (and control decisions available to) the *C&P∧D&A architecture*?"

On the other hand, the existence of a condition for each of the architectures indicates that a given DSCOP is not always solvable. In such a case one may be interested to find instead a sublanguage $L' \subseteq L(E)$ such that $L(f_{\mathcal{N}}/G) = L'$. Since an optimal solution does not necessarily exist, it is then interesting to give upper/lower bounds. This has led to the discussion of the "strong" versions of the various co-observability conditions [TKU05]. These strong versions are obtained by pure algebra and hence

offer little intuition of how strong they are.

3C Epistemic Expressions of decentralized control conditions

This section recalls decentralized control conditions by Rudie and Wonham [RW92], Prosser, Kam, and Kwatny [PKK97], Yoo and Lafortune [YL02], and Takai, Kumar, and Ushio [TKU05]. Before proceeding to deriving epistemic expressions for each of the conditions, we provide some informal descriptions.

Both the conjunctive architecture [RW92] and the disjunctive architecture [PKK97] are traditionally described as using two control decisions, **enable** and **disable**, conveniently denoted as Boolean values $\mathbf{1}$ and $\mathbf{0}$. The supervisors subscribe to the following strategy:

In the conjunctive architecture, the fusion rule resolves conflict by the fact that $\mathbf{0} \wedge \mathbf{1} = \mathbf{0}$, i.e., in case of conflict, $\mathbf{0}$ takes precedence over $\mathbf{1}$. Hence, the supervisors' strategy is to issue $\mathbf{0}$ when determined to disable an event, and issue $\mathbf{1}$ when uncertain and expect the best. Hence from each supervisor's perspective, this strategy is permissive. Due to this perspective, the conjunctive architecture was called the conjunctive and permissive (C&P) architecture.

Dually, in the disjunctive architecture, the fusion rule resolves conflict by the fact that $\mathbf{0} \vee \mathbf{1} = \mathbf{1}$, i.e., in case of conflict, $\mathbf{1}$ takes precedence over $\mathbf{0}$. Hence, the supervisors' strategy is to issue $\mathbf{1}$ when determined to enable an event, and issue $\mathbf{0}$ when uncertain and expect the best. Hence from each supervisor's perspective, this strategy is anti-permissive. Due to this perspective, the disjunctive architecture has been called the disjunctive and anti-permissive (D&A) architecture.

From now on, we will use the shorter terms C&P and D&A instead of conjunctive and disjunctive.

The remaining architectures can be thought as a combination of the conjunctive architecture and disjunctive architectures. The interpretations are more involved, hence we postpone them until all architectures have been cast into a uniform description (which uses epistemic logic).

We now proceed to derive epistemic expressions for the various co-observability conditions and related conditions. As it is the point of this chapter that epistemic expressions are closer to human language and are thus easier to interpret, we refrain

from explaining the conditions produced by the original sources and reproduced here. Those conditions are typically expressed in terms of strings and languages, in contrast to the epistemic logic expressions that we will give. Indeed, while the verbal explanations of the language-based expressions can often be found in these original sources, the explanations provided can be seen as an informal attempt to interpret the expressions epistemically.

3C1 C&P co-observability

To obtain an epistemic expression of C&P co-observability, we begin from the following standard definition of C&P co-observability from Rudie and Wonham [RW92]:

$$\begin{aligned} &\forall\, s, \{\, s_i \,\}_{i\in\mathcal{N}}\,. \\ &\quad \left[\bigwedge_{i\in\mathcal{N}} P_i(s) = P_i(s_i)\right] \\ &\quad \Rightarrow \forall\, \sigma \in \Sigma_c\,.\ \bigvee_{i\in\mathcal{N}_\sigma} \left[\begin{array}{l} s_i\sigma \in L(E) \wedge s \in L(E) \wedge s\sigma \in L(G) \\ \Rightarrow s\sigma \in L(E) \end{array}\right] \end{aligned} \tag{3.1}$$

For our later convenience, we extract the quantification $\forall\, \sigma \in \Sigma_c$ outwards. We need the logic identity $[P \Rightarrow \forall\, x\,.\ Q(x)] \Leftrightarrow [\forall\, x\,.\ P \Rightarrow Q(x)]$, where x does not occur free in P, and the commutativity of universal quantifications (under the restriction of no free occurrence). So we have equivalently

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\ \forall\, s, \{\, s_i \,\}_{i\in\mathcal{N}_\sigma}\,. \\ &\quad \left[\bigwedge_{i\in\mathcal{N}_\sigma} P_i(s) = P_i(s_i)\right] \\ &\quad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} \left[\begin{array}{l} s_i\sigma \in L(E) \wedge s \in L(E) \wedge s\sigma \in L(G) \\ \Rightarrow s\sigma \in L(E) \end{array}\right] \end{aligned}$$

Notice we have also quantified all i's over $\mathcal{N}_\sigma$ instead of $\mathcal{N}$, since for $i \in \mathcal{N} \setminus \mathcal{N}_\sigma$ the statement is vacuously satisfied as the consequent of (3.1) is quantified only over $i \in \mathcal{N}_\sigma$.

Since i does not occur freely in $s \in L(E) \wedge s\sigma \in L(G)$, we proceed to extract the term outwards. Using the logic identity $\bigvee_x [P \Rightarrow Q(x)] \Leftrightarrow [P \Rightarrow \bigvee_x Q(x)]$, where x does not occur freely in P, together with currying and commutativity, we have

equivalently

$$\begin{array}{l}\forall\,\sigma \in \Sigma_c\,.\;\forall\,s, \{\, s_i \,\}_{i\in\mathcal{N}_\sigma}\,.\\ \quad\left[\bigwedge_{i\in\mathcal{N}_\sigma} P_i(s) = P_i(s_i)\right]\\ \quad\Rightarrow s \in L(E) \wedge s\sigma \in L(G)\\ \qquad\Rightarrow \bigvee_{i\in\mathcal{N}_\sigma}\left[\begin{array}{l} s_i\sigma \in L(E)\\ \Rightarrow s\sigma \in L(E)\end{array}\right]\end{array}$$

By applying modus tollens, we have equivalently

$$\begin{array}{l}\forall\,\sigma \in \Sigma_c\,.\;\forall\,s, \{\, s_i \,\}_{i\in\mathcal{N}_\sigma}\,.\\ \quad\left[\bigwedge_{i\in\mathcal{N}_\sigma} P_i(s) = P_i(s_i)\right]\\ \quad\Rightarrow s \in L(E) \wedge s\sigma \in L(G)\\ \qquad\Rightarrow \bigvee_{i\in\mathcal{N}_\sigma}\left[\begin{array}{l} s\sigma \notin L(E)\\ \Rightarrow \neg(s_i\sigma \in L(E))\end{array}\right]\end{array}$$

Notice that we wrote $\neg(s_i\sigma \in L(E))$ instead of $s_i\sigma \notin L(E)$ so that further on in our development the format will allow us to exploit the fact that $s_i\sigma \in L(E)$ implies $s_i\sigma \in L(G)$.

Again, we have a term $s\sigma \notin L(E)$ in which i does not occur free, so we extract it outwards and have equivalently

$$\begin{array}{l}\forall\,\sigma \in \Sigma_c\,.\;\forall\,s, \{\, s_i \,\}_{i\in\mathcal{N}_\sigma}\,.\\ \quad\left[\bigwedge_{i\in\mathcal{N}_\sigma} P_i(s) = P_i(s_i)\right]\\ \quad\Rightarrow s \in L(E) \wedge s\sigma \in L(G) \wedge s\sigma \notin L(E)\\ \qquad\Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} \neg(s_i\sigma \in L(E))\end{array}$$

Since $L(E)$ is prefix-closed and $L(E) \subseteq L(G)$, the statement $s_i\sigma \in L(E)$ is equivalent

to $s_i \in L(E) \wedge s_i\sigma \in L(G) \wedge s_i\sigma \in L(E)$. Hence we have equivalently

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, s, \{\, s_i \,\}_{i\in\mathcal{N}_\sigma}\,.\\
&\quad \left[\bigwedge_{i\in\mathcal{N}_\sigma} P_i(s) = P_i(s_i)\right]\\
&\quad \Rightarrow s \in L(E) \wedge s\sigma \in L(G) \wedge s\sigma \notin L(E)\\
&\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} [s_i \notin L(E) \vee s_i\sigma \notin L(G) \vee s_i\sigma \notin L(E)]
\end{aligned}$$

We then extract $s \in L(E)$ and $s_i \in L(E)$ outwards. For $s_i \in L(E)$, since i occurs freely in it, we use the logic identity $[\bigvee_x (P(x) \vee Q(x))] \Leftrightarrow [\bigvee_x P(x) \vee \bigvee_x Q(x)] \Leftrightarrow [\bigwedge_x \neg P(x) \Rightarrow \bigvee_x Q(x)]$, and hence have equivalently

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, s \in L(E), \{\, s_i \in L(E) \,\}_{i\in\mathcal{N}_\sigma}\,.\\
&\quad \left[\bigwedge_{i\in\mathcal{N}_\sigma} P_i(s) = P_i(s_i)\right]\\
&\quad \Rightarrow s\sigma \in L(G) \wedge s\sigma \notin L(E)\\
&\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} \left[\begin{array}{l} s_i\sigma \in L(G)\\ \Rightarrow s_i\sigma \notin L(E)\end{array}\right]
\end{aligned} \tag{3.2}$$

We can observe that if $s, s_i \in L(E)$, there must be $w, w' \in Q'$ such that $w_e, w'_e \in Q^E$ and $\delta'(q'_0, s) = w, \delta'(q'_0, s_i) = w'$. Also, if $P_i(s) = P_i(s_i)$ and $s, s_i \in L(E)$, must have $w' \sim_i w$. The reverse directions of the two implications are reasoned similarly. Hence expression (3.2) is equivalent to

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,.\\
&\quad [(I, w) \models \sigma_G \wedge \neg\, \sigma_E]\\
&\quad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} \forall\, w' \sim_i w\,.\ [(I, w') \models \sigma_G \Rightarrow \neg\, \sigma_E]
\end{aligned}$$

which is equivalent to

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,.\\
&\quad (I, w) \models \sigma_G \wedge \neg\, \sigma_E\\
&\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \neg\, \sigma_E)
\end{aligned}$$

By $\sigma_E \Rightarrow \sigma_G$ the expression above can be equivalently simplified to

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\, \forall\, w \in Q' \text{ such that } w_e \in Q^E\,. \\ &\quad (I, w) \models \sigma_G \wedge \neg\, \sigma_E \\ &\qquad\qquad \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \end{aligned} \tag{3.3}$$

which is an epistemic expression of C&P co-observability.

With an epistemic expression, we interpret C&P co-observability as follows: The expression (3.3) says, after a legal sequence s, for an event σ, if $(I, w) \models (\sigma_G \wedge \neg\, \sigma_E)$ (i.e., not only is σ possible to happen, but it is also illegal and hence must be disabled), then a supervisor i controlling σ can correctly disable it by knowing that disabling σ does not violate the control requirement. If none of the supervisors disables σ, then either σ cannot happen after s (signified by $\neg\, \sigma_G$), or if it can happen, it must be legal (signified by σ_E), and hence the fused decision of σ can be defaulted to enable.

Specifically, in the conjunctive architecture, the control decision **1** does not actively enable an event. It merely indicates that a supervisor issuing such a decision cannot confidently disable the event and abstains from voting. This asymmetry between the two control decisions **0** and **1** corresponds to the property of Boolean conjunction that the operation yields **0** as soon as a **0** operand is present, and it only yields **1** when all operands are **1**.

Therefore, it is more appropriate to relate the control decisions **0** and **1** to the control decisions **off** and **abstain** used in our earlier work [RR23]. We will formally define the semantics (through the fusion rule) of these two control decisions in Section 3C6.

Readers familiar with the work of Ricker and Rudie [RR00] may notice a similarity between (3.3) and the expression of Kripke-observability discussed by them. We now proceed to discuss their relationship.

One of the two ways in which Kripke-observability differs from (3.3) is that (3.3) quantifies over only controllable events, while Kripke-observability quantifies over all events. We emphasize that this is not a shortcoming of our expression. We proceed by showing that Kripke-observability, in a sense, has combined two orthogonal conditions: C&P co-observability and controllability.

Definition 3C1.1 (Controllability [RW87])
A prefix-closed language $L(E) \subseteq L(G)$ is said to be controllable w.r.t. G and Σ_{uc} whenever

$$L(E)\Sigma_{uc} \cap L(G) \subseteq L(E)$$

or equivalently,

$$\forall\, \sigma \in \Sigma_{uc}.\ s \in L(E) \wedge s\sigma \in L(G) \Rightarrow s\sigma \in L(E)$$

Hence, C&P co-observability together with controllability is equivalent to

$$\begin{aligned}\forall\, \sigma \in \Sigma.\qquad \sigma \in \Sigma_c \Rightarrow \forall\, w \in Q' \text{ such that } w_e \in Q^E.\\ (I, w) \models \sigma_G \wedge \neg\, \sigma_E\\ \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E)\\ \wedge\ \sigma \in \Sigma_{uc} \Rightarrow \forall\, w \in Q' \text{ such that } w_e \in Q^E.\\ (I, w) \models \neg\, \sigma_G \vee \sigma_E\end{aligned}$$

where the first conjunction is C&P co-observability, and the second conjunction is controllability.

Since $\neg\, \sigma_G \vee \sigma_E$ is equivalent to $(\sigma_G \wedge \neg\, \sigma_E) \Rightarrow \bot$, and for $\sigma \in \Sigma_{uc}$ and any expression ϕ, $\bigvee_{\mathcal{N}_\sigma} \phi = \bigvee_{\varnothing} \phi = \bot$, the above is equivalent to

$$\begin{aligned}\forall\, \sigma \in \Sigma.\qquad \sigma \in \Sigma_c \Rightarrow \forall\, w \in Q' \text{ such that } w_e \in Q^E.\\ (I, w) \models \sigma_G \wedge \neg\, \sigma_E\\ \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E)\\ \wedge\ \sigma \in \Sigma_{uc} \Rightarrow \forall\, w \in Q' \text{ such that } w_e \in Q^E.\\ (I, w) \models \sigma_G \wedge \neg\, \sigma_E\\ \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E)\end{aligned}$$

and hence equivalent to

$$\begin{aligned}\forall\, \sigma \in \Sigma.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E.\\ (I, w) \models \sigma_G \wedge \neg\, \sigma_E\\ \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E)\end{aligned}$$

This expression differs from that of Kripke-observability [RR00] only in the Kripke structure used: the Kripke structure used by the same authors [RR00] is infinite if the plant language contains infinitely many strings. The later work by Ricker and Rudie [RR07] does use a finite structure, but does not explicitly state that the approach can be applied to Kripke-observability.

While one can combine C&P co-observability with controllability into one single clean and compact expression, we refrain from doing so: as we will demonstrate with D&A co-observability as an example, controllability condition does not usually combine nicely with observability-related ones.

3C2 D&A co-observability

To obtain an epistemic expression of D&A co-observability, we begin from the following definition of D&A co-observability [PKK97]:

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, s, \{\, s_i \,\}_{i\in\mathcal{N}_\sigma}\,.\\
&\quad [\forall\, i \in \mathcal{N}_\sigma\,.\ P_i(s) = P_i(s_i)]\\
&\quad \Rightarrow s \in L(E) \wedge s\sigma \in L(E)\\
&\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} \left[\begin{array}{l} s_i \in L(E) \wedge s_i\sigma \in L(G)\\ \Rightarrow s_i\sigma \in L(E)\end{array}\right]
\end{aligned}$$

Following similar steps as we did in Section 3C1, namely by reordering logic connectives and converting from language-theoretical expressions to epistemic-logic expressions, we have equivalently

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,.\\
&\quad (I, w) \models \sigma_G \wedge \sigma_E\\
&\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E)
\end{aligned} \tag{3.4}$$

which is an epistemic expression of D&A co-observability. Notice that we intentionally refrained from contracting $\sigma_G \wedge \sigma_E$ into σ_E for uniformity with other expressions.

With an epistemic expression, we interpret D&A co-observability as follows: The expression (3.4) says, after a legal sequence s, for an event σ, if $(I, w) \models \sigma_E$ (i.e. not only is σ possible to happen, but it is also legal and hence must be enabled), then a supervisor i controlling σ can correctly enable it by knowing that enabling σ does not violate the control requirement. If none of the supervisors enable σ, then either σ cannot happen after s, or if it can happen, it must be illegal, and hence the fused decision of σ can be defaulted to disable.

Specifically, in the conjunctive architecture, the control decision **0** does not actively disable an event. It merely indicates that a supervisor issuing such a decision cannot confidently enable the event and abstains from voting. This asymmetry between the

two control decisions **0** and **1** corresponds to the property of Boolean disjunction that the operation yields **1** as soon as a **1** operand is present, and it only yields **0** when all operands are **0**.

Therefore, it is more appropriate to relate the control decisions **0** and **1** to the control decisions **abstain** and **on** used in our earlier work [RR23]. We will formally define the semantics (through the fusion rule) of these two control decisions in Section 3C6.

The case of D&A co-observability demonstrates the reason that we refrain from combining controllability condition with co-observability conditions into one expression. On the one hand, unlike the case of C&P co-observability, there is no compact expression that expresses both D&A co-observability and controllability. Similar comments can be made to other co-observability conditions as well. On the other hand, if we begin from

$$\begin{aligned} &\forall\, \sigma \in \Sigma\,.\, \forall\, w \in Q'\,. \\ &\quad (I, w) \models \sigma_E \\ &\qquad\qquad \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \end{aligned}$$

we'd then have equivalently

$$\begin{aligned} \forall\, \sigma \in \Sigma\,. \quad & \sigma \in \Sigma_c \Rightarrow \ \ldots \\ & \wedge\, \sigma \in \Sigma_{uc} \Rightarrow \forall\, w \in Q'\,. \\ & \qquad\qquad (I, w) \models \neg\, \sigma_E \end{aligned}$$

where the second part can be equivalently expressed as

$$\forall\, \sigma \in \Sigma_{uc}\,.\ s \in L(G) \Rightarrow s\sigma \notin L(E)$$

or equivalently

$$L(G)\Sigma_{uc} \cap L(E) = \varnothing$$

which expresses that no legal string ends with an uncontrollable event, or, any string ends with an uncontrollable event is illegal.

3C3 Strong C&P co-observability

To obtain an epistemic expression of strong C&P co-observability, we begin from the following definition of strong C&P co-observability (Takai and Ushio [TU01,

Proposition 1], Takai, Kumar, and Ushio [TKU05]):

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, s, \{\, s_i \,\}_{i\in\mathcal{N}_\sigma}\,.\\
&\quad [\forall\, i \in \mathcal{N}_\sigma\,.\ P_i(s) = P_i(s_i)]\\
&\quad \Rightarrow s \in L(G) \wedge s\sigma \in L(G) - L(E)\\
&\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} \left[\begin{array}{l} s_i \in L(G) \wedge s_i\sigma \in L(G)\\ \Rightarrow s_i\sigma \notin L(E) \end{array}\right]
\end{aligned} \tag{3.5}$$

We can observe that if $s, s_i \in L(G)$, there must be $w, w' \in Q'$ and $\delta'(q'_0, s) = w, \delta'(q'_0, s_i) = w'$. Also, if $P_i(s) = P_i(s_i)$ and $s, s_i \in L(G)$, must have $w' \simeq_i w$. The reverse directions of the two implications are reasoned similarly. Hence the expression (3.5) is equivalent to

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q'\,.\\
&\quad [(\overline{I}, w) \models \sigma_G \wedge \neg\, \sigma_E]\\
&\quad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} \forall\, w' \simeq_i w\,.\ [(\overline{I}, w') \models \sigma_G \Rightarrow \neg\, \sigma_E]
\end{aligned}$$

which is equivalent to

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q'\,.\\
&\quad (\overline{I}, w) \models \sigma_G \wedge \neg\, \sigma_E\\
&\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} K_i(\sigma_G \Rightarrow \neg\, \sigma_E)
\end{aligned}$$

By $\sigma_E \Rightarrow \sigma_G$ the expression above can be equivalently simplified to

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q'\,.\\
&\quad (\overline{I}, w) \models \sigma_G \wedge \neg\, \sigma_E\\
&\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} K_i(\neg\, \sigma_E)
\end{aligned} \tag{3.6}$$

Compare the epistemic expression of strong C&P co-observability (3.6) with that of C&P co-observability (3.3). The only difference is between the Kripke structures I and $\overline{I}$, or more specifically, the accessibility relations $\sim_i$ and $\simeq_i$. More is required for an agent to "know" something under the interpretation $\overline{I}$ than under the interpretation I.

To formalize this point, the condition

$$(\overline{I}, w) \models K_i(\phi)$$

which is equivalent to

$$\forall\, w' \simeq_i w\,.\ (\overline{I}, w') \models \phi$$

and further equivalent to

$$\forall\, w' \text{ s.t. } w'_i = w_i\,.\ (\overline{I}, w') \models \phi$$

which is strictly stronger than

$$\begin{aligned}\forall\, w' \text{ s.t. } & w'_i = w_i \text{ and } w'_e \in Q_E \text{ and } w_e \in Q_E\,.\\ & (I, w') \models \phi,\end{aligned}$$

which is equivalent to

$$(I, w) \models K_i(\phi).$$

Informally, strong C&P co-observability requires correct control decisions even at illegal states. This is more than necessary since illegal states are not reachable if the language is C&P co-observable and hence the desired control requirement can be met. This is exactly how much stronger strong C&P co-observability is than C&P co-observability.

Since strong C&P co-observability is a stronger version of C&P co-observability, one may expect that it too can be combined with controllability. However, this is not the case.

Instead, if we begin from

$$\begin{aligned}\forall\, \sigma \in \Sigma\,.\ \forall\, w \in Q'\,.&\\ (\overline{I}, w) \models \sigma_G \wedge \neg\, \sigma_E & \\ \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} & K_i(\neg\, \sigma_E)\end{aligned}$$

we'd then have equivalently

$$\begin{aligned}\forall\, \sigma \in \Sigma\,. \quad \sigma \in \Sigma_c \Rightarrow \forall\, w \in Q'\,.& \\ (\overline{I}, w) \models \sigma_G \wedge \neg\, \sigma_E & \\ \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} & K_i(\neg\, \sigma_E)\\ \wedge\, \sigma \in \Sigma_{uc} \Rightarrow \forall\, w \in Q'\,.& \\ (\overline{I}, w) \models \neg\, \sigma_G \vee \sigma_E &\end{aligned}$$

where the first part is strong C&P co-observability.

The second part can be equivalently expressed as

$$\forall\, \sigma \in \Sigma_{uc}\,.\ s \in L(G) \wedge s\sigma \in L(G) \Rightarrow s\sigma \in L(E)$$

or equivalently

$$L(G)\Sigma_{uc} \cap L(G) = L(E)$$

which requires all strings ending with uncontrollable events to be legal. The expression appears very similar to controllability, hence we call it strong controllability. Its relation to controllability is exactly like that between strong C&P co-observability and C&P co-observability.

3C4 Strong D&A co-observability

To obtain an epistemic expression of the strong D&A co-observability, we begin from the following definition of strong D&A co-observability (Takai and Ushio [TU01, Proposition 1], Takai, Kumar, and Ushio [TKU05]):

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\, \forall\, s, \{\, s_i \,\}_{i\in\mathcal{N}}\,.\\
&\quad [\forall\, i \in \mathcal{N}_\sigma\,.\, P_i(s) = P_i(s_i)]\\
&\quad \Rightarrow s \in L(G) \wedge s\sigma \in L(E)\\
&\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} \left[\begin{array}{l} s_i \in L(G) \wedge s_i\sigma \in L(G)\\ \Rightarrow s_i\sigma \in L(E)\end{array}\right]
\end{aligned}$$

Following similar steps as we did in Section 3C3, namely by reordering logic connectives and converting from language-theoretical expressions to epistemic-logic expressions, we have equivalently

$$\begin{aligned}
&\forall\, \sigma \in \Sigma_c\,.\, \forall\, w \in Q'\,.\\
&\quad (\overline{I}, w) \models \sigma_G \wedge \sigma_E\\
&\qquad\qquad \Rightarrow \bigvee_{i\in\mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E)
\end{aligned} \tag{3.7}$$

In the expression above, it is possible to further restrict the quantification of w, equivalently, to $\forall w \in Q'$ such that $w_e \in Q^E$. However we refrain from doing so to keep the expression analogous to that of (3.6).

3C5 C&P∧D&A co-observability

Say that the prefix-closed language $L(E)$ is C&P∧D&A co-observable if $L(E)$ is both C&P co-observable and D&A co-observable [TKU05]. Hence C&P∧D&A co-

observability is equivalent to

$$\begin{bmatrix} \forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,. \\ \quad (I, w) \models \sigma_G \wedge \neg\, \sigma_E \\ \qquad\qquad \Rightarrow \bigvee\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \end{bmatrix} \\ \wedge \begin{bmatrix} \forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,. \\ \quad (I, w) \models \sigma_G \wedge \sigma_E \\ \qquad\qquad \Rightarrow \bigvee\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \end{bmatrix}$$

which is equivalent to

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,. \\ &(I, w) \models \sigma_G \Rightarrow \begin{bmatrix} \neg\, \sigma_E \Rightarrow \bigvee\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \\ \wedge \quad \sigma_E \Rightarrow \bigvee\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \end{bmatrix} \end{aligned} \tag{3.8}$$

The expression (3.8) can be interpreted as follows: if an event is possible, then if it's not legal some agent knows it can be disabled (because it is illegal), and if it's legal some agent knows it can be enabled (either because it is legal or cannot even occur).

More compactly, we can write

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,. \\ &(I, w) \models \sigma_G \Rightarrow \begin{bmatrix} \bigvee\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \\ \vee \bigvee\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \end{bmatrix} \end{aligned}$$

Finally, we proceed to remove the superfluous $\sigma_G \Rightarrow \ldots$. Consider a world $w \in Q'$ such that $w_e \in Q^E$. Consider also all worlds $w' \in [w]_i$ for some i. If at some w' we have σ_G, then expression in the square brackets holds at w. If at all w' we have $\neg\, \sigma_G$, then have $K_i(\neg\, \sigma_G)$ at w, so the aforementioned expression in square brackets also holds at w. Hence, we have equivalently

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,. \\ &(I, w) \models \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \\ &\qquad\qquad \vee \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \end{aligned} \tag{3.9}$$

With an epistemic expression, we interpret C&P∧D&A co-observability as follows: after a legal sequence s, for an event σ, if $(I, w) \models \sigma_G$ (i.e. σ is possible to happen, hence a decision about σ is obliged), then a supervisor can make a correct decision.

Specifically, in the C&P∧D&A architecture, three control decisions are used: **on**, **off** and **abstain**, where the fused decision is **enable** as soon as a **on** is issued; **disable** as soon as a **off** is issued.

The condition guarantees that at least one **on** or at least one **off** is issued, so it is not the case that all supervisors **abstain**. And it is physically impossible that both **on** and **off** is issued. Together the fusion rule is well-defined.

3C6 C&P∨D&A co-observability

We proceed to derive an epistemic logic expression of C&P∨D&A co-observability as follows:

Definition 3C6.1 (C&P∨D&A co-observability [YL02])
The language $L(E)$ is C&P∨D&A co-observable if there exists a partition $\{\Sigma_{c,e}, \Sigma_{c,d}\}$ of Σ_c such that $L(E)$ is C&P co-observable with respect to $\Sigma_{c,e}$ and D&A co-observable with respect to $\Sigma_{c,d}$

By (3.3) and (3.4), the C&P∨D&A co-observability of the language $L(E)$ is equivalent to

$$\begin{aligned} \forall\, \sigma \in \Sigma_c. \quad & \forall\, w \in Q' \text{ such that } w_e \in Q^E. \\ & (I, w) \models \neg(\neg\, \sigma_G \vee \sigma_E) \\ & \qquad \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \\ \vee\, & \forall\, w \in Q' \text{ such that } w_e \in Q^E. \\ & (I, w) \models \neg(\neg\, \sigma_E) \\ & \qquad \Rightarrow \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \end{aligned} \tag{3.10}$$

There are two approaches to derive more tidy expressions facilitating informal interpretation. The first approach, similar to our earlier work [RR23], provides a more compact expression and interpretation, whereas the second approach provides additional insights.

Approach 1

Since whenever $\neg(\neg\,\sigma_G \vee \sigma_E)$, it cannot be the case that $K_i(\neg\,\sigma_G \vee \sigma_E)$; and similarly, whenever $\neg(\neg\,\sigma_E)$, it cannot be $K_i(\neg\,\sigma_E)$. Hence, we can reach a simpler epistemic expression of C&P∨D&A co-observability.

Definition 3C6.2
Let $\phi_\sigma^+ = \neg\,\sigma_G \vee \sigma_E$, $\phi_\sigma^- = \neg\,\sigma_E$.

The language $L(E)$ is said to be C&P∨D&A co-observable whenever for any $\sigma \in \Sigma_c$, there is a certain $\phi_\sigma \in \{\,\phi_\sigma^+, \phi_\sigma^-\,\}$ for this σ, so that for all $w \in W$ such that $w_e \in Q^E$, we have

$$(I, w) \models \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\,\sigma_G \vee \sigma_E) \tag{3.11.1}$$

$$\vee \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\,\sigma_E) \tag{3.11.2}$$

$$\vee \phi_\sigma \tag{3.11.3}$$

Following Ritsuka and Rudie [RR23] we know that the language $L(E)$ being C&P∨D&A co-observable is the necessary and sufficient condition for the DSCOP with the following architecture to be solvable: the set of control decisions $\mathcal{CD}$ is $\{$ **on**, **off**, **abstain** $\}$, the fusion rule $f_\sigma \in \{\,f^+, f^-\,\}$ can be chosen for each $\sigma \in \Sigma_c$, where f^+, f^- are defined as in Fig. 3.1. We have to emphasize that even though there are three control decisions, the two decisions **on**, **off** will never be issued for an event σ simultaneously at a state where the event σ is physically possible (i.e., at a state where σ_G holds).

- $\exists\, i \in \mathcal{N}_\sigma\,.\; cd_i =$ **on** $\Rightarrow f^*(s,\sigma) =$ **enable**
- $\exists\, i \in \mathcal{N}_\sigma\,.\; cd_i =$ **off** $\Rightarrow f^*(s,\sigma) =$ **disable**
- Otherwise, $f^+(s,\sigma) =$ **enable**, $f^-(s,\sigma) =$ **disable**.

Figure 3.1: Fusion rule, where $cd_i = f_i(P_i(s),\sigma)$ for short and f^* stands for both f^+ and f^-.

A control policy can be synthesized following the epistemic expression. For each agent i, when the plant is in State q (so by construction agent i is in State q_i^{obs} and $q \in q_i^{obs}$), for every event σ that agent i controls, agent i should issue control decision **on** if (3.11.1) holds for i; **off** if (3.11.2) holds for i; **abstain** if otherwise (i.e., only (3.11.3) holds for i).

Hence for each event σ and the states such that neither (3.11.1) nor (3.11.2) holds for some i (i.e., the states in which none of the agents can make a decision), if either in all these states (3.11.3) holds for ϕ_σ^+ (in which case we take the default action **enable**), or in all these states (3.11.3) holds for ϕ_σ^- (in which case we take the default action **disable**), so that the default action is unambiguous, then the control requirement is achievable.

Approach 2

The second approach provides some additional insights. Specifically, we realized that in the previous sections, the control decision **abstain** has been used not only when a supervisor wishes to assert no influence over the fused decision. To demonstrate this, we derive alternative expressions to separate the other semantics from the decision **abstain**. After separating the semantics, we will demonstrate that it is without loss of generality to require that never will all supervisors abstain, and thus eliminate the need for choosing a default decision for each event.

To simplify the discussion, we first look back to the expression (3.3) of C&P co-observability. We give yet another expression for C&P co-observability.

Theorem 3C6.3
Expression (3.3) is equivalent to the following expression:

$$\forall\, \sigma \in \Sigma_c\,.\, \forall\, w \in Q' \text{ such that } w_e \in Q^E\,.$$

$$(I, w) \models \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \tag{3.12.1}$$

$$\vee \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \tag{3.12.2}$$

$$\vee \bigvee_{i \in \mathcal{N}_\sigma} K_i(\sigma_G \wedge \neg\, \sigma_E \Rightarrow \bigvee_{\substack{j \in \mathcal{N}_\sigma \\ j \neq i}} K_j(\neg\, \sigma_E)) \tag{3.12.3}$$

Before we proceed to a proof, we explain the requirement of $j \neq i$: this requirement is actually redundant in (3.12.3), however, keeping this requirement creates a line-by-line correspondence between the co-observability condition (the problem solvability condition) and the control protocol, as discussed in Ritsuka and Rudie [RR23]. The following lemma, which is a generalization of Lemma 3.5 and Lemma 3.6 in Ritsuka and Rudie [RR23], can be readily proven:

Lemma 3C6.4

$$(I, w) \models K_i(\sigma_E \Rightarrow \bigvee_{j \in \mathcal{N}_\sigma} K_j(\neg\, \sigma_G \vee \sigma_E)) \tag{3.13}$$

iff

$$\begin{aligned}(I,w) \models\ & K_i(\neg\,\sigma_E) \\ & \vee K_i(\neg\,\sigma_G \vee \sigma_E) \\ & \vee K_i(\sigma_E \Rightarrow \bigvee_{\substack{j\in\mathcal{N}_\sigma \\ j\neq i}} K_j(\neg\,\sigma_G \vee \sigma_E))\end{aligned} \tag{3.14}$$

The proof proceeds analogously to those of Lemma 3.5 and Lemma 3.6 in Ritsuka and Rudie [RR23].

Proof. ($\Leftarrow$): We have

$$\begin{aligned}&(I,w) \models K_i(\neg\,\sigma_E) \\ \Rightarrow &(I,w) \models K_i(\neg\,\sigma_E \vee \bigvee_{j\in\mathcal{N}_\sigma} K_j(\neg\,\sigma_G \vee \sigma_E)) \\ \Rightarrow &(I,w) \models K_i(\sigma_E \Rightarrow \bigvee_{j\in\mathcal{N}_\sigma} K_j(\neg\,\sigma_G \vee \sigma_E))\end{aligned}$$

Moreover,

$$\begin{aligned}&(I,w) \models K_i(\neg\,\sigma_G \vee \sigma_E) \\ \Rightarrow &\forall\, w' \in [w]_i\,.\ (I,w') \models K_i(\neg\,\sigma_G \vee \sigma_E) \\ \Rightarrow &\forall\, w' \in [w]_i\,.\ (I,w') \models \bigvee_{j\in\mathcal{N}_\sigma} K_j(\neg\,\sigma_G \vee \sigma_E) \\ \Rightarrow &\forall\, w' \in [w]_i\,.\ (I,w') \models \sigma_E \Rightarrow \bigvee_{j\in\mathcal{N}_\sigma} K_j(\neg\,\sigma_G \vee \sigma_E) \\ \Rightarrow &(I,w) \models K_i(\sigma_E \Rightarrow \bigvee_{j\in\mathcal{N}_\sigma} K_j(\neg\,\sigma_G \vee \sigma_E))\end{aligned}$$

The last case is straightforward.

($\Rightarrow$):

Assume $(I,w) \models K_i(\sigma_E \Rightarrow \bigvee_{j\in\mathcal{N}_\sigma} K_i(\neg\,\sigma_G \vee \sigma_E))$.

Hence we have equivalently

$\forall\, w' \in [w]_i\,.\ (I,w') \models \sigma_E \Rightarrow \bigvee_{j\in\mathcal{N}_\sigma} K_i(\neg\,\sigma_G \vee \sigma_E)$.

Hence we have equivalently

$\forall\, w' \in [w]_i\,.\ (I,w') \models \neg\,\sigma_E \vee \bigvee_{j\in\mathcal{N}_\sigma} K_i(\neg\,\sigma_G \vee \sigma_E)$. $(*)$

Therefore, either

A: $w_e \in Q^E$; or

B: $w_e \notin Q^E$.

Case A: $w_e \in Q^E$

Hence $[w]_i$ is not empty.

We have either

A.1: $\exists\, w' \in [w]_i\,.\ (I, w') \models K_i(\neg\, \sigma_G \vee \sigma_E)$; or
A.2: $\neg\, \exists\, w' \in [w]_i\,.\ (I, w') \models K_i(\neg\, \sigma_G \vee \sigma_E)$
// The trick here is to not split the disjunctions in $(*)$.
Case A.1: $\exists\, w' \in [w]_i\,.\ (I, w') \models K_i(\neg\, \sigma_G \vee \sigma_E)$
Obtain w' such that
$w' \in [w]_i$ and
$(I, w') \models K_i(\neg\, \sigma_G \vee \sigma_E)$.
Hence $\forall\, w'' \in [w']_i\,.\ (I, w'') \models K_i(\neg\, \sigma_G \vee \sigma_E)$.
With $[w']_i = [w]_i$,
we have $\forall\, w'' \in [w]_i\,.\ (I, w'') \models K_i(\neg\, \sigma_G \vee \sigma_E)$.
Hence $(I, w) \models K_i(\neg\, \sigma_G \vee \sigma_E)$.
Case A.2: $\neg\, \exists\, w' \in [w]_i\,.\ (I, w') \models K_i(\neg\, \sigma_G \vee \sigma_E)$
Hence $\forall\, w' \in [w]_i\,.\ (I, w') \models \neg\, K_i(\neg\, \sigma_G \vee \sigma_E)$.
With $(*)$,
we have $\forall\, w' \in [w]_i\,.\ (I, w') \models \neg\, \sigma_E \vee \bigvee_{\substack{j \in \mathcal{N}_\sigma \\ j \neq i}} K_j(\neg\, \sigma_G \vee \sigma_E)$.
Thus $(I, w) \models K_i(\neg\, \sigma_E \vee \bigvee_{\substack{j \in \mathcal{N}_\sigma \\ j \neq i}} K_j(\neg\, \sigma_G \vee \sigma_E))$.
Case B: $w_e \notin Q^E$
Hence $[w]_i$ is empty.
Hence $\forall\, w' \in [w]_i\,.\ (I, w') \models \phi$ holds vacuously true.
Hence $(I, w) \models K_i(\phi)$ holds vacuously true.

□

We return now to the proof of Thm. 3C6.3.

Proof (Thm. 3C6.3). Consider an arbitrary event $\sigma \in \Sigma_c$ and an arbitrary world $w \in Q'$ such that $w_e \in Q^E$.

((3.3) $\Rightarrow$ (3.12)) Suppose that for some $i \in \mathcal{N}_\sigma$, at some $w' \in [w]_i$, it is the case that $K_i(\neg\, \sigma_E)$. Thus we also have $K_i(\neg\, \sigma_E)$ (3.12.1) at w.

Suppose that for no $i \in \mathcal{N}_\sigma$, at no world $w' \in [w]_i$, it is the case that $K_i(\neg\, \sigma_E)$. Then at all $w' \in [w]_i$ it must be $\neg\, \sigma_G \vee \sigma_E$. Thus for any $i \in \mathcal{N}_\sigma$, we have $K_i(\neg\, \sigma_G \vee \sigma_E)$ (3.12.2) at w.

((3.3) $\Leftarrow$ (3.12)) Suppose that for some $i \in \mathcal{N}_\sigma$, we have $K_i(\neg_E)$. Then (3.3) is automatic.

Suppose that for some $i \in \mathcal{N}_\sigma$, we have $K_i(\neg\, \sigma_G \vee \sigma_E)$. Since $w_e \in Q^E$, so $w \in [w]_i$, hence we also have $\neg\, \sigma_G \vee \sigma_E$. Thus expression (3.3) holds.

Suppose that for some $i \in \mathcal{N}_\sigma$, we have $K_i(\sigma_G \wedge \neg\, \sigma_E \Rightarrow \bigvee_{\substack{j \in \mathcal{N}_\sigma \\ j \neq i}} K_j(\neg\, \sigma_E))$. Since

$w_e \in Q^E$, so $w \in [w]_i$, hence we also have $\sigma_G \wedge \neg\,\sigma_E \Rightarrow \bigvee_{\substack{j\in\mathcal{N}_\sigma \\ j\neq i}} K_j(\neg\,\sigma_E)$, which implies (3.3). □

Notice that all three disjunctions in (3.12) are epistemic formulae, unlike (3.3) where there is a propositional disjunction. Very informally, (3.12.3) comes from the fact that whenever the language is C&P co-observable, all supervisors know it is so. That is, C&P co-observability is equivalent to

$$\begin{aligned} &\forall\,\sigma \in \Sigma_c\,.\,\forall\,w \in Q' \text{ such that } w_e \in Q^E\,. \\ &\quad (I,w) \models \bigwedge_{i\in\mathcal{N}_\sigma} K_i(\sigma_G \wedge \neg\,\sigma_E \Rightarrow \bigvee_{j\in\mathcal{N}_\sigma} K_j(\neg\,\sigma_E)) \end{aligned} \tag{3.15}$$

which is a fact that can be used in an alternative proof for Thm. 3C6.3.

We can apply the same technique to the expression (3.4) of D&A co-observability as well. Together with the expression (3.12), we can derive an alternative definition of C&P∧D&A co-observability.

Definition 3C6.5 (alternative to Defn. 3C6.2)
Let

$$\psi_\sigma^+ = \bigvee_{i\in\mathcal{N}_\sigma} K_i(\sigma_G \wedge \neg\sigma_E \Rightarrow \bigvee_{\substack{j\in\mathcal{N}_\sigma \\ j\neq i}} K_j(\qquad\quad \neg\,\sigma_E)) \tag{3.16}$$

$$\psi_\sigma^- = \bigvee_{i\in\mathcal{N}_\sigma} K_i(\qquad\quad \sigma_E \Rightarrow \bigvee_{\substack{j\in\mathcal{N}_\sigma \\ j\neq i}} K_j(\neg\,\sigma_G \vee \quad \sigma_E)) \tag{3.17}$$

The language $L(E)$ is said to be C&P∨D&A co-observable whenever for any $\sigma \in \Sigma_c$, there is a certain $\psi_\sigma \in \{\,\psi_\sigma^+, \psi_\sigma^-\,\}$ for this σ, so that for all $w \in W$ such that $w_e \in Q^E$, we have

$$(I,w) \models \bigvee_{i\in\mathcal{N}_\sigma} K_i(\neg\,\sigma_G \vee \sigma_E) \tag{3.18.1}$$

$$\vee \bigvee_{i\in\mathcal{N}_\sigma} K_i(\neg\,\sigma_E) \tag{3.18.2}$$

$$\vee\psi_\sigma \tag{3.18.3}$$

What makes (3.12) and Defn. 3C6.5 interesting is that they reveal two distinct semantics of the control decision **abstain** used in (3.3) and Defn. 3C6.2. Specifically, other than indicating a supervisor's intention to assert no influence over the fused

decision, there is a kind of *conditional decision* — the name is coined by Yoo and Lafortune [YL04] — but this analysis shows that conditional decisions exist since the dawn of decentralized control, when the first decentralized architecture, the C&P architecture, is studied [RW92].

The conditional decision is associated with higher-order knowledge: not only can a supervisor introspect its own knowledge, but it can also rely on the knowledge of other supervisors. Specifically, taking the expression (3.12.3) and (3.16) as an example, it says that Supervisor i can "conditionally enable" the event σ, as it knows, if enabling σ is a mistake, there will be some other Supervisor j to correct its mistake by disabling σ. Since in this case we would like Supervisor j's decision to override that of Supervisor i, Supervisor i's decision must be "weaker", which is to be formally reflected in the fusion rule. Similarly, the expression (3.17) talks about the use of a "conditionally disable" control decision. We denote the conditional enable (resp., disable) decision as **weak on** (resp., **weak off**).

Therefore, although historically the C&P architecture is introduced first [RW90], with the D&A architecture derived as its dual [PKK97] and C&P∧D&A architecture and C&P∨D&A derived as Boolean combinations of the previous two, we propose a different perspective regarding the four architectures.

Recall what we have interpreted in Section 3C5: in the C&P∧D&A architecture there are two control decisions **on** and **off** dedicated to express the intent of a supervisor's certain wish to disable or enable an event, whereas the third control decision **abstain** has no role in shaping the fused decision, as we have demonstrated that, when C&P∧D&A co-observability holds, never will all supervisors issue the decision **abstain** so that at least one of the supervisors will issue either **on** or **off**. Notice how the expression (3.9) corresponds with (3.11.1) and (3.11.2).

In Section 3C1 (resp., Section 3C2), we have seen that in the C&P (resp., D&A) architecture, only one absolute control decision is explicitly used: **off** (resp., **on**), which is traditionally written as **0** (resp., **1**). The other absolute control decision is in fact not lost, unlike what (3.3) (resp., (3.4)) seems to be suggesting. We see from (3.12) (resp., a similar expression we did not explicitly give), that both absolute control decisions are present, and that the C&P (resp., D&A) architecture can be obtained by adding a "conditionally enable" (resp., "conditionally disable") decision to the C&P∧D&A architecture. What happened in Section 3C1 (resp., Section 3C2), when we were interpreting the more compact expressions, is that the absolute control decision **on** (resp., **off**) is "lost", because its semantics coincide with that of both the "conditionally enable" (resp., "conditionally disable") decision and the "abstain" decision, in terms of how they shape the fused decision. Hence in the traditional notation, the decision **1** (resp., **0**) plays three distinct roles; the presentation in Section 3C1 (resp., Section 3C2) instead denotes this decision as **abstain** to emphasize its distinction from the other decision.

To understand more clearly why we can aggregate the three roles into one control decision, let us coalesce the aforementioned three roles into the conditional decision instead of into the **abstain** decision. Then, since whenever the event shall be enabled (resp., disabled), no supervisor would issue the absolute decision **off** (resp., **on**), therefore the conditional decision will not be overridden, hence there is no harm to use the conditional decision instead of the more explicit absolute decision. Also, expression (3.12) (resp., a similar expression we did not explicitly give) says never will all supervisors abstain. Then the "abstain" decision will always be overridden, by either the conditional decision or the absolute decision, hence the conditional decision can safely be conflated with the role to signify "abstaining".

To conclude briefly, in the C&P (resp., D&A) architecture there are semantically four distinct control decisions, where three of the decisions are represented jointly. Hence traditionally the control decisions are represented as binary values and the fusion rule is taken to be Boolean operations.

Lastly we see that the C&P∨D&A architecture is obtained by adding the other conditional decision to either the C&P architecture or the D&A architecture. Hence, there are semantically five distinct control decisions. Moreover, the expression (3.18) guarantees never will all supervisors abstain.

Looking back to the alternative interpretation in Section 3C6, there were only three control decisions. This is because whenever C&P∨D&A co-observability is satisfied, the two different kinds of conditional decisions will never be simultaneously issued, together with the fact that never will all supervisors abstain, these three decisions were aggregated into **abstain** in Section 3C6.

Therefore, we finally see what makes it possible to "default" the decision for an event when all supervisors abstain in Section 3C6. Very informally, if the language is C&P∨D&A co-observable, the supervisors knows so, and they also know to which default an event should be assigned, therefore, the supervisors can use a suitable conditional decision to express that knowledge, while reserving the **abstain** decision specifically to when they do not intend to shape the fused decision.

We noticed, with Defn. 3C6.2 and the interpretation of "default" decisions, a "default" has to be chosen a priori for each event; whereas with Defn. 3C6.5, we have a richer coding space: two conditional decisions. Then we realized that in the C&P∨D&A architecture for each event one has to choose only one of the two conditional decisions. Removing the restriction on the condition decisions leads to a generalized C&P∨D&A architecture, whose condition is given as

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,. \\ &(I, w) \models \bigvee_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \end{aligned} \tag{3.19.1}$$

$$\vee \bigvee_{i\in\mathcal{N}_\sigma} K_i(\neg\sigma_G \vee \sigma_E) \tag{3.19.2}$$

$$\vee \bigvee_{i\in\mathcal{N}_\sigma} K_i(\sigma_G \wedge \neg\sigma_E \Rightarrow \bigvee_{\substack{j\in\mathcal{N}_\sigma \\ j\neq i}} K_j(\neg\sigma_E)) \tag{3.19.3}$$

$$\vee \bigvee_{i\in\mathcal{N}_\sigma} K_i(\sigma_E \Rightarrow \bigvee_{\substack{j\in\mathcal{N}_\sigma \\ j\neq i}} K_j(\neg\sigma_G \vee \sigma_E)) \tag{3.19.4}$$

For the subsequent discussion, we refer to the architecture as generalized C&P∨ D&A architecture, and its condition generalized C&P∨D&A co-observability.

It is obvious that each of the four lines of the expression (3.19) corresponds to one distinct control decision, which we call **off**, **on**, **weak off**, and **weak on**, with the additional decision **abstain**[1]. We would like to emphasize that the correspondence among lines of the condition describing the local property of a plant's state facilitating a feasible control, the reasoning of the exact control decision choices of the supervisors, and the informal interpretation of the semantics of the control decisions. This correspondence is a major principle followed by the work of Ritsuka and Rudie [RR23].

This analysis also suggests that the requirement of not-all-abstaining is superfluous: it can always be achieved by adding higher levels of inferences [2] using the same technique we demonstrated here. Since admitting the not-all-abstaining requirement occasionally makes the discussion simpler, we opt for admitting it on those occasions.

3C7 Local Observability

Definition 3C7.1 (Observability [LW88])
The prefixed language $L(E)$ is observable with respect to P_i and $\Sigma_{i,c}$ whenever for any $s, s' \in L(E)$ and any $\sigma \in \Sigma_{i,c}$, we have

$$P_i(s) = P_i(s') \wedge s\sigma \in L(E) \wedge s'\sigma \in L(G) \Rightarrow s'\sigma \in L(E)$$

[1] For detailed discussion on the control decisions **weak off** and **weak on**, see the works by Yoo and Lafortune [YL04] and Ricker and Rudie [RR07] and Ritsuka and Rudie [RR23] and Ritsuka and Rudie [RR21]. Note that Yoo and Lafortune [YL04] and Ricker and Rudie [RR07] use different names for the control decisions.

[2] The notion of higher levels of inferences is discussed by Kumar and Takai [KT05]. A visualization and some supplements are presented by Ritsuka and Rudie [RR21], where the epistemic interpretation is also informally discussed.

Definition 3C7.2 (Local observability [TKU05])
The prefixed language $L(E)$ is locally observable whenever $L(E)$ is observable with respect to P_i and $\Sigma_{i,c}$ for all $i \in \mathcal{N}$.

Local observability is equivalent to

$$\begin{aligned} &\forall\, i \in \mathcal{N}\,.\, \forall\, \sigma \in \Sigma_{i,c}\,.\, \forall\, s, s' \in L(E)\,. \\ &\quad P_i(s) = P_i(s') \wedge s\sigma \in L(E) \wedge s'\sigma \in L(G) \\ &\quad \Rightarrow s'\sigma \in L(E) \end{aligned} \tag{3.20}$$

On the one hand, (3.20) is equivalent to

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\, \forall\, s \in L(E), \{\, s_i \in L(E)\,\}_{i \in \mathcal{N}_\sigma}\,. \\ &\quad [\forall\, i \in \mathcal{N}_\sigma\,.\, P_i(s) = P_i(s_i)] \\ &\quad \Rightarrow s\sigma \in L(E) \\ &\qquad \Rightarrow \bigwedge_{i \in \mathcal{N}_\sigma} s_i\sigma \in L(G) \\ &\qquad\quad \Rightarrow s_i\sigma \in L(E) \end{aligned}$$

which is equivalent to

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\, \forall\, w \in Q' \text{ such that } w_e \in Q^E\,. \\ &\quad (I, w) \models \sigma_G \wedge \sigma_E \\ &\qquad \Rightarrow \bigwedge_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \end{aligned}$$

This is only half the story. On the other hand, by exploiting the symmetry between s and s', with a different approach this time, the expression (3.20) is also equivalent to

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\, \forall\, s \in L(E), \{\, s_i \in L(E)\,\}_{i \in \mathcal{N}_\sigma}\,. \\ &\quad [\forall\, i \in \mathcal{N}_\sigma\,.\, P_i(s) = P_i(s_i)] \\ &\quad \Rightarrow s\sigma \in L(G) \wedge s\sigma \notin L(E) \\ &\qquad \Rightarrow \bigwedge_{i \in \mathcal{N}_\sigma} s_i\sigma \notin L(E) \end{aligned}$$

which is equivalent to

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\, \forall\, w \in Q' \text{ such that } w_e \in Q^E\,. \\ &\quad (I, w) \models \sigma_G \wedge \neg\, \sigma_E \\ &\qquad \Rightarrow \bigwedge_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \end{aligned}$$

Together, we have

$$\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,.$$
$$(I, w) \models \begin{bmatrix} \sigma_G \wedge \sigma_E \\ \Rightarrow \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \end{bmatrix} \wedge \begin{bmatrix} \sigma_G \wedge \neg\, \sigma_E \\ \Rightarrow \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \end{bmatrix}$$

which is equivalent to

$$\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,.$$
$$(I, w) \models \sigma_G \Rightarrow \begin{bmatrix} \sigma_E \Rightarrow \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \\ \wedge \neg\, \sigma_E \Rightarrow \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \end{bmatrix} \tag{3.21}$$

or more compactly,

$$\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,.$$
$$(I, w) \models \sigma_G \Rightarrow \begin{bmatrix} \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \\ \vee \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \end{bmatrix} \tag{3.22}$$

The epistemic expression suggests immediately that local observability is stronger than C&P∧D&A co-observability: the latter requires at least one supervisor capable making the correct decision, while the former requires every supervisor capable making the correct decision. Another way is to regard local observability as the condition for the architecture similar to the C&P∧D&A architecture except the control decision **abstain** is removed.

Similar to the case of C&P∧D&A co-observability, the $\sigma_G \Rightarrow \ldots$ is superfluous and can be removed.

3C8 Strong Local Observability

Definition 3C8.1 (Strong Observability [TKU05])
The prefixed language $L(E)$ is strongly observable with respect to P_i and $\Sigma_{i,c}$

whenever for any $s, s' \in L(G)$ and any $\sigma \in \Sigma_{i,c}$, we have

$$P_i(s) = P_i(s') \wedge s\sigma \in L(E) \wedge s'\sigma \in L(G) \Rightarrow s'\sigma \in L(E)$$

Definition 3C8.2 (Strong Local Observability [TKU05])
The prefixed language $L(E)$ is strongly locally observable whenever $L(E)$ is strongly observable with respect to P_i and $\Sigma_{i,c}$ for all $i \in \mathcal{N}$.

Following the same strategy of deriving an epistemic of local observability, strong local observability is equivalent to

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q'\,. \\ &(\overline{I}, w) \models \sigma_G \Rightarrow \begin{bmatrix} \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_E) \\ \vee \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_E) \end{bmatrix} \end{aligned} \tag{3.23}$$

Again, the difference between strong local observability (3.23) and local observability (3.22) is at the Kripke structures I and $\overline{I}$, or more specifically, the accessibility relations $\sim_i$ and $\simeq_i$. More is required for an agent to "know" something under the interpretation $\overline{I}$ than under the interpretation I. The argument is exactly the same as we have made in Section 3C3 Informally, strong local observability requires correct control decision even at illegal states. This is more than necessary since illegal states are not reachable if the language is locally observable and hence the desired control requirement can be met. This is exactly how much stronger strong local observability is than local observability.

A seeming generalization of strong local observability is

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q'\,. \\ &(\overline{I}, w) \models \sigma_G \Rightarrow \bigwedge_{i \in \mathcal{N}_\sigma} \begin{bmatrix} K_i(\neg\, \sigma_G \vee \sigma_E) \\ \vee K_i(\neg\, \sigma_E) \end{bmatrix} \end{aligned} \tag{3.24}$$

However, this is equivalent to strong local observability, since $(\overline{I}, w) \not\models K_i(\neg\, \sigma_G \vee \sigma_E) \wedge K_j(\neg\, \sigma_E)$.

3C9 Strong C&P∧D&A co-observability

We define a notion of strong C&P∧D&A co-observability, which is to C&P∧D&A co-observability as strong C&P co-observability is to C&P co-observability.

$$\forall\, \sigma \in \Sigma_c\,.\ \forall\, w \in Q'\,.$$
$$(\overline{I}, w) \models \sigma_G \Rightarrow \left[\begin{array}{l} \bigvee\limits_{i\in\mathcal{N}_\sigma} K_i(\neg\,\sigma_G \vee \sigma_E) \\ \vee \bigvee\limits_{i\in\mathcal{N}_\sigma} K_i(\neg\,\sigma_E) \end{array}\right] \tag{3.25}$$

From the epistemic expressions we can infer immediately, that strong C&P∧D&A co-observability is stronger than C&P∧D&A co-observability, as intended. Moreover, strong C&P∧D&A co-observability is weaker than strong local observability, just as C&P∧D&A co-observability is weaker than local observability. We therefore appreciate the epistemic expressions for providing such apparentness.

3C10 Weak Co-normality

Weak co-normality is defined in terms of the following modified projection function:

Definition 3C10.1 (Modified Projection Function [RW90])
Define the modified projection function $\tilde{P}_i : \Sigma^* \to \Sigma_{i,o}^*\Sigma \cup \{\varepsilon\}$ as

$$\tilde{P}_i(s) = \begin{cases} \varepsilon, & \text{if } s = \varepsilon \\ P_i(s')\sigma, & \text{if } s = s'\sigma \end{cases}$$

that is, $\tilde{P}_i$ behaves like P_i except it does not erase the last event of the string s, when s is not empty. ∎

Then weak co-normality can be defined as follows:

Definition 3C10.2 (Weak Co-normality [TKU05])
The prefixed language $L(E)$ is weakly co-normal whenever

$$\left[\bigcup_{i\in\mathcal{N}} \tilde{P}_i^{-1}\tilde{P}_i(L(E)) \cap L(G)\right] \subseteq L(E)$$

Note the inclusion in the other direction trivially holds. ∎

The language $L(E)$ is weakly co-normal iff

$$\forall\, s \in \Sigma^*\,.\ \Big[s \in L(G) \wedge \exists\, i \in \mathcal{N}\,.\ \Big[s \in \tilde{P}_i^{-1}\tilde{P}_i(L(E))\Big]\Big] \Rightarrow s \in L(E)$$

which is equivalent to

$$\forall\, s \in \Sigma^*\,.\; \exists\, i \in \mathcal{N}\,.\; \left[s \in \tilde{P}_i^{-1}\tilde{P}_i(L(E)) \wedge s \in L(G)\right] \Rightarrow s \in L(E)$$

Moving the existential quantifier out of the implications, we have equivalently

$$\forall\, s \in \Sigma^*\,.\; \forall\, i \in \mathcal{N}\,.\; s \in \tilde{P}_i^{-1}\tilde{P}_i(L(E)) \wedge s \in L(G) \Rightarrow s \in L(E)$$

which is equivalent to

$$\forall\, s \in \Sigma^*\,.\; \forall\, i \in \mathcal{N}\,.\; \left[\exists\, s' \in L(E)\,.\; \tilde{P}_i(s) = \tilde{P}_i(s')\right] \wedge s \in L(G) \Rightarrow s \in L(E)$$

Moving the existential quantifier out of the implications, we have equivalently

$$\forall\, s, s' \in \Sigma^*\,.\; \forall\, i \in \mathcal{N}\,.\; \tilde{P}_i(s) = \tilde{P}_i(s') \wedge s' \in L(E) \wedge s \in L(G) \Rightarrow s \in L(E)$$

Since for any i it is always the case that $\tilde{P}_i^{-1}\tilde{P}_i(\{\,\varepsilon\,\}) = \{\,\varepsilon\,\}$, hence we only need to consider non-empty strings. Therefore, the expression above is equivalent to

$$\begin{aligned}&\forall\, s, s'\Sigma^*\,.\; \forall\, \sigma \in \Sigma\,.\; \forall\, i \in \mathcal{N}\,.\\ &\quad \tilde{P}_i(s\sigma) = \tilde{P}_i(s'\sigma) \wedge s'\sigma \in L(E) \wedge s\sigma \in L(G) \Rightarrow s\sigma \in L(E)\end{aligned}$$

which is equivalent to

$$\begin{aligned}&\forall\, s, s' \in \Sigma^*\,.\; \forall\, \sigma \in \Sigma\,.\; \forall\, i \in \mathcal{N}\,.\\ &\quad P_i(s) = P_i(s') \wedge s'\sigma \in L(E) \wedge s\sigma \in L(G) \Rightarrow s\sigma \in L(E)\end{aligned} \tag{3.26}$$

In the same way the expression in Defn. 3C8.1 gives rise to the epistemic expression (3.23) of strong local observability, by noticing how the restrictions of the quantifications are relaxed in (3.26) from Defn. 3C8.1, the expression (3.26) can be written as

$$\begin{aligned}&\forall\, \sigma \in \Sigma\,.\; \forall\, w \in Q'\,.\\ &(\overline{I}, w) \models \sigma_G \Rightarrow \left[\begin{array}{l}\displaystyle\bigwedge_{i\in\mathcal{N}} K_i(\neg\,\sigma_G \vee \sigma_E)\\ \vee \displaystyle\bigwedge_{i\in\mathcal{N}} K_i(\neg\,\sigma_E)\end{array}\right]\end{aligned} \tag{3.27}$$

With this epistemic expression, we thus interpret weak co-normality as follows: for every event, including even the controllable ones, at any state, including even the illegal states (which the plant would never enter should correct control decisions be enforced), every supervisor must know whether the event can be disabled or enabled, even if that supervisor does not control that event.

3C11 Summary and Discussion

To better understand the relationships between the various decentralized DES conditions established, we first provide some shorthand for epistemic formulae. We will then be able to capture the relationships in a figure.

The following expressions are all implicitly parameterized by an event σ known from the context.

First, phrases regarding the desired decision of σ:

$e = \neg\, \sigma_G \vee \sigma_E$	σ can be enabled
$d = \neg\, \sigma_E$	σ can be disabled
$\underline{e} = \sigma_E$	σ must be enabled
$\underline{d} = \sigma_G \wedge \neg\, \sigma_E$	σ must be disabled

Then, define the modal operator "someone knows…":

$$S\phi = \vee_{i \in \mathcal{N}_\sigma} K_i \phi$$

Define the modal operator "everyone knows…":

$$E\phi = \wedge_{i \in \mathcal{N}_\sigma} K_i \phi$$

With an agent i known from the context, define a variant of the modal operator "someone knows" as "some other one (other than i) knows…":

$$O\phi = \vee_{\substack{j \in \mathcal{N}_\sigma \\ j \neq i}} K_j \phi$$

Hence, in a condition the presence of the phrases Se / Ee, Sd / Ed, $S(\underline{e} \Rightarrow Oe)$, $S(\underline{d} \Rightarrow Od)$ indicates the availabilities of the control decisions **on**, **off**, **weak off**, **weak on**, respectively.

With these shorthands, we summarize the epistemic expressions of the co-observability conditions and related variations in Fig. 3.2.

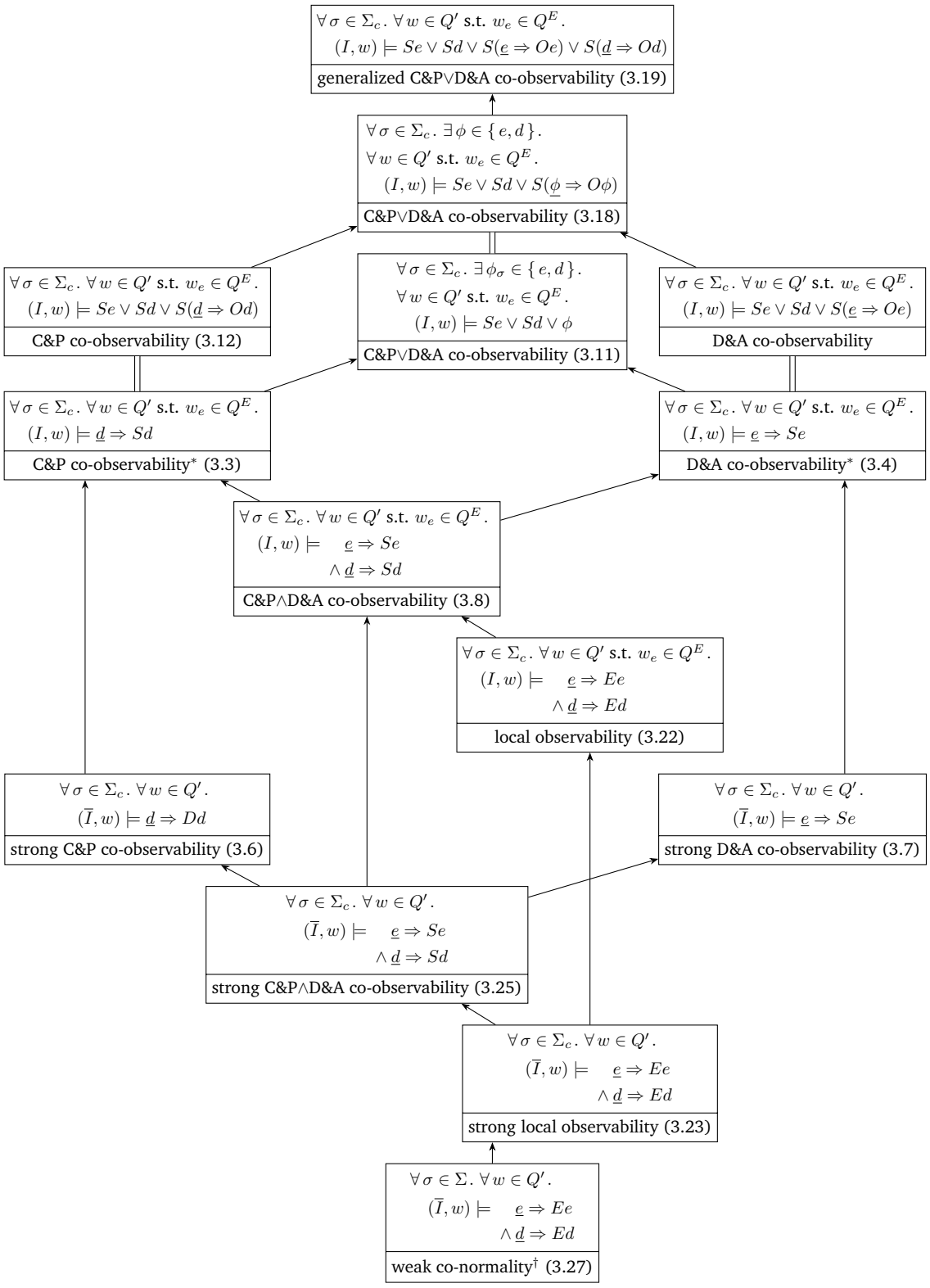

Figure 3.2: Lattice of co-observability conditions and related variations. Implications go in the direction of arrows.
Note: all expressions $\underline{e}\Rightarrow\phi\wedge\underline{d}\Rightarrow\psi$ can be contracted into $\phi\vee\psi$ (see Section 3C5).
*: the lack of one of $\underline{e}\Rightarrow Se$ and $\underline{d}\Rightarrow Sd$ entails the existence of a conditional decision.
†: the expression is written with $i\in\mathcal{N}$ instead of $i\in\mathcal{N}_\sigma$, which distinguishes weak co-normality from strong local observability.

The logical expressions in this chapter are independent of the finite automaton used to represent the plant, which enabled us to consider a finite, state-based Kripke structure, as in Ricker and Rudie [RR07]. However, if one wishes to apply a similar technique to other conditions (such as those with higher-order knowledge), then it would have to be first ascertained that the conditions are invariant to the plant representations. If they are not, then one could use a finer, (possibly infinite) string-based Kripke structure as described by Ricker and Rudie [RR00], along with corresponding definitions of relations $\simeq_i$ and $\sim_i$.

3D Discussion on Closure Under Set Union

Takai and Ushio [TU01] showed that strong local observability has the desirable property of being closed under set union. Through our earlier discussions in Sections 3C5 and 3C8 (see also Fig. 3.2), we realized that strong local observability is much stronger than what suffices for the decentralized problem to be solvable. Hence, we dedicate this section to the investigation of what makes strong local observability closed under union, and whether there are extraneous restrictions that can be removed to derive a weaker condition which is still closed under set union.

With the epistemic interpretation, we realized that strong local observability has two constraints in addition to C&P∧D&A co-observability. One is to require more for an agent to "know" something, through the use of the interpretation $\overline{I}$ instead of the interpretation I. The second is to require that not just at least one, but all supervisors are capable of making the correct decision. These two constraints are informally reflected by the words "strong" and "local" respectively. Therefore, it is interesting to see how removing either one of these two constraints breaks closure under set union. Namely, we will provide intuitive understanding of why strong C&P∧D&A co-observability (having only "strong") and local observability (having only "local") are not closed under set union.

3D1 Strong C&P∧D&A Co-observability is not Closed under Set Union

Strong C&P∧D&A co-observability is not closed under set union. To demonstrate this, we assume that the two languages $L(E_1)$ and $L(E_2)$ are strongly C&P∧D&A co-observable, and show that the language $L(E) = L(E_1) \cup L(E_2)$ is not necessarily strongly C&P∧D&A co-observable, by a satisfiability problem. That is, we will inspect the epistemic expression which expresses the strong C&P∧D&A co-observabilities

of $L(E_1)$ and $L(E_2)$ and the non- strong C&P∧D&A co-observability of $L(E)$, and deliberately construct a Kripke structure model of the expression.

Without loss of generality we assume that E_1, E_2 have been arranged to be subautomata of E, and that E_1, E_2 and E are subautomata of G. For our purpose, we extend the Kripke structure in the obvious way, so that, for example, $\pi(w, \sigma_{E_1}) = \textbf{true}$ whenever $\delta^{E_1}(w, \sigma)!$. Notice that $[(\overline{I}, w) \models \sigma_E] \Leftrightarrow [(\overline{I}, w) \models \sigma_{E_1}] \vee [(\overline{I}, w) \models \sigma_{E_2}]$.

Then by the strong C&P∧D&A co-observabilities of $L(E_1)$ and $L(E_2)$, we have

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\, \forall\, w \in Q'\,. \\ &(\overline{I}, w) \models \sigma_G \Rightarrow \bigwedge_{E_k \in \{\, E_1, E_2 \,\}} \left[\begin{array}{l} \bigvee\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_{E_k}) \\ \vee \bigvee\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_{E_k}) \end{array} \right] \end{aligned} \tag{3.28}$$

Consider an event $\sigma \in \Sigma_c$, a world $w \in Q'$, construct the model so that $(\overline{I}, w) \models \sigma_G$. Then it is possible to deliberately construct the model so that only the cases

$$\begin{aligned} (\overline{I}, w) &\models K_i(\neg\, \sigma_{E_1}) \\ (\overline{I}, w) &\models K_j(\neg\, \sigma_{E_2}) \end{aligned} \tag{3.29}$$

hold, and they hold only for two distinct supervisors $i, j \in \mathcal{N}_\sigma$.

For $L(E)$ to be not strongly C&P∧D&A co-observable, we need that

$$\begin{aligned} (\overline{I}, w) &\not\models K_i(\quad \neg\, (\sigma_{E_1} \vee \sigma_{E_2})) \\ (\overline{I}, w) &\not\models K_i(\sigma_G \Rightarrow (\sigma_{E_1} \vee \sigma_{E_2})) \end{aligned}$$

or equivalently, for some $w', w'' \in [w]_i$,

$$\begin{aligned} (\overline{I}, w') &\models \quad \sigma_{E_1} \vee \quad \sigma_{E_2} \\ (\overline{I}, w'') &\models \sigma_G \wedge \neg\, \sigma_{E_1} \wedge \neg\, \sigma_{E_2} \end{aligned}$$

Therefore, in the presence of (3.29), we need

$$\begin{aligned} (\overline{I}, w') &\models \quad \neg\, \sigma_{E_1} \wedge \quad \sigma_{E_2} \\ (\overline{I}, w'') &\models \sigma_G \wedge \neg\, \sigma_{E_1} \wedge \neg\, \sigma_{E_2} \end{aligned}$$

Since $\sigma_{E_2} \Rightarrow \sigma_G$, so we need

$$\begin{aligned} (\overline{I}, w') &\models \sigma_G \wedge \neg\, \sigma_{E_1} \wedge \sigma_{E_2} \\ (\overline{I}, w'') &\models \sigma_G \wedge \neg\, \sigma_{E_1} \wedge \neg\, \sigma_{E_2} \end{aligned}$$

That is, we confuse Supervisor i with event σ_{E_2} at State w_e, which is indeed possible. Similar argument applies for j as well. Therefore, as long as at State w_e, there is an event observable to one of i, j but not to the other, and vice versa, the requirement is satisfiable.

Using the satisfiability problem as guidance, we can explicitly construct the following example. Consider two supervisors with observed event sets $\Sigma_{1,o} = \{a\}$, $\Sigma_{2,o} = \{b\}$, and controlled event sets $\Sigma_{1,c} = \Sigma_{2,c} = \{c\}$. Let Fig. 3.3 depict the automaton $G' = G \times P_1(G) \times P_2(G)$. Since G' and G happen to be isomorphic in this example, we do not draw G separately. Let E_1 mark all states in G except States $1', 2'$ (i.e., all states whose left side is shaded), and E_2 mark all states in G except States $1', 3'$ (i.e., all states whose right side is shaded), so that E marks all states except State $1'$ (i.e., all states which have any shading).

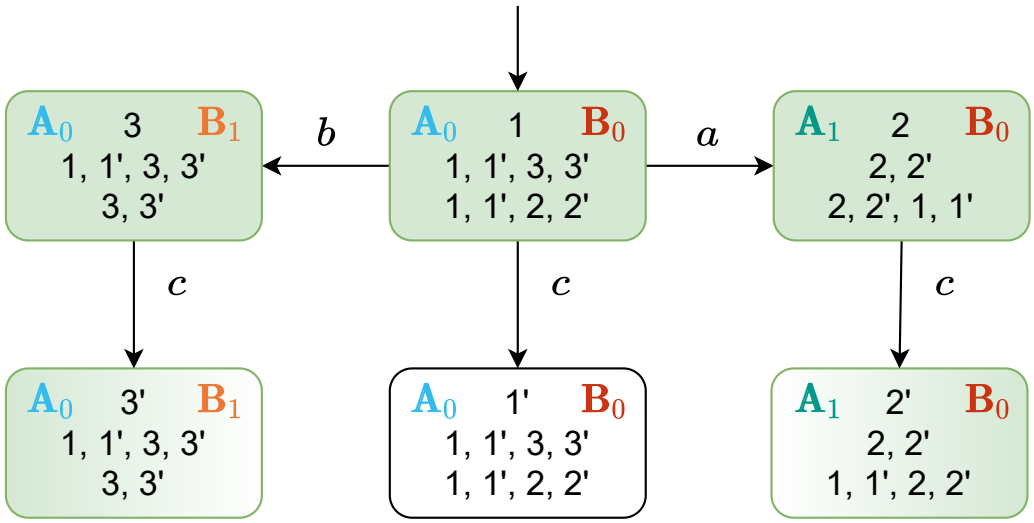

Figure 3.3: The automaton $G' = G \times P_1(G) \times P_2(G)$. A state $(q_G, q_1^{obs}, q_2^{obs})$ is represented in the figure with q, q_1^{obs}, q_2^{obs} stacked vertically in that order. States are also labelled by their equivalence classes $\ker \simeq_1$ (resp., $\ker \simeq_2$) in the upper left (resp., upper right) corner.

Since in this example no event may happen at illegal states, hence strong C&P∧D&A co-observability coincides with C&P∧D&A co-observability. We can verify that $L(E_1)$ and $L(E_2)$ are both (strongly) C&P∧D&A co-observable. However, (strong) C&P∧D&A co-observability of $L(E)$ is violated at State 1, since neither supervisor knows the correct control decision for the event c at State 1. This can be seen as follows. Supervisor 1 confuses sequences ε and b (i.e., States 1 and 3, respectively) and the former when followed by c is not in E but the latter when followed by c is, hence Supervisor 1 does not know whether c should be disabled at State 1. Similarly, Supervisor 2 confuses ε and a (i.e., States 1 and 2, respectively) and again c is illegal but ac is legal. As a result, Supervisor 2 also does not know whether c should be disabled at State 1. Therefore, (strong) C&P∧D&A co-observability is not closed

under union.

Intuitively, since (strong) C&P∧D&A co-observability does not require every supervisor to be certain but only some to be, we are able to confuse some supervisor with one language and all other remaining supervisors with the other language, so that all supervisors become confused in the union language. In the example above, at State 1, $L(E_1)$ is intended to confuse Supervisor 1 but not Supervisor 2, and $L(E_2)$ is intended to confuse Supervisor 2 but not Supervisor 1. Moreover, $L(E_1)$ and $L(E_2)$ are constructed so their union $L(E)$ does not resolve confusion for Supervisors 1 and 2 since after the union enough states are still illegal that both supervisors are confused at State 1.

3D2 Local Observability is not Closed under Set Union

We approach non-closure of local observability similarly as we did for strong C&P∧D&A co-observability in the previous section.

Since we are dealing with two languages, and I (more specifically, $\sim_i$) is language-dependent, we instead use the Kripke structure $\overline{I}$, and add the propositions w_{E_k}, so that $\pi(w, w_{E_k}) =$ **true** exactly when $w_e \in Q^{E_k}$. Therefore, for propositional formula ϕ involving only σ_G and σ_E, for all $w \in Q'$,

$$w_e \in Q^{E_k} \Rightarrow (I, w) \models K_i(\phi)$$

exactly when

$$(\overline{I}, w) \models w_{E_k} \Rightarrow K_i(w_{E_k} \Rightarrow \phi)$$

Assuming $L(E_1)$ and $L(E_2)$ are locally observable, we have

$$\forall\, \sigma \in \Sigma_c .\ \forall\, w \in Q' .$$
$$\bigwedge_{E_k \in \{\, E_1, E_2 \,\}} (\overline{I}, w) \models w_{E_k} \Rightarrow \sigma_G \Rightarrow \left[\begin{array}{l} \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(w_{E_k} \Rightarrow \neg\, \sigma_G \vee \sigma_{E_k}) \\ \vee \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(w_{E_k} \Rightarrow \neg\, \sigma_{E_k}) \end{array} \right]$$

Since $w_e \in Q^E$ exactly when $w_e \in Q^{E_1}$ or $w_e \in Q^{E_2}$, we have $(\overline{I}, w) \models w_E$ exactly

when $(\overline{I}, w) \models w_{E_1}$ or $(\overline{I}, w) \models w_{E_2}$. Hence we have

$$\forall\, \sigma \in \Sigma_c.\ \forall\, w \in Q' \text{ such that } w_e \in Q^E\,.$$
$$(\overline{I}, w) \models w_E \Rightarrow \sigma_G \Rightarrow \bigvee_{E_k \in \{\, E_1, E_2 \,\}} \left[\begin{array}{l} \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(w_{E_k} \Rightarrow \neg\, \sigma_G \vee \sigma_{E_k}) \\ \vee \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(w_{E_k} \Rightarrow \neg\, \sigma_{E_k}) \end{array} \right] \tag{3.30}$$

Similar to our approach in the previous section, we deliberately construct the model so that $(\overline{I}, w) \models w_E \wedge \sigma_G$ and the only disjunction that holds is

$$\bigwedge_{i \in \mathcal{N}_\sigma} K_i(w_{E_1} \Rightarrow \neg\, \sigma_{E_1})$$

for some $i \in \mathcal{N}_\sigma$. That is, we ensure that $w_e \notin Q^{E_2}$.

Since we desire to show that $L(E)$ is not locally observable, we would like to construct, for some $i \in \mathcal{N}_\sigma$, some $w' \in [w]_i$, so that

$$\begin{aligned} (\overline{I}, w) &\models \sigma_G \wedge w_{E_1} \wedge \neg w_{E_2} \wedge \neg\, \sigma_{E_1} \wedge \neg \sigma_{E_2} \\ (\overline{I}, w') &\models \sigma_G \wedge w_{E_1} \wedge \quad w_{E_2} \wedge \neg\, \sigma_{E_1} \wedge \quad \sigma_{E_2} \end{aligned}$$

which requires, for that specific i, for all $w'' \in [w']_i = [w]_i$, that

$$(\overline{I}, w'') \models w_{E_2} \Rightarrow \sigma_G \Rightarrow \sigma_{E_2}$$

Therefore, to make it easier for us, we ensure that w' is the only world in $[w']_i = [w]_i$ such that $(\overline{I}, w') \models w_{E_2} \wedge \sigma_G$.

Since $(\overline{I}, w') \models \sigma_G \wedge w_{E_2} \wedge \sigma_{E_2}$, it must be that for all $w'' \in [w']_i = [w]_i$, $(\overline{I}, w'') \models w_{E_2} \Rightarrow \neg\, \sigma_G \vee \sigma_{E_2}$. Specifically, since $w \in [w]_i$, $(\overline{I}, w) \models w_{E_2} \Rightarrow \neg\, \sigma_G \vee \sigma_{E_2}$. Again, for convenience, we ensure that at no other world in $[w]_i$, we have $w_{E_2} \Rightarrow \neg\, \sigma_G \vee \sigma_{E_2}$.

To summarize, since we are aiming for a compact example, and since the epistemic expression permits us to do so, the example is deliberately constructed so that there are only two states in $[w]_i$, for only one particular i.

Using the satisfiability problem as guidance, we can explicitly construct the following example. Consider two supervisors with observed event sets $\Sigma_{1,o} = \varnothing$, $\Sigma_{2,o} = \{\, a \,\}$, and controlled event sets $\Sigma_{1,c} = \Sigma_{2,c} = \{\, a, c \,\}$. Let Fig. 3.4 depict the automaton $G' = G \times P_1(G) \times P_2(G)$. Since G' and G happen to be isomorphic in this example, we do not draw G separately. Let E_1 mark States $1, 2$ (i.e., all states whose left side is shaded), and E_2 mark States $1, 1'$ (i.e., all states whose right side is shaded), so that E marks all states except State $2'$ (i.e., all states which is somehow shaded).

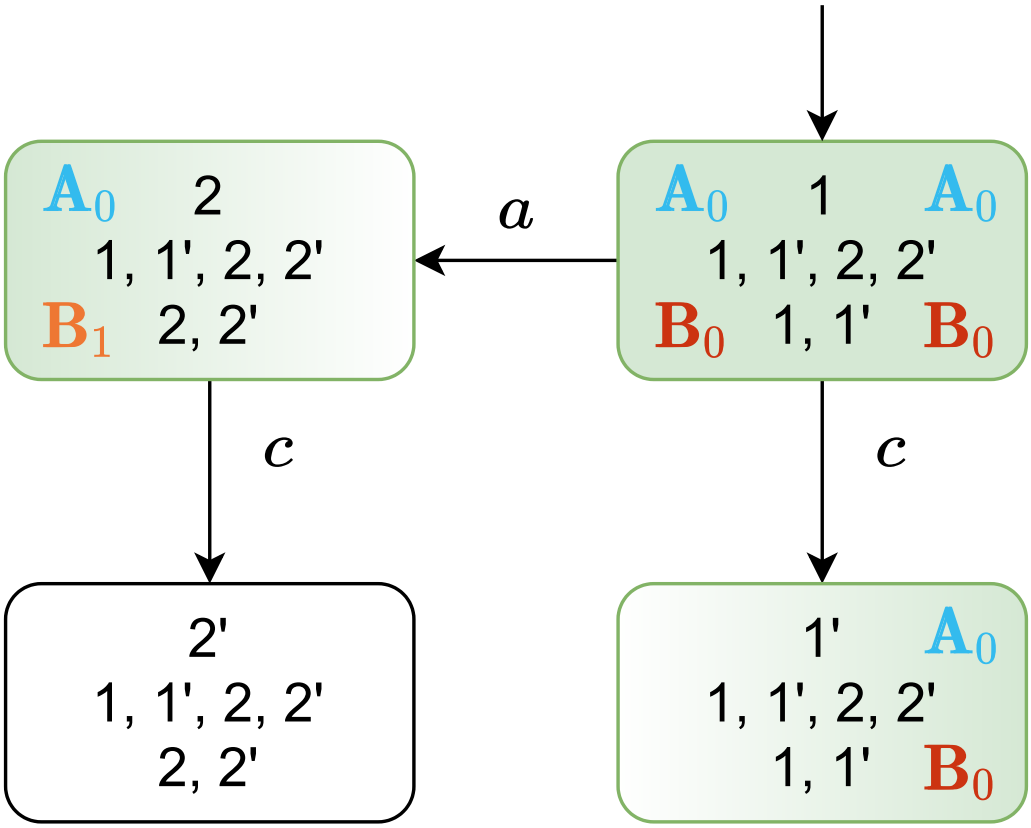

Figure 3.4: The automaton $G' = G \times P_1(G) \times P_2(G)$. A state $(q_G, q_1^{obs}, q_2^{obs})$ is represented in the figure with q, q_1^{obs}, q_2^{obs} stacked vertically in that order. Note that we label states by their equivalence classes, $\ker \sim_{1,E_1}$, $\ker \simeq_{1,E_2}$ $\ker \sim_{2,E_1}$, $\ker \simeq_{2,E_2}$ in the upper left, upper right, lower left, lower right corners.

We can verify that $L(E_1)$ and $L(E_2)$ are both locally observable. However, local observability of $L(E)$ is violated at State 2, since Supervisor 1 does not know the correct control decision for the event c at States 1 and 2. This can be seen as follows. Supervisor 1 confuses sequences ε and a (i.e., States 1 and 2, respectively) and the former when followed by c is not in E but the latter when followed by c is, hence Supervisor 1 does not know whether c should be disabled at State 1. Therefore, local observability is not closed under union.

Intuitively, restricting the quantification of $w \in Q'$ to exclude worlds such that w_e is not in, say, Q^{E_2}, makes the supervisors vulnerable at states that are otherwise reachable in the union language (since if $w_e \in Q^{E_1}$, then $w_e \in Q^E$). Notice how, in the example above, State 2 is not in E_2, yet created confusion when brought in to E by E_1.

3D3 Strong Local Observability is Closed under Set Union

Although it has already been proven that strong local observability is closed under set union [TU02], here we are interested in seeing how we were able to construct the counterexamples in the previous two sections, and how strong local observability would prevent us doing so.

Suppose that $L(E_1)$ and $L(E_2)$ are strongly locally observable, and we derive

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c\,.\, \forall\, w \in Q'\,. \\ &(I, w) \models \sigma_G \Rightarrow \bigwedge_{E_k \in \{\, E_1, E_2 \,\}} \left[\begin{array}{l} \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_G \vee \sigma_{E_k}) \\ \vee \bigwedge\limits_{i \in \mathcal{N}_\sigma} K_i(\neg\, \sigma_{E_k}) \end{array} \right] \end{aligned} \tag{3.31}$$

Compare (3.31) with (3.28) and (3.30). What allowed us to construct the counterexamples are specific disjunctions in (3.28) and (3.30), which are changed to conjunctions in (3.31).

3D4 Revisiting Strong C&P∧D&A Co-observability

In Section 3D1 we demonstrated that strong C&P∧D&A co-observability is not closed under union. That is, given two strongly C&P∧D&A co-observable languages $L(E_1)$ and $L(E_2)$, their union $L(E) = L(E_1) \cup L(E_2)$ is not necessarily synthesisable under the C&P∧D&A architecture.

However, Section 3D1 does not claim that the language $L(E)$ is not synthesisable at all. In fact, since any strongly C&P∧D&A co-observable language is also strongly D&A co-observable, and strong D&A co-observability is closed under union, it follows that given two strongly C&P∧D&A co-observable languages $L(E_1)$ and $L(E_2)$, their union $L(E) = L(E_1) \cup L(E_2)$ is synthesisable under the D&A architecture[3].

Then it is interesting to ask, given that D&A architecture suffices, whether it is also necessary; and moreover, what D&A architecture provides to facilitate the synthesis of the union of two strongly C&P∧D&A languages.

Assuming strong C&P∧D&A co-observability of two languages $L(E_1)$ and $L(E_2)$, we perform a case analysis for expression (3.28).

Case 1: if at w we have $K_i(\neg\sigma_G \vee \sigma_{E_1})$ for some $i \in \mathcal{N}_\sigma$, then we can derive

$$K_i(\neg\sigma_G \vee (\sigma_{E_1} \vee \sigma_{E_2}))$$

and thus

$$K_i(\neg\sigma_G \vee \sigma_E)$$

Case 2: if at w we have $K_i(\neg\sigma_{E_1})$ for some $i \in \mathcal{N}_\sigma$,

Now consider two separate cases.

Case 2.1: if at all $w' \in [w]_i$, whenever σ_G holds we have either $K_i(\neg\sigma_G \vee \sigma_{E_2})$ or $K_i(\neg\sigma_{E_2})$, then we have two cases:

Case 2.1.1: if at some $w' \in [w]_i$ such that σ_G holds, we have $K_i(\neg\sigma_G \vee \sigma_{E_2})$, then we also have $K_i(\neg\sigma_G \vee (\sigma_{E_1} \vee \sigma_{E_2}))$, that is, $K_i(\neg\sigma_G \vee \sigma_E)$, which must hold at w as well.

Case 2.1.2: if at some $w' \in [w]_i$ such that σ_G holds, we have $K_i(\neg\sigma_{E_2})$, then since $K_i(\neg\sigma_{E_1})$ holds at w' as well, we have $K_i(\neg\sigma_{E_1} \wedge \neg\sigma_{E_2})$, that is, $K_i(\neg(\sigma_{E_1} \vee \sigma_{E_2}))$, and hence $K_i(\neg\sigma_E)$, which must hold at w as well.

Case 2.2: if at none of the $w' \in [w]_i$ such that σ_G holds, we have either $K_i(\neg\sigma_G \vee \sigma_{E_2})$ or $K_i(\neg\sigma_{E_2})$, then by (3.28), we have either $K_j(\neg\sigma_G \vee \sigma_{E_2})$ or $K_j(\neg\sigma_{E_2})$ for some $j \in \mathcal{N}_\sigma$, where j must be different from i. Then consider an arbitrary $w' \in [w]_i$ such

[3] This result gives us an inspiration: given two languages $L(E_1)$ and $L(E_2)$ synthesisable in the architecture A, we should not confine ourselves in synthesizing the union language $L(E)$ in the architecture A, but instead be willing to look for an alternative architecture B in which we can synthesize $L(E)$. If we let all architectures be ordered by their strength, we'd then like to ask: does there exist a supremal architecture?

that σ_G holds (for otherwise the result follows vacuously). We have two cases to consider here.

Case 2.2.1: if at w' we have $K_j(\neg\sigma_G \vee \sigma_{E_2})$, then with a similar argument as case 1, we have $K_j(\neg\sigma_G \vee \sigma_E)$, and hence automatically $\sigma_E \Rightarrow K_j(\neg\sigma_G \vee \sigma_E)$.

Case 2.2.2: if at w' we have $K_j(\neg\sigma_{E_2})$, then we have $\neg\sigma_{E_1} \wedge \neg\sigma_{E_2}$. That is, $\neg(\sigma_{E_1} \vee \sigma_{E_2})$, hence, $\neg\sigma_E$. Thus vacuously $\sigma_E \Rightarrow K_j(\neg\sigma_G \vee \sigma_E)$.

Together, for case 2.2, at any $w' \in [w]_i$ we have $\sigma_E \Rightarrow K_j(\neg\sigma_G \vee \sigma_E)$. Thus at w we have $K_i(\sigma_E \Rightarrow K_j(\neg\sigma_G \vee \sigma_E))$.

Together, we have

$$\forall\sigma \in \Sigma_c.\ \forall w \in Q'.$$
$$(\overline{I}, w) \models \sigma_G \Rightarrow \left[\begin{array}{l} \bigvee\limits_{i\in\mathcal{N}_\sigma} K_i(\neg\sigma_G \vee \sigma_E) \\ \vee \bigvee\limits_{i\in\mathcal{N}_\sigma} K_i(\neg\sigma_E) \\ \vee \bigvee\limits_{i\in\mathcal{N}_\sigma} K_i(\sigma_E \Rightarrow \bigvee\limits_{\substack{j\in\mathcal{N}_\sigma \\ j\neq i}} K_j(\neg\sigma_G \vee \sigma_E)) \end{array}\right]$$

Following an approach similar to how we proved Thm. 3C6.3, we see the expression above is equivalent to strong D&A co-observability. Hence, given two languages $L(E_1)$ and $L(E_2)$, if they are strongly C&P∧D&A co-observable, then their union $L(E)$ must be strongly D&A co-observable. That is, the D&A architecture is indispensable to the synthesis of the union of two strongly C&P∧D&A co-observable languages.

3E Conclusion

This chapter presents epistemic characterizations of co-observability conditions. Such characterizations provide more intuitive understanding of these conditions. Closures under set union of some of the conditions are also discussed, where we provide a systematic approach to find counterexamples.

References

[Cie+88] R. Cieslak, C. Desclaux, A. S. Fawaz, and P. Varaiya. "Supervisory control of discrete-event processes with partial observations". In: *IEEE Transactions on Automatic Control* 33.3 (Mar. 1988), pp. 249–260. DOI: 10.1109/9.402. [cit. on p. 18].

[KT05] R. Kumar and S. Takai. "Inference-based Ambiguity Management in Decentralized Decision-Making: Decentralized Control of Discrete Event Systems". In: *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005. DOI: 10.1109/CDC.2005.1582701. [cit. on p. 39].

[LW88] F. Lin and W. M. Wonham. "On observability of discrete-event systems". In: *Information Sciences* 44.3 (Apr. 1988), pp. 173–198. DOI: 10.1016/0020-0255(88)90001-1. [cit. on p. 39].

[PKK97] J. H. Prosser, M. Kam, and H. G. Kwatny. "Decision fusion and supervisor synthesis in decentralized discrete-event systems". In: *Proceedings of the American Control Conference*. IEEE, 1997. DOI: 10.1109/ACC.1997.608978. [cit. on p. 17. 18. 19. 25. 37].

[RR00] S. L. Ricker and K. Rudie. "Know means no: Incorporating knowledge into discrete-event control systems". In: *IEEE Transactions on Automatic Control* 45.9 (2000), pp. 1656–1668. DOI: 10.1109/9.880616. [cit. on p. 23. 24. 47].

[RR07] S. L. Ricker and K. Rudie. "Knowledge Is a Terrible Thing to Waste: Using Inference in Discrete-Event Control Problems". In: *IEEE Transactions on Automatic Control* 52.3 (Mar. 2007), pp. 428–441. DOI: 10.1109/TAC.2007.892371. [cit. on p. 24. 39. 47].

[RR21] K. Ritsuka and Karen Rudie. "A Visualization of Inference-Based Supervisory Control in Discrete-Event Systems". In: *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, Dec. 2021. DOI: 10.1109/cdc45484.2021.9683210. [cit. on p. 39].

[RR23] K. Ritsuka and K. Rudie. *Do What You Know: Coupling Knowledge with Action in Discrete-Event Systems*. Submitted for publication. 2023. [cit. on p. 17. 23. 26. 31. 32. 33. 34. 39].

[RW87] P. J. Ramadge and W. M. Wonham. "Supervisory Control of a Class of Discrete Event Processes". In: *SIAM Journal on Control and Optimization* 25.1 (Jan. 1987), pp. 206–230. DOI: 10.1137/0325013. [cit. on p. 23].

[RW90] Karen Rudie and W. Murray Wonham. "The infimal prefix-closed and observable superlanguange of a given language". In: *Systems & Control Letters* 15.5 (Dec. 1990), pp. 361–371. DOI: 10.1016/0167-6911(90)90059-4. [cit. on p. 37. 43].

[RW92] K. Rudie and W. M. Wonham. “Think globally, act locally: decentralized supervisory control”. In: *IEEE Transactions on Automatic Control* 37.11 (1992), pp. 1692–1708. DOI: 10.1109/9.173140. [cit. on p. 17. 18. 19. 20. 37].

[TKU05] S. Takai, R. Kumar, and T. Ushio. “Characterization of co-observable languages and formulas for their super/sublanguages”. In: *IEEE Transactions on Automatic Control* 50.4 (Apr. 2005), pp. 434–447. DOI: 10.1109/tac.2005.844724. [cit. on p. 17. 18. 19. 27. 29. 40. 41. 42. 43].

[TU01] S. Takai and T. Ushio. “Strong co-observability conditions for decentralized supervisory control of discrete event systems”. In: *Proceedings of the 40th IEEE Conference on Decision and Control*. IEEE, 2001. DOI: 10.1109/cdc.2001.980821. [cit. on p. 26. 29. 47].

[TU02] Shigemasa Takai and Toshimitsu Ushio. “A modified normality condition for decentralized supervisory control of discrete event systems”. In: *Automatica* 38.1 (Jan. 2002), pp. 185–189. DOI: 10.1016/s0005-1098(01)00187-x. [cit. on p. 53].

[YL02] T.-S. Yoo and Stéphane Lafortune. “A General Architecture for Decentralized Supervisory Control of Discrete-Event Systems”. In: *Discrete Event Dynamic Systems* 12.3 (2002), pp. 335–377. DOI: 10.1023/a:1015625600613. [cit. on p. 18. 19. 31].

[YL04] T.-S. Yoo and S. Lafortune. “Decentralized Supervisory Control With Conditional Decisions: Supervisor Existence”. In: *IEEE Transactions on Automatic Control* 49.11 (Nov. 2004), pp. 1886–1904. DOI: 10.1109/tac.2004.837595. [cit. on p. 37. 39].

4 Do What You Know: Coupling Knowledge with Action in Discrete-Event Systems

An epistemic model for decentralized discrete-event systems with non-binary control is presented. This framework combines existing work on inference-based control decisions with existing work on formal reasoning about knowledge in discrete-event systems. The novelty in the epistemic formalism is in providing an approach to derive problem solvability conditions and problem solutions. The derived expressions directly encapsulate the actions that supervisors must take. This direct coupling between knowledge and action — in a formalism that mimics natural language — makes it easier, when the solvability condition fails, to understand why the condition fails and may aid in determining how the problem requirements could be revised.

4A Introduction

The emergence of networked systems, including smart vehicles, home automation, and wearables has increased the need for decentralized supervisory control: the concept that the control is performed by not a monolithic, but many individual entities — or agents — separated by the environment. This chapter focuses on systems modelled as discrete-event systems (DES).

With control actions performed jointly, a mechanism — called a *fusion rule* — is needed to combine control decisions of the agents. Decentralized control of discrete-event systems under partial observations began with allowing only Boolean control decisions, and synthesis of the control policy has been studied when the fusion rule is conjunctive [Cie+88; RW92], and later other fusion rules are considered [PKK97; YL02], during which time the fusion rules can be interpreted as simply an arbiter to resolve conflicting control decisions. Further work by Yoo and Lafortune [YL04] extended the approach and proposed a conditional architecture to allow non-binary control decisions with a more sophisticated fusion rule, so that supervisors can "conditionally" turn on/off events based on the actions of other supervisors. Yoo and Lafortune gave necessary and sufficient conditions for the existence of supervisors [YL04] and a realization of the supervisors [YL05] in the conditional architecture.

In existing DES research, the conditions for solvability and supervisor synthesis

(when those conditions are satisfied) typically each rely on constructions that are divorced from each other, and a similar remark applies also to their respective proofs of correctness. The constructions appear to be creations *ex nihilo*, and thus do not provide insight into how they came to be. Moreover, the formal approach used is almost always the linguistic approach — where one reasons about strings in the relevant language representing the DES. Verifying that the solvability conditions are correct or that the corresponding supervisors solve the problem requires the reader to come up with their own informal understanding and interpretation of the conditions/constructions.

With a different formalism, Ricker and Rudie [RR07] gave an epistemic interpretation to the conditional architecture, where the use of the formal language of epistemic logic enabled one to discuss the supervisory control in an anthropomorphic manner, which gives a more intuitive understanding for how control decisions are made. Their epistemic modelling resolves the drawback of the aforementioned linguistic approach, namely the meaning of an epistemic expression is immediately understandable at a glance, so that an expression of the form $K_1\phi$ means "Supervisor 1 knows ϕ". The interpretation is only partial as their epistemic expression only captures a weaker architecture. Moreover, in their epistemic logic formulation, there is a tenuous connection between the solvability conditions and the actions to be prescribed for supervisors in a construction that exploits the conditions.

In our earlier work [RR22c], we adapted Ricker and Rudie's epistemic formalism to the interpretation of some other commonly known architectures [Cie+88; RW92; PKK97; YL02]. The result was fruitful, in giving concise epistemic characterizations to the various architectures in a way such that each characterization is constituted by a disjunction of epistemic terms, and each term corresponds to a specific control decision. As such, the characterizations differ by only the presence or absence of terms in the disjunction, corresponding to the presence or absence of control decisions available in an architecture. This was achieved by reformulating an architecture as a result of observing that some control decisions plays multiple distinct roles.

This chapter continues our advocacy of applying epistemic formalism and interpretation as an umbrella framework to the study of decentralized problems. In the present work, we cast a standard and representative but more complex conditional architecture in epistemic logic as well. But unlike our earlier work and any other prior works, which simply *present* supervisor existence and realization and *then* establish their correctness, the novelty of this work is the direct derivation of existence and realization expressions methodologically *from* the fusion rule. Notably, the derivation results in a direct link between the condition that must hold for a solution to exist and the control protocol that must be followed when the condition holds. That is, the result has a line-by-line correspondence between the expressions of the *knowledge* the supervisors must possess and the *actions* they must take.

We have chosen to demonstrate an epistemic characterization only of the conditional architecture instead of over the more general inference-based architectures [KT07]. This choice is made as the demonstration presented here will be sufficiently instructive for how the methodology can be routinely applied in extending the result over the inference-based architectures. Hence here we put emphasis on the process over the result.

4B Direct Derivation of Supervisor Existence and Realization for Conditional Architecture

The conditional architecture of Yoo and Lafortune [YL04] admits five possible local decisions: "enable", "disable", "enable if nobody disables", "disable if nobody enables", and "no decision". As argued in [RR22c], to remove the potential confusion of a local "enable" (resp., "disable") and a global "enable" (resp., "disable") decisions, we renamed the former to **on** (resp., **off**). We also give more compact names to the conditional decisions "enable if nobody disables" and "disable if nobody enables" and called them **weak on** and **weak off**, respectively. Finally, we consider "no decision" as a decision and hence call it **abstain**.

We now present the conditional architecture from Yoo and Lafortune [YL04] as follows. The set of control decisions is $\mathcal{CD} = \{$ **on**, **off**, **weak on**, **weak off**, **abstain** $\}$. For each $\sigma \in \Sigma_c$, a default action $\mathtt{dft} \in \{$ **enable**, **disable** $\}$ must be chosen as part of the solution. By letting the collection of local decisions for σ after string s be $cd = \{ f_i(P_i(s), \sigma) \}_{i \in \mathcal{N}_\sigma}$ for short, the fusion rule $f_\sigma^{\mathtt{dft}}$ for σ is defined as

$$
f_\sigma^{\mathtt{dft}}(cd) = \begin{cases}
\textbf{enable} & \text{if} \quad \textbf{on} \in cd, \quad \textbf{off} \notin cd & (4.1.1)\\
\textbf{disable} & \text{if} \quad \textbf{on} \notin cd, \quad \textbf{off} \in cd & (4.1.2)\\
\textbf{enable} & \text{if} \quad \textbf{on} \notin cd, \quad \textbf{off} \notin cd, & (4.1.3)\\
 & \qquad \textbf{weak on} \in cd, \ \textbf{weak off} \notin cd & \\
\textbf{disable} & \text{if} \quad \textbf{on} \notin cd, \quad \textbf{off} \notin cd, & (4.1.4)\\
 & \qquad \textbf{weak on} \notin cd, \ \textbf{weak off} \in cd & \\
\mathtt{dft} & \text{if} \quad \textbf{on} \notin cd, \quad \textbf{off} \notin cd, & (4.1.5)\\
 & \qquad \textbf{weak on} \notin cd, \ \textbf{weak off} \notin cd &
\end{cases}
$$

Hence, a solution to the DSCOP, in addition to constructing the supervisors, also requires that for each $\sigma \in \Sigma_c$ one chooses either the fusion rules $f^{\textbf{enable}}$ or the fusion rule $f^{\textbf{disable}}$.

Ideally, we would like to gradually *derive* the expressions of the problem solvability condition and derive construction of the supervisors directly from the fusion rule. However, as the written form of communication prevents us from doing so, we have to give both of them *a priori*. Nonetheless, the development process will still be apparent from the proof. We stress that the expressions are not creations *ex nihilo*, but are obtained *from* the fusion rule. This process is quite methodologically, as in every step there is only one sensible choice. Hence our methodology contrasts with the traditional approaches, which generally involve some human cleverness.

For ease of understanding and compactness, we define the following shorthand notation for epistemic formulae.

The following shorthand are defined in terms of σ_G and σ_E. We use over-bar instead of the standard symbol for logical negative in expressions when it makes it clearer to see at a glance which propositions in a compound proposition are or are not negated.

1. $\sigma_{e!} \coloneqq \sigma_E = \sigma_G \wedge \sigma_E$, reads "$\sigma$ must be enabled to satisfy the control requirement";

2. $\sigma_{d!} \coloneqq \sigma_G \wedge \overline{\sigma_E}$, reads "$\sigma$ must be disabled to satisfy the control requirement";

3. $\sigma_e \coloneqq \overline{\sigma_G} \vee \sigma_E = \sigma_G \Rightarrow \sigma_E$, reads "$\sigma$ can be enabled without violating the control requirement" or alternatively, "if σ is even possible then it ought to be enabled; otherwise, it does not matter";

4. $\sigma_d \coloneqq \overline{\sigma_E}$, reads "$\sigma$ can be disabled without violating the control requirement", or alternatively, "if σ is even possible then it ought to be disabled; otherwise, it does not matter".

Note that σ_d could have been defined as $\overline{\sigma_G} \vee \overline{\sigma_E}$ to parallel our definition of σ_e, but since an event that is not legal is also not possible, $\overline{\sigma_E}$ implies $\overline{\sigma_G}$ already.

The following expressions are all implicitly parameterized by an event σ known from the context.

Define the modal operator "someone knows…":

$$S\phi \coloneqq \bigvee_{i \in \mathcal{N}_\sigma} K_i \phi$$

With a supervisor i known from the context, define a variant of the modal operator

"someone knows" as "some other supervisor (other than i) knows…":

$$O\phi \coloneqq \bigvee_{\substack{j\in\mathcal{N}_\sigma \\ j\neq i}} K_i\phi$$

Finally, we need the following shorthand for some frequently needed epistemic expressions.

$$\begin{aligned} K_i^0\sigma_e &\coloneqq K_i\sigma_e \\ K_i^0\sigma_d &\coloneqq K_i\sigma_d \\ K_i^1\sigma_e &\coloneqq K_i(\sigma_{d!} \Rightarrow O\sigma_d) \\ K_i^1\sigma_d &\coloneqq K_i(\sigma_{e!} \Rightarrow O\sigma_e) \end{aligned}$$

Although we plan to *derive* the solvability condition and a control policy, it is nonetheless beneficial to first consider a tentative, but tangible proposal. Consider a tentative knowledge-based control policy $(G_i^{obs}, \mathcal{KP}_i)$, where $\mathcal{KP}_i$ is defined according to (4.2):

$$\mathcal{KP}_i(w,\sigma) = \begin{cases} \textbf{on} & \text{if } (I,w) \models K_i^0\sigma_e \wedge \overline{K_i^0\sigma_d} & (4.2.1) \\ \textbf{off} & \text{if } (I,w) \models \overline{K_i^0\sigma_e} \wedge K_i^0\sigma_d & (4.2.2) \\ \textbf{weak on} & \text{if } (I,w) \models \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge K_i^1\sigma_e \wedge \overline{K_i^1\sigma_d} & (4.2.3) \\ \textbf{weak off} & \text{if } (I,w) \models \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge \overline{K_i^1\sigma_e} \wedge K_i^1\sigma_d & (4.2.4) \\ \textbf{abstain} & \text{otherwise} & (4.2.5) \end{cases}$$

This construction appears to be quite natural. Even without formally deriving it from the fusion rule (4.1), one may still be able to instinctively come up with it, as the epistemic expressions directly capture what decisions are desirable. The point can be made stronger, if one temporarily ignores the semantics of the epistemic formulae and focus on how the form of (4.2) parallels with that of the fusion rule (4.1).

Clearly, the cases defining (4.2) are mutually exclusive and exhaustive. Intuitively, one can see that the correctness of cases (4.2.1) to (4.2.4) is guaranteed by the epistemic formulae, but since the case (4.2.5) does not involve any epistemic expressions, a condition to ensure its correctness is needed. This condition is our conditional-co-observability.

Definition 4B.1
The Kripke structure I is said to be conditional-co-observable whenever for each $\sigma \in \Sigma_c$, there is a choice of $*$ from e and d for this σ, such that for any string

$s \in L(E)$, where w is the world s leads to, it must be that

$$(I, w) \models \left[\bigwedge_{i \in \mathcal{N}_\sigma} \overline{K_i^0 \sigma_d} \wedge \overline{K_i^0 \sigma_e} \wedge \overline{K_i^1 \sigma_d} \wedge \overline{K_i^1 \sigma_e} \right] \Rightarrow \sigma_*,$$

i.e.,

$$\begin{aligned} (I, w) \models \quad & S^0 \sigma_e && (4.3.1) \\ \vee \; & S^0 \sigma_d && (4.3.2) \\ \vee \; & S^1 \sigma_e && (4.3.3) \\ \vee \; & S^1 \sigma_d && (4.3.4) \\ \vee \; & \sigma_* && (4.3.5) \end{aligned}$$

As it will turn out, whenever Defn. 4B.1 holds, a solution to the DSCOP exists and can be expressed as (4.2). While (4.3) resembles the expressions Ricker and Rudie [RR07] had, it is not ideal as its last disjunction is not an epistemic formula, and hence not very illuminating, since it doesn't describe the knowledge that an agent must possess. Ultimately we will replace (4.3.5) with an epistemic formula.

The last two ingredients we need before showing that conditional-co-observable is necessary and sufficient to solve DSCOP are two characterizations of "solving" DSCOP. The first characterization arises by noticing that the definition of $L(f_\mathcal{N}/G)$ (Defn. 2A1.1) is such that $L(f_\mathcal{N}/G) = L(E)$, i.e., the joint supervision $f_\mathcal{N}$ solves the DSCOP, iff

$$\begin{aligned} & s \in L(E) \wedge s\sigma \in L(G) \wedge \sigma \in \Sigma_{uc} \\ & \quad \Rightarrow \quad s\sigma \in L(E) && (4.4.1) \\ & s \in L(E) \wedge s\sigma \in L(G) \wedge \sigma \in \Sigma_c \\ & \quad \Rightarrow \quad f_\mathcal{N}(s, \sigma) = \textbf{enable} \Rightarrow s\sigma \in L(E) && (4.4.2) \\ & \qquad \wedge \, f_\mathcal{N}(s, \sigma) = \textbf{disable} \Rightarrow s\sigma \notin L(E) && (4.4.3) \end{aligned}$$

Equation (4.4) expresses that a solution to DSCOP must ensure that uncontrollable events do not lead to illegality (4.4.1), and that if a controllable event is allowed to happen (4.4.2), it leads to a legal string; and if it is prevented from happening (4.4.3), it leads to an illegal string.

On the other hand, we also have that $L(f_\mathcal{N}/G) = L(E)$ iff

$$
\begin{aligned}
& s \in L(E) \wedge s\sigma \in L(G) \wedge \sigma \in \Sigma_{uc} \\
& \quad \Rightarrow \quad s\sigma \in L(E) && (4.5.1) \\
& s \in L(E) \wedge s\sigma \in L(G) \wedge \sigma \in \Sigma_{c} \\
& \quad \Rightarrow \quad s\sigma \in L(E) \Rightarrow f_{\mathcal{N}}(s,\sigma) = \textbf{enable} && (4.5.2) \\
& \qquad \wedge s\sigma \notin L(E) \Rightarrow f_{\mathcal{N}}(s,\sigma) = \textbf{disable} && (4.5.3)
\end{aligned}
$$

Equation (4.5) expresses that a solution to DSCOP must ensure that uncontrollable events do not lead to illegality (4.5.1), and that if a controllable event is legal, it is allowed to happen (4.5.2), and if it is illegal, it is prevented from happening (4.5.3).

We can now proceed to the development of our main result.

Theorem 4B.2
In the conditional architecture, there exists a set $\mathcal{N}$ of n supervisors that solves the DSCOP iff I is controllable and conditional-co-observable.

Moreover, whenever a solution exists, the knowledge-based control policy in (4.2) is a solution.

Again, we emphasize that for the sake of the statement, we take (4.2) and Defn. 4B.1 as given. But we will actually *derive* them during the proof.

Informally, for the necessity part of the proof, we will perform a case analysis on all possible combinations of local control decisions after a string, i.e., on (4.1). The fused decision will imply some global properties of the string, i.e., whether it can be followed by a legal/illegal event, as for this direction, we are assuming the supervisors solve DSCOP. From the local decisions, we can *derive* epistemic characterizations of the supervisors' knowledge in the following way. By feasibility, we know that a supervisor has to issue an identical decision for other strings indistinguishable from the actual string. Then it is possible to extrapolate the possible global decisions at those strings, and hence their global properties. The common global properties of all these strings is then the supervisor's knowledge. After we exhaust all combinations of local control decisions, we will obtain the proposed epistemic expression of conditional-co-observability.

We now provide our formal proof.

Proof.

Conditional-coobservability is necessary ($\Rightarrow$)

For the proof of necessity, we use (4.4) as the characterization of what it means to solve DSCOP. Condition (4.4.1) directly implies controllability. What is left is to show that (4.4.2) and (4.4.3) imply conditional-co-observability.

Suppose there exists such a set $\mathcal{N} = (f_1, \ldots, f_n)$ of n supervisors, such that (4.4.2) and (4.4.3) hold.

Although the proof would be much easier by showing that if otherwise conditional-co-observability fails, then a contradiction arises, for our purpose, we explicitly *derive* conditional-co-observability as a necessity.

Consider some $s \in L(E)$, $\sigma \in \Sigma_c$ such that $s\sigma \in L(G)$. Let w be the world s leads to; since $s \in L(E)$, $w_e \in Q^E$.

Now we perform a case analysis on the possible combinations of local decisions, in the same order as specified in the fusion rule (4.1).

In each case there will be a specific supervisor i that we will focus our attention on. For this supervisor we will consider strings $s' \in L(E)$ such that $P_i(s') = P_i(s)$ and consider the global decision $f_{\mathcal{N}}(s', \sigma)$. Either the global decision at s' is the same as that at s, or they differ. We will further assume that $s'\sigma \in L(G)$ so that the difference, if present, is material by (4.4). Since the string s' is arbitrary, if we can conclude proposition ϕ for s', we can conclude it for all $w' \in [w]_i$, and hence conclude $K_i\phi$ for w.

Since the control policy for each controllable event is designed individually [RR22a], for brevity, going forward when we speak of supervisors i, j, we implicitly mean $i, j \in \mathcal{N}_\sigma$, i.e., supervisor i and supervisor j each controls σ.

Case A. Suppose that for some i, the local decision is $f_i(P_i(s), \sigma) =$ **on**, and consequently by (4.1.1) the fused decision must be $f_\sigma(s) =$ **enable**.

By feasibility, it must be that $f_i(P_i(s'), \sigma) =$ **on** as well, and consequently $f_\sigma(s') =$ **enable** as well. That is, in this case the global decision at s' must be the same as at s. Because we assumed the supervisors solve the DSCOP, it must be that $s'\sigma \in L(G) \Rightarrow s'\sigma \in L(E)$ by (4.4.2), which is equivalent to $s'\sigma \notin L(G) \vee s'\sigma \in L(E)$. Hence, for this particular agent i, at world w, we have

$$K_i\sigma_e = K_i^0\sigma_e$$

and

$$\neg\, K_i\sigma_d = \overline{K_i^0\sigma_d}$$

holds. Hence

$$f_i(P_i(s), \sigma) = \textbf{on} \Rightarrow K_i^0 \sigma_e \wedge \overline{K_i^0 \sigma_d} \tag{4.6}$$

Also, from $K_i^0 \sigma_e$ we have $S^0 \sigma_e$, which is exactly (4.3.1).

Case B. The case in which for some i, $f_i(P_i(s), \sigma) = \textbf{off}$ is reasoned analogously to Case A, from which it follows that

$$f_i(P_i(s), \sigma) = \textbf{off} \Rightarrow \overline{K_i^0 \sigma_e} \wedge K_i^0 \sigma_d \tag{4.7}$$

and (4.3.2).

Note that in Case A and Case B, when defining the control protocol (4.2) we explicitly excluded the situation where $K_i^0 \sigma_e \wedge K_i^0 \sigma_d$ holds at world w, which is equivalent to $K_i(\sigma_e \wedge \sigma_d)$ and implies that $\sigma_e \wedge \sigma_d$, i.e., $(\neg\, \sigma_G \vee \sigma_E) \wedge \neg\, \sigma_E$, which, by disjunctive syllogism (modus tollendo ponens), is in turn equivalent to $\neg\, \sigma_G$. Although this contradicts the fact that $s\sigma \in L(G)$ anyway and thus is redundant, we nonetheless choose to preclude $K_i^0 \sigma_e \wedge K_i^0 \sigma_d$ explicitly in (4.2.1) and (4.2.2).

Case C. Suppose that for some i, $f_i(P_i(s), \sigma) = \textbf{weak on}$, but for no j, $f_j(P_j(s), \sigma) = \textbf{on}, \textbf{off}, \textbf{weak off}$. Consequently by (4.1.3) the fused decision must be $f_\sigma(s) = \textbf{enable}$.

Suppose that the global decision at s' differs from that at s, i.e., suppose $f_{\mathcal{N}}(s', \sigma) = \textbf{disable}$, which is equivalent to $s'\sigma \notin L(E)$ by (4.4.3). Moreover, suppose that the difference is material, i.e., suppose $s'\sigma \in L(G)$. That is, we have assumed that $\sigma_{d!}$. Then, since supervisor i's decision for s' cannot be different from that supervisor's decision for s—by feasibility, there must be some supervisor j other than i such that supervisor j's decision for s' differs from that supervisor's decision for s, i.e., $f_j(P_j(s'), \sigma) = \textbf{off}$. By the argument in case B applied to s', we have $K_j^0 \sigma_d$ (which is $O^0 \sigma_d$ because $j \neq i$) hold at w', provided the assumption that $\sigma_{d!}$ holds at w', i.e., $\sigma_{d!} \Rightarrow O^0 \sigma_d$. Hence, for this particular agent i, at world w, we have $K_i(\sigma_{d!} \Rightarrow O^0 \sigma_d) = K_i^1 \sigma_e$, i.e.,

$$f_i(P_i(s), \sigma) = \textbf{weak on} \Rightarrow K_i^1 \sigma_e. \tag{4.8}$$

Further, from $K_i^1 \sigma_e$ we have

$$S^1 \sigma_e$$

(i.e., (4.3.3)) holds at w.

Case D. The case in which for some i, $f_i(P_i(s), \sigma) = \textbf{weak off}$, but for no j, $f_j(P_j(s), \sigma) = \textbf{on}, \textbf{off}, \textbf{weak on}$ is reasoned analogously as in Case C. We can derive that

$$f_i(P_i(s), \sigma) = \textbf{weak off} \Rightarrow K_i^1 \sigma_d \tag{4.9}$$

and (4.3.4).

Case E. Finally, suppose that $f_i(P_i(s), \sigma) =$ **abstain** for all i. If $\mathtt{dft} =$ **enable** $= f_{\mathcal{N}}(s, \sigma)$, it must be $s\sigma \notin L(G) \vee s\sigma \in L(E)$ by (4.4.2). If $\mathtt{dft} =$ **disable** $= f_\sigma(s)$, it must be $s\sigma \notin L(E)$ by (4.4.3), which gives $\phi = \sigma_d$. Thus we derive (4.3.5).

Conditional-coobservability is sufficient ($\Rightarrow$)

For the proof of sufficiency, we use (4.5) as the characterization of what it means to solve DSCOP. Controllability directly implies condition (4.5.1). What is left is to show that whenever conditional-co-observability holds, our knowledge-based control policy (4.2) is a solution to the problem under the required fusion rule (4.1), i.e., it satisfies (4.5.2) and (4.5.3).

Before proceeding to the proof, as we have promised, we need to show how the knowledge-based control policy was derived.

Recall the proof for the necessity part. Gathering (4.6) to (4.9), we have that for any solution f_i, it is necessary that

$$\begin{aligned}
f_i(P_i(s), \sigma) = & \quad \textbf{on} \Rightarrow K_i^0\sigma_e \wedge \overline{K_i^0\sigma_d} \\
f_i(P_i(s), \sigma) = & \quad \textbf{off} \Rightarrow \overline{K_i^0\sigma_e} \wedge K_i^0\sigma_d \\
f_i(P_i(s), \sigma) = & \ \textbf{weak on} \Rightarrow K_i^1\sigma_e \\
f_i(P_i(s), \sigma) = & \ \textbf{weak off} \Rightarrow K_i^1\sigma_d
\end{aligned}$$

To recover a design of the agents, we essentially need to establish the implication in the other direction, with the additional requirement that the cases of the definition must be exhaustive and mutually exclusive. We attempt to establish the mutual exclusiveness in the most obvious way: i.e., define the control protocol as (4.2). It is clearly fully defined due to the "otherwise" clause.

Then note that conditional-co-observability is equivalent to

$$\begin{aligned}
(I, w) \models \bigvee_{i \in \mathcal{N}_\sigma} & K_i^0\sigma_e \vee K_i^0\sigma_d \\
& \vee K_i^1\sigma_e \vee K_i^1\sigma_d \\
& \vee \sigma_*.
\end{aligned} \tag{4.10}$$

So, when (4.10) holds, according to (4.2), $f_i(P_i(s, \sigma)) =$ **abstain** if and only if

$$(I, w) \models K_i^0\sigma_e \wedge K_i^0\sigma_d \tag{4.11.1}$$

$$\vee \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge K_i^1\sigma_e \wedge K_i^1\sigma_d \tag{4.11.2}$$

$$\vee \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge \overline{K_i^1\sigma_e} \wedge \overline{K_i^1\sigma_d} \wedge \sigma_*, \tag{4.11.3}$$

hence we can replace the "otherwise" clause in (4.2) by (4.11). We now reproduce the knowledge-based protocol as follows:

$$\mathcal{KP}_i(w,\sigma) = \begin{cases} \textbf{on} & \text{if } (I,w) \models K_i^0\sigma_e \wedge \overline{K_i^0\sigma_d} & (4.12.1)\\ \textbf{off} & \text{if } (I,w) \models \overline{K_i^0\sigma_e} \wedge K_i^0\sigma_d & (4.12.2)\\ \textbf{weak on} & \text{if } (I,w) \models \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge K_i^1\sigma_e \wedge \overline{K_i^1\sigma_d} & (4.12.3)\\ \textbf{weak off} & \text{if } (I,w) \models \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge \overline{K_i^1\sigma_e} \wedge K_i^1\sigma_d & (4.12.4)\\ \textbf{abstain} & \text{if } (I,w) \models K_i^0\sigma_e \wedge K_i^0\sigma_d & (4.12.5)\\ & \quad \vee \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge K_i^1\sigma_e \wedge K_i^1\sigma_d & (4.12.6)\\ & \quad \vee \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge \overline{K_i^1\sigma_e} \wedge \overline{K_i^1\sigma_d} \wedge \sigma_*, & (4.12.7) \end{cases}$$

By intentionally putting (4.11) into disjunctive normal form, we reveal two distinct roles of the **abstain** decision. First, as discussed, the situation (4.11.1) cannot happen unless $s\sigma \notin L(G)$. Then, (4.11.2) is the true abstaining decision, since regardless of the legality of σ, there is always some other supervisor that can make a correct decision. In the case of (4.11.3), since all epistemic formulae are negated, we take, for now, that (4.11.3) expresses the situation that the supervisor is in a "doesn't know" situation.

To verify the correctness of our knowledge-based control protocol (4.12), we perform a case analysis over conditional-co-observability. The trick is how to split the cases so that in each case we can infer the local decisions. Then a natural way to proceed is to split the cases according to the lines defining (4.12). Recall that the cases are exhaustive given conditional-co-observability, since that is how (4.11) was obtained.

To establish (4.5.2) and (4.5.3), fix an $s \in L(E)$, $\sigma \in \Sigma_c$ such that $s\sigma \in L(G)$. Let w be the world s leads to. Since conditional-co-observability (4.10) holds at w, there is a supervisor i for which $K_i^0\sigma_e \vee K_i^0\sigma_d \vee K_i^1\sigma_e \vee K_i^1\sigma_d \vee \sigma_*$ holds. Consider the following cases, which, as argued above, are exhaustive. We will show that in each case, (4.5.2) and (4.5.3) hold.

Case 1. If $K_i^0\sigma_e \wedge \overline{K_i^0\sigma_d}$ (i.e., (4.12.1)), then locally we have that i issues **on** by (4.12.1), and by $K_i^0\sigma_e$ globally we have $s\sigma \in L(E)$. So (4.5.3) holds vacuously. To show (4.5.2), it suffices to show that the fused decision is **enable**. We argue that it is impossible for there to be some supervisor j that issues **off**. If that were possible, we'd have $K_j^0\sigma_d$, which, together with $K_i^0\sigma_e$, would imply $s\sigma \notin L(G)$, contradicting the assumption.

Case 2. The case $\overline{K_i^0\sigma_e} \wedge K_i^0\sigma_d$ (i.e., (4.12.2)) is reasoned analogously to Case 1.

Case 3. If $\overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge K_i^1\sigma_e \wedge \overline{K_i^1\sigma_d}$ (i.e., (4.12.3)), then locally we have that i issues **weak on** by (4.12.3).

a) If $s\sigma \in L(E)$, we desire that the fused decision be **enable**.

i. If some supervisor j issues the decision **on**, then as argued in Case 1, there cannot be a third supervisor k issuing the decision **off**, thus the fused decision must be **enable** by (4.1.1).

ii. If some j issues the decision **off**, then, by (4.12.2) we'd have $\overline{K_j^0\sigma_e} \wedge K_j^0\sigma_d$, and the argument can be established by letting j play the role of i in Case 2.

iii. If some j issues the decision **weak off**, then by (4.12.4), we have $K_j^1\sigma_d$, which implies that $\sigma_{e!} \Rightarrow \bigvee_{k\neq j} K_k\sigma_e$. Since we do have $\sigma_{e!}$ by $s\sigma \in L(E)$, there is some k such that $K_k\sigma_e$. Moreover, we have that $\overline{K_k\sigma_d}$ since $s\sigma \in L(G)$. Then the argument can be established by letting k play the rule of i in Case 1.

iv. Otherwise, the fused decision must be **enable**, as desired.

b) If $s\sigma \notin L(E)$, we desire that the fused decision be **disable**. The argument is analogous to Case 3(a)

Case 4. The case $\overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge \overline{K_i^1\sigma_e} \wedge K_i^1\sigma_d$ (i.e., (4.12.4)) is reasoned analogously to Case 3.

Case 5. The case $K_i^0\sigma_e \wedge K_i^0\sigma_d$ (i.e., (4.12.5)) contradicts $s\sigma \in L(G)$ as argued.

Case 6. If $\overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge K_i^1\sigma_e \wedge K_i^1\sigma_d$ (i.e, (4.12.6)), then locally we have that i issues **abstain** by (4.12.6). Recall that by our analysis of the **abstain** decision, this represents the true abstaining decision. The argument is established analogously to Case 3 and Case 4.

Case 7. If $\overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge \overline{K_i^1\sigma_e} \wedge \overline{K_i^1\sigma_d} \wedge \sigma_*$ (i.e., (4.12.7)), then locally we have that i issues **abstain** by (4.12.7). Recall that by our analysis of the **abstain** decision, this represents the "doesn't know" decision.

a) If $* = e$, i.e., $s\sigma \in L(E)$, we desire that the fused decision be **enable**. If there is some supervisor j that issues the decision **on**, **off**, **weak off**, then the fused decision is **enable** by an argument exactly the same as Case 3(a). If there is some supervisor j that issues the decision **weak on**, then the fused decision is **enable** by letting j play the rule of i in Case 3. In

the last case where all supervisors issue the decision **abstain**, the desired fused decision can be achieved by setting dft to **enable** in (4.1.5).

b) The case that $* = d$, i.e., $s\sigma \notin L(E)$, is argued analogously. □

We have thus completed the derivation of the problem solvability condition and a knowledge-based control protocol from the fusion rule. Additionally, the process is entirely methodologically and at no point requires human cleverness. We hence propose this process as a prototypical example of a more uniform, formal approach to study other decentralized architectures.

One drawback of the expressions (4.3.5) and (4.12.7) is that they contain the non-epistemic term σ_* (which becomes either σ_e or σ_d), and therefore does not provide any insight into what knowledge an agent must possess to support the agent's actions. Consequently, we interpreted the situation (4.12.7) as that the supervisor possesses no knowledge. However, we will demonstrate that it does, in fact, possess some knowledge.

We resume the proof of Case E and show that further progression will lead to an epistemic expression in place of (4.3.5). We start by aggregating local properties of strings s' indistinguishable from s to a specific supervisor i as we have done for all other cases, so that we can obtain an epistemic expression.

Similar to the argument in Case C, suppose the global decision at s' differs from that at s, i.e., suppose $f_{\mathcal{N}}(s', \sigma) =$ **disable**, which is equivalent to $s'\sigma \notin L(E)$ by (4.5.3). Moreover, suppose that the difference is material, i.e., suppose $s'\sigma \in L(G)$. That is, we have assumed $\sigma_{d!}$. Now let j be the supervisor such that $f_j(P_j(s'), \sigma) =$ **off** or **weak off** (these are the only two possibilities to get $f_{\mathcal{N}}(s\,\sigma) =$ **disable** by (4.1)). Applying the argument in Case B or Case C to s' with j playing the role of i, we have $K_j\sigma_d$ (which is $O\sigma_d$ because $j \neq i$) or $K_j(\sigma_{d!} \Rightarrow O(\sigma_d))$ (which is $O(\sigma_{d!} \Rightarrow O\sigma_d)$).

Hence at w, if dft $=$ **enable**, we have

$$K_i^2\sigma_e \coloneqq K_i(\sigma_{d!} \Rightarrow O\sigma_d \vee O(\sigma_{d!} \Rightarrow O\sigma_d));$$

and by a similar argument, if dft $=$ **disable**, we have

$$K_i^2\sigma_d \coloneqq K_i(\sigma_{e!} \Rightarrow O\sigma_e \vee O(\sigma_{e!} \Rightarrow O\sigma_e)).$$

I.e.,

$$f_i(P_i(s), \sigma) = \textbf{abstain} \Rightarrow K_i^2\sigma_* \tag{4.13}$$

Also, we have $S^2\sigma_*$.

That is, (4.3) can now be formally replaced with

$$\begin{aligned}(I,w) \models \quad & S^0\sigma_e \vee S^0\sigma_d \\ & \vee S^1\sigma_e \vee S^1\sigma_e \\ & \vee S^2\sigma_*,\end{aligned} \tag{4.14}$$

where $*$ is either e or d.

With the reformulated expression of conditional-co-observability, in exactly the same way we obtained (4.11) in the original proof, now we can further establish that $f_i(P_i(s,\sigma)) =$ **abstain** if and only if

$$(I,w) \models K_i^0\sigma_e \wedge K_i^0\sigma_d \tag{4.15.1}$$

$$\vee \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge K_i^1\sigma_e \wedge K_i^1\sigma_d \tag{4.15.2}$$

$$\vee \overline{K_i^0\sigma_e} \wedge \overline{K_i^0\sigma_d} \wedge \overline{K_i^1\sigma_e} \wedge \overline{K_i^1\sigma_d} \wedge K_i^2\sigma_*. \tag{4.15.3}$$

We have discussed the meaning of the first two disjuncts in the proof. In the last case, **abstain** is instead used as an even weaker version of **weak on** or **weak off**, as indicated by the expression $K_i^2\sigma_*$. But in any case, it is not entirely illustrative to say that the supervisor "doesn't know", which is what Ricker and Rudie [RR07] called the "abstain" decision.

Finally, while we only demonstrated the epistemic formalism on the conditional architecture, the approach can be systematically extended over the more general inference-based architectures [KT07]. One note is that as the level of inference increases, the recursive structure of epistemic expressions K_i^M and K_i^N explodes in size quickly. Hence it is not advised to explicitly expand out the expression for evaluation, but to employ dynamic programming.

4B1 A Visualization to Aid in the Revision of Problem Requirements

A procedure to synthesize a sublanguage is already provided by Takai and Kumar [TK08], presented in a non-epistemic formalism. We will demonstrate, with an example, that our epistemic logic formalism — or more specifically, the Kripke structures — can provide a visual aid to understand the approach of Takai and Kumar [TK08]. We also adopt the same approach of Takai and Kumar [TK08] to the synthesis of a superlanguage. While we will not pursue it in the demonstration, one will see that the same methodology can be extended to revise the problem requirement to synthesize incomparable languages. We refer the reader to [RR21] for an

even more compact visualization, underlying which is nonetheless the epistemic interpretation.

Consider the following example. The set of possible events is $\Sigma = \{\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma, \mu\}$, observable event sets are $\Sigma_{1,o} = \{\mu\}$, $\Sigma_{2,o} = \{\beta_1, \beta_2\}$, and controllable event sets are $\Sigma_{1,c} = \Sigma_{2,c} = \{\gamma\}$. The plant G and legal language specification E are captured in the automaton $G' = G \times P_1(G) \times P_2(G)$ depicted in Fig. 4.1. The language $L(E)$ is marked by states with double borders. Since the states of G' are the worlds of its Kripke structure, we can embed the Kripke structure in the representation of G' as shown in Fig. 4.1. The problem is to determine whether there exist two supervisors with the observable and controllable event sets given above, such that $L(f_{\mathcal{N}}/G) = L(E)$.

Let us focus on γ since it is the only controllable event. Hence we focus on states 0, 2, 3, 4, 5, since these are the states where γ can happen.

In state 4 (resp. 5), supervisor 2 can enable (resp. disable) γ. In state 0, supervisor 1 can enable γ. But in states 2, 3, which are indistinguishable to both supervisors, since they are both in the same equivalence classes ($\mathbf{M_1}$ for supervisor 1 and $\mathbf{B_0}$ for supervisor 2), neither supervisor 1 nor 2 can control γ unambiguously. Hence the language $L(E)$ is not conditional-co-observable.

The representation of G' and the epistemic interpretation of conditional control decisions provides guidance for how to modify the control requirement to obtain a conditional-co-observable language.

If we are looking for a sublanguage, we can only make legal states illegal but not vice versa. By our previous analysis, at least one supervisor is able to make a correct control decision unambiguously in states $S = \{0, 1, 4, 5, 7, 8, 10\}$, hence all we need to worry about are the states in the set $\mathbf{M_1} - S = \mathbf{B_0} - S = \{2, 3\}$. To resolve the conflict that γ is legal at state 3 but illegal at state 2, we can make state 7 illegal.

To see how making state 7 illegal gives a conditional-co-observable sublanguage, let's look at states in $\mathbf{M_1}$ and $\mathbf{B_0}$. At states in $\mathbf{M_1}$, γ is illegal at states $2, 3, 5$ but is legal at state 4. With only binary control decisions, supervisor 1 cannot possibly make an unambiguous decision. We can see that supervisor 2 is in a similar situation by examining states in $\mathbf{B_0}$. However, with the ability to infer the knowledge of other supervisors and the conditional decisions at their disposal, the desired control requirement can be achieved. Suppose that supervisor 1 is an intelligent being, and let's imagine how the intelligent being may attempt to solve the dilemma. Consider what if supervisor 1 were to try to guess the legality of γ after it sees μ. Clearly this guess is not always correct, i.e., it is false at exactly state 4. But knowing that the other supervisor can unambiguously enable γ if the plant is indeed at state 4 supervisor 1 is then able to focus on only the rest of the states in $\mathbf{M_1}$, and fortunately,

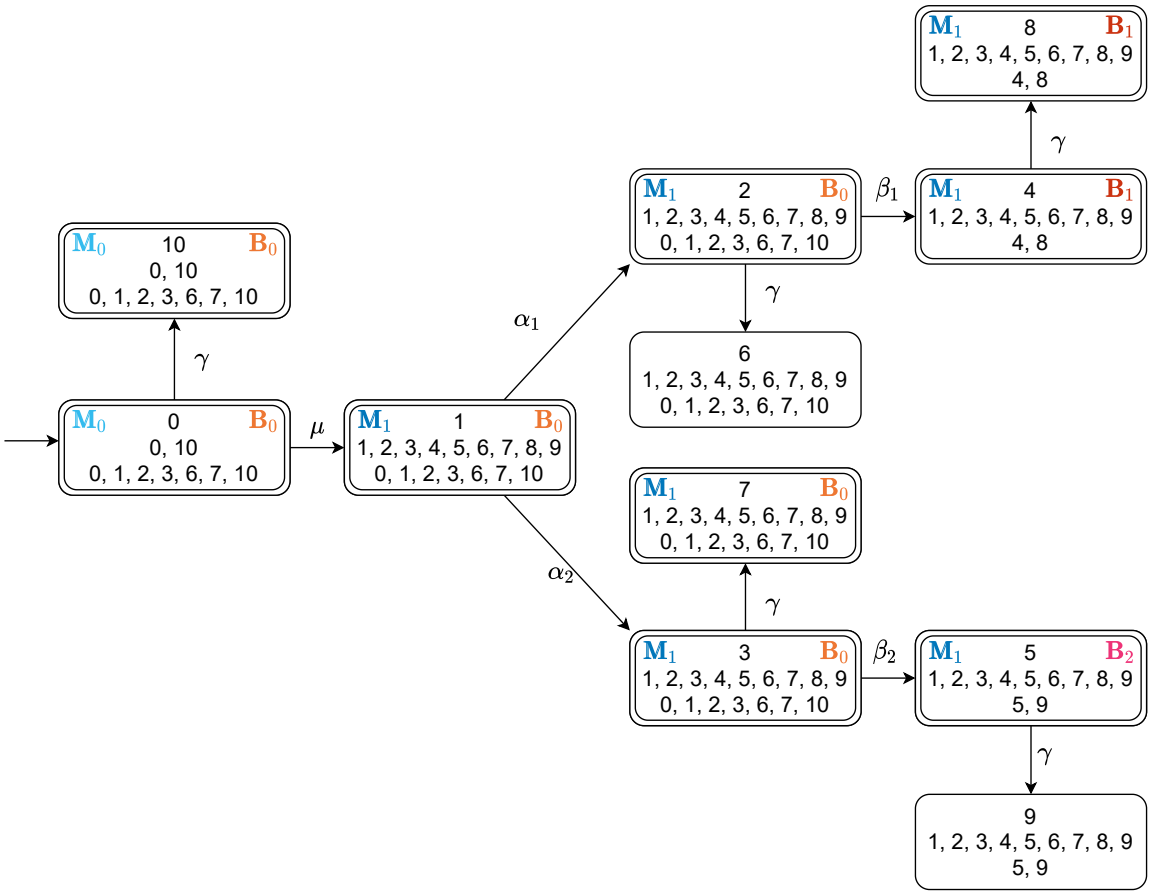

Figure 4.1: The automaton $G' = G \times P_1(G) \times P_2(G)$ with its corresponding Kripke structure embedded in it. A state $(q_G, q_1^{obs}, q_2^{obs})$ is represented in the figure with q, q_1^{obs}, q_2^{obs} stacked vertically in that order. The equivalence classes are marked according to the following rule: a state is marked at the upper left (resp. upper right) corner according to its containing equivalence class formed by the accessibility relation $\sim_1$ (resp. $\sim_2$); the symbols for the equivalence classes are deliberately chosen, so, for instance, the states that supervisor 2 thinks the plant could be in after it sees β_1 are in the equivalence class $\mathbf{B_1}$.

its guess is correct in all of them. Hence supervisor 1 can confidently disable γ at states in $\mathbf{M_1}$ unambiguously, knowing its mistake would be corrected by the other supervisor. Similar reasoning is also carried out by supervisor 2.

The design of the fusion rule is exactly to allow the correction of mistakes. A **weak off** is issued by a supervisor knowing that if disabling the event is incorrect then another supervisor can correct the first supervisor by a definite **on** decision.

Formally, with state 7 made illegal, states $\{2, 3\}$ are unambiguous. However, since the set $\{2, 3\}$ is a proper subset of both $\mathbf{M_1}$ and $\mathbf{B_0}$, and states in both sets $\mathbf{M_1}$ and $\mathbf{B_0}$ remain ambiguous, the conditional decision, i.e., **weak off** has to be issued at states in the set $\mathbf{M_1}$ (resp. $\mathbf{B_0}$) by supervisor 1 (resp. supervisor 2).

If it is reasonable for the problem at hand to admit a solution that is not necessarily a sublanguage, we can also make state 6 legal too. By similar reasoning as we just did, supervisor 1 should issue decision **weak on** at states 2, 3; and supervisor 2 can issue decision **on** at states in the set $\mathbf{B_1}$, since this set is no longer ambiguous.

4C Conclusion

In this chapter, we discuss how decentralized control problems can benefit from the use of epistemic logic.

We point out that epistemic logic can be used to discuss not only some specific classes of DSCOP [RR00; RR07], but also it can be used more broadly to describe other classes of decentralized supervisory control problems. The use of epistemic formalism provides a formal approach towards describing decentralized problems, and consequently allows mechanical derivation of problem solvability conditions and solution constructions. The derivation also results in direct coupling between the expression of problem solvability condition and the expression describing the control policies. This line-by-line coupling allows us to use the same expression throughout the discussions of proving necessary and sufficient conditions, of describing the algorithm to construct the supervisors, and of verifying the correctness of the algorithm.

From the forgoing discussions, we would expect other decentralized control or diagnosis conditions could be treated in a comparable fashion. For instance, consider the work of Kumar and Takai [KT07], which is more general than that of Yoo and Lafortune [YL05]. We developed our epistemic expressions based on Yoo and Lafortune [YL05] because it is simpler and thus we are able to demonstrate our key ideas without more complex (yet not conceptually different) technical development.

The same principles demonstrated here could apply to Kumar and Takai [KT07] as well. The only technical difference is that one would need to use a finer, (possibly infinite) string-based Kripke structure as described by Ricker and Rudie [RR00], along with a corresponding definition of relations $\sim_i$.

Casting the decentralized problem the way we did makes it easier to understand the reasoning behind various control decisions. We believe that one advantage of our framework is that in trying to come up with solutions to future DES problems, this framework can aid in going directly from a working supervisor solution to the necessary and sufficient conditions that would match such a solution. Moreover, if the constraints of some given problem are not met (and hence that problem is not solvable as is using decentralized control), our model makes it more apparent how to alter the constraints in a way that is meaningful for the application at hand.

References

[Cie+88] R. Cieslak, C. Desclaux, A. S. Fawaz, and P. Varaiya. “Supervisory control of discrete-event processes with partial observations”. In: *IEEE Transactions on Automatic Control* 33.3 (Mar. 1988), pp. 249–260. DOI: 10.1109/9.402. [cit. on p. 59. 60].

[KT07] R. Kumar and S. Takai. “Inference-Based Ambiguity Management in Decentralized Decision-Making: Decentralized Control of Discrete Event Systems”. In: *IEEE Transactions on Automatic Control* 52.10 (Oct. 2007), pp. 1783–1794. DOI: 10.1109/TAC.2007.906158. [cit. on p. 61. 72. 75].

[PKK97] J. H. Prosser, M. Kam, and H. G. Kwatny. “Decision fusion and supervisor synthesis in decentralized discrete-event systems”. In: *Proceedings of the American Control Conference*. IEEE, 1997. DOI: 10.1109/ACC.1997.608978. [cit. on p. 59. 60].

[RR00] S. L. Ricker and K. Rudie. “Know means no: Incorporating knowledge into discrete-event control systems”. In: *IEEE Transactions on Automatic Control* 45.9 (2000), pp. 1656–1668. DOI: 10.1109/9.880616. [cit. on p. 75].

[RR07] S. L. Ricker and K. Rudie. “Knowledge Is a Terrible Thing to Waste: Using Inference in Discrete-Event Control Problems”. In: *IEEE Transactions on Automatic Control* 52.3 (Mar. 2007), pp. 428–441. DOI: 10.1109/TAC.2007.892371. [cit. on p. 60. 64. 72. 75].

[RR21] K. Ritsuka and Karen Rudie. “A Visualization of Inference-Based Supervisory Control in Discrete-Event Systems”. In: *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, Dec. 2021. DOI: 10.1109/cdc45484.2021.9683210. [cit. on p. 72].

[RR22a] K. Ritsuka and Karen Rudie. *A correspondence between control and observation problems in decentralized discrete-event systems*. 2022. arXiv: 2204.10792 [eess.SY]. [cit. on p. 66].

[RR22c] K. Ritsuka and Karen Rudie. "Epistemic interpretations of decentralized discrete-event system problems". In: *Discrete Event Dynamic Systems* 32.3 (June 2022), pp. 359–398. DOI: 10.1007/s10626-022-00363-7. [cit. on p. 60. 61].

[RW92] K. Rudie and W. M. Wonham. "Think globally, act locally: decentralized supervisory control". In: *IEEE Transactions on Automatic Control* 37.11 (1992), pp. 1692–1708. DOI: 10.1109/9.173140. [cit. on p. 59. 60].

[TK08] Shigemasa Takai and Ratnesh Kumar. "Synthesis of Inference-Based Decentralized Control for Discrete Event Systems". In: *IEEE Transactions on Automatic Control* 53.2 (Mar. 2008), pp. 522–534. DOI: 10.1109/tac.2007.915171. [cit. on p. 72].

[YL02] T.-S. Yoo and Stéphane Lafortune. "A General Architecture for Decentralized Supervisory Control of Discrete-Event Systems". In: *Discrete Event Dynamic Systems* 12.3 (2002), pp. 335–377. DOI: 10.1023/a:1015625600613. [cit. on p. 59. 60].

[YL04] T.-S. Yoo and S. Lafortune. "Decentralized Supervisory Control With Conditional Decisions: Supervisor Existence". In: *IEEE Transactions on Automatic Control* 49.11 (Nov. 2004), pp. 1886–1904. DOI: 10.1109/tac.2004.837595. [cit. on p. 59. 61].

[YL05] Tae-Sic Yoo and S. Lafortune. "Decentralized supervisory control with conditional decisions: supervisor realization". In: *IEEE Transactions on Automatic Control* 50.8 (Aug. 2005), pp. 1205–1211. DOI: 10.1109/tac.2005.852556. [cit. on p. 59. 75].

5 Unification of the Conditional Architecture and Inference-Based Architectures

Kumar and Takai [KT07] demonstrated formally that the conditional architecture sits within the hierarchy of inference-based architectures. However, the conditional architecture is formally specified in a way that is different from how the other inference-based architectures are specified. This difference is manifested in two different forms. Locally, the conditional architecture allows all supervisors to simultaneously **abstain**; whereas other architectures in the hierarchy do not permit simultaneous abstention. Globally, solving a DSCOP with the conditional architecture involves choosing a fusion rule from $f^{\textbf{enable}}$ and $f^{\textbf{disable}}$, i.e., a default decision, for each event; whereas the other inference-based architectures eliminate the necessity for a default decision. Using the reformulation in Section 4B, we are able to explain how the need for a default decision is eliminated, namely by separating **abstain** into a true abstaining decision and a higher-level inferencing decision. Then in the case where all supervisors abstain, it will turn out that some of the **abstain** are actually a higher-level inferencing decision.

The epistemic formalism is illuminating and suggestive. As discussed in the proof, the **abstain** decision plays two roles. Hence we proceed to reformulate the conditional architecture by splitting the roles of the **abstain** decision. Consider the set of control decisions $\mathcal{CD} = \{\, \textbf{enable}_0,\, \textbf{disable}_0,\, \textbf{enable}_1,\, \textbf{disable}_1,\, *_2,\, \perp \,\}$ (we intentionally use tokens distinguished from what we have been using), where $*_2$ has to be chosen from $\textbf{enable}_2$ and $\textbf{disable}_2$ for each $\sigma \in \Sigma_c$, the fusion rule f_σ for σ is defined to take cd as the fused decision if cd_i is the local decision with the smallest i. Note that this implies that 1) if, say, $\textbf{enable}_i$ is the local decision with the smallest i, then there must be no supervisor issuing $\textbf{disable}_i$; and 2) at least one supervisor must issue a decision that is not $\perp$, i.e., not all supervisors abstain.

We intentionally used different symbols for the control decisions in the reformulation so it is clear whether a symbol refers to a decision in the original or the re-formulation.

It can be seen intuitively that the original and the reformulation are equivalent.

First, the original can be embedded in (translated to) the reformulation by the

following mapping:

$$\begin{aligned} \textbf{on} &\mapsto \textbf{enable}_0 \\ \textbf{off} &\mapsto \textbf{disable}_0 \\ \textbf{weak on} &\mapsto \textbf{enable}_1 \\ \textbf{weak off} &\mapsto \textbf{disable}_1 \\ \textbf{abstain} &\mapsto *_2 \end{aligned}$$

and vice versa

$$\begin{aligned} \textbf{enable}_0 &\mapsto \textbf{on} \\ \textbf{disable}_0 &\mapsto \textbf{off} \\ \textbf{enable}_1 &\mapsto \textbf{weak on} \\ \textbf{disable}_1 &\mapsto \textbf{weak off} \\ *_2 &\mapsto \textbf{abstain} \\ \bot &\mapsto \textbf{abstain} \end{aligned}$$

In particular, the second mapping shows the two roles that **abstain** plays, where $\bot$ is the true abstaining decision. With this reformulation, it is not possible for all supervisors to simultaneously issue the $\bot$ decision, i.e., they can't all truly be issuing a "don't know" decision.

For a more formal justification behind the embedding argument and a general discussion on its application in demonstrating relative strength of two architectures (especially equivalence), we defer to Chapter 8.

It is now clear how the conditional architecture belongs to the hierarchy of inference-based architectures. Consider arranging the control decisions in any inference-based architecture into two chains:

$$\textbf{enable}_0 \leq \textbf{disable}_1 \leq \textbf{enable}_2 \leq \ldots \leq a_N$$

and

$$\textbf{disable}_0 \leq \textbf{enable}_1 \leq \textbf{disable}_2 \leq \ldots \leq b_M.$$

Then the fusion rule of an inference-based architecture can be considered as an arbiter that resolves conflicts in local decisions, and the problem solvability condition for that architecture is essentially requiring that the conflicts are always resolvable. Since it is the supervisors that infer, and the architecture is arbitrating, we call an inference-based architecture an *arbitration architecture*, and call its respective problem solvability *co-inferability*. Then an arbitration architecture can be identified by a pair of numbers (N, M) that describe the chains above. Hence the conditional architecture, based on the choice of x_i for an event $\sigma \in \Sigma_c$, is either a $(1, 2)$-

arbitration architecture, or a $(2,1)$-arbitration architecture for σ. Hence we call the conditional architecture "$[(1,2)/(2,1)]$-arbitration architecture".

In the original numbering of the arbitration architectures, Kumar and Takai [KT07] would assign an (N,M)-architecture the number $\max\{\, N, M \,\} - 1$, hence in their numbering many architectures of different capabilities receive the same number.

Our new numbering scheme allows more precise placement of many known architectures within the hierarchy as depicted in Fig. 5.1.

It should be noted that the architectures $(0,1)/(1,0)$ and $(1,2)/(2,1)$ are not each a single architecture but a compound architecture. For example, our analysis above shows that the conditional architecture (now identified as $(1,2)/(2,1)$) selects one of the two $(1,2)$ and $(2,1)$ architecture for each controllable event. If one excludes the compound architectures from Fig. 5.1, then Fig. 5.1 would depict part of the lattice of the hierarchy of arbitration architectures: given two architectures (N_1, M_1) and (N_2, M_2), their join and meet are given by component-wise max and min, respectively. Note that the lattice of co-inferability conditions, ordered by logical implication, is opposite to the lattice of arbitration architectures.

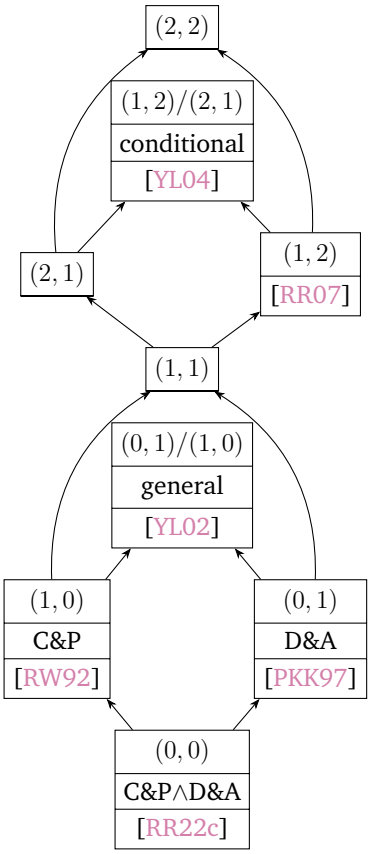

Figure 5.1: Some commonly known architectures are placed within the arbitration hierarchy, ordered by the "more general than" relation. An informal name is given alongside the numeric identification, if such a name has been established for either the architecture itself or for the respective co-inferability condition (e.g., the co-inferability of the $(1, 0)$-arbitration architecture is called C&P co-observability). Some architectures are given with no reference, if they have not been studied in particular, but are placed in the graph for completeness.

References

[KT07] R. Kumar and S. Takai. “Inference-Based Ambiguity Management in Decentralized Decision-Making: Decentralized Control of Discrete Event Systems”. In: *IEEE Transactions on Automatic Control* 52.10 (Oct. 2007), pp. 1783–1794. DOI: 10.1109/TAC.2007.906158. [cit. on p. 79. 81].

[PKK97] J. H. Prosser, M. Kam, and H. G. Kwatny. “Decision fusion and supervisor synthesis in decentralized discrete-event systems”. In: *Proceedings of the American Control Conference*. IEEE, 1997. DOI: 10.1109/ACC.1997.608978. [cit. on p. 82].

[RR07] S. L. Ricker and K. Rudie. “Knowledge Is a Terrible Thing to Waste: Using Inference in Discrete-Event Control Problems”. In: *IEEE Transactions on Automatic Control* 52.3 (Mar. 2007), pp. 428–441. DOI: 10.1109/TAC.2007.892371. [cit. on p. 82].

[RR22c] K. Ritsuka and Karen Rudie. “Epistemic interpretations of decentralized discrete-event system problems”. In: *Discrete Event Dynamic Systems* 32.3 (June 2022), pp. 359–398. DOI: 10.1007/s10626-022-00363-7. [cit. on p. 82].

[RW92] K. Rudie and W. M. Wonham. “Think globally, act locally: decentralized supervisory control”. In: *IEEE Transactions on Automatic Control* 37.11 (1992), pp. 1692–1708. DOI: 10.1109/9.173140. [cit. on p. 82].

[YL02] T.-S. Yoo and Stéphane Lafortune. “A General Architecture for Decentralized Supervisory Control of Discrete-Event Systems”. In: *Discrete Event Dynamic Systems* 12.3 (2002), pp. 335–377. DOI: 10.1023/a:1015625600613. [cit. on p. 82].

[YL04] T.-S. Yoo and S. Lafortune. “Decentralized Supervisory Control With Conditional Decisions: Supervisor Existence”. In: *IEEE Transactions on Automatic Control* 49.11 (Nov. 2004), pp. 1886–1904. DOI: 10.1109/tac.2004.837595. [cit. on p. 82].

6 A Visualization of Inference-Based Supervisory Control in Discrete-Event Systems

A visualization to aid in the construction of inference-based decentralized supervisors is presented. In the inference-based architecture, supervisors have different levels of ambiguity, which reflects to what degree a supervisor is confident in its control decision and to what degree a supervisor infers a control decision based on the supervisor's knowledge of another supervisor's control decision.

6A Introduction

The problem of decentralized supervisory control of discrete-event systems, considers restricting a plant's behaviour with a group of local supervisors, and requires each local supervisor to judge, according to its partial observation of the plant, suitable control decisions in order to achieve desired fused decisions.

While it has not been stated explicitly, since the earliest study of decentralized supervisory control, the control architectures have been designed so that supervisors issue control decisions according to their varying degree of "confidence". The earliest architecture considered by Rudie and Wonham [RW92] allows supervisors to disable events when they are in total confidence, entailing that the problem is solvable whenever at all states at which an event must be disabled, at least one supervisor is totally confident that the event must be disabled. Dually, Prosser, Kam, and Kwatny [PKK97] allow supervisors to enable events when they are in total confidence.

It has been known that the architectures of Rudie and Wonham [RW92] and Prosser, Kam, and Kwatny [PKK97] are not compatible, so that there are problems solvable in one architecture but not solvable in the other. To bring both architectures into unity, Yoo and Lafortune [YL02] considered an architecture in which supervisors are allowed to both disable and enable events when they are in total confidence.

Kumar and Takai [KT07] then called the lack of confidence *ambiguity*. They studied and then concluded that "unconfident" supervisors can still contribute in shaping the fused decisions, even when supervisors are ambiguous in some special ways.

Such special ambiguities, as Kumar and Takai [KT07] have put it, come in levels of gradations, so that the control decision of a supervisor who is in an ambiguity of a lower gradation should be preferred over that of one who is in a higher level of ambiguity. This extension is formulated in the N-inferencing architecture by Kumar and Takai [KT07], where N is the highest level of ambiguity one would like to permit.

Kumar and Takai [KT07] gave a verifiable necessary and sufficient condition for a decentralized control problem to be solvable under the N-inferencing architecture, which they called *N-inference-observable*. Later Takai and Kumar [TK08] provide an algorithm to synthesize the supervisors whenever it is possible.

We see that the verification and supervision synthesis processes by Takai and Kumar are described with vigorous formalism, but may not be accessible to DES researchers not already expert in the inferencing architecture. Moreover, the solutions do not offer much insight into understanding why they work. Therefore, this chapter provides a visualization of inference-based supervisory control. The visualization is done for an algorithm that performs both verification and supervision synthesis concurrently, and is slightly modified to guarantee convergence. Potential implications of this modification are discussed as well.

6B Inference-based Architecture

The inference-based architecture, as expressed in Defn. 2A1.1, consists of the set of control decisions $\mathcal{CD} = \{\, \textbf{enable}_i, \textbf{disable}_i \,\}_{i\in\mathbb{N}} \cup \{\, \textbf{abstain} \,\}$, where the control decisions $\textbf{enable}_i$ (resp., $\textbf{disable}_i$) is used by a supervisor with the intention to **enable** (resp., **disable**) an event, and the number i indicates the level of ambiguity of a supervisor. The special decision **abstain** is used to denote a supervisor that refrains from voting. Alternatively, we use the notation $(\textbf{enable}, i)$ to denote $\textbf{enable}_i$. We do similarly for $\textbf{disable}_i$.

As we would like to prioritize more highly the decisions of the supervisors who are more certain, and the higher the ambiguity level of a supervisor, the less certain the supervisor is. We reflect this in the partial ordering $<$ over $\mathcal{CD}$ defined as follows:

- For $cd_1, cd_2 \in \{\, \textbf{enable}, \textbf{disable} \,\}$ and $i, j \in \mathbb{N}$, whenever $i < j$, let
$$(cd_1, i) < (cd_2, j)$$
- For all $(cd, i) \in \mathcal{CD}$, let
$$(cd, i) < \textbf{abstain}$$

That is, a decision with smaller index trumps decisions with larger indices. The relation is illustrated in Fig. 6.1.

With the ordering $<$ over $\mathcal{CD}$, we can loosely express the fusion rule f compactly as

$$f_\sigma(\{\, cd_i \,\}_{i\in\mathcal{N}_\sigma}) := \min\{\, cd_i \,\}_{i\in\mathcal{N}_\sigma}$$

Even though the expression is somewhat lax, the fusion rule is indeed well-defined, as we will demonstrate that the minimal element in $\{\, cd_i \,\}_{i\in\mathcal{N}}$ is unique, at any legal state for any physically possible event. That is, it is impossible for both **enable**$_i$ and **disable**$_i$ to be minimal. Consequently, the notion of *validity*[1] introduced by Kumar and Takai [KT07], which requires that $f_\sigma(\{\, cd_i \,\}_{i\in\mathcal{N}_\sigma})$ be a total function, becomes redundant.

Kumar and Takai [KT07] provide a necessary and sufficient condition for DSCOP to be solvable when the level of ambiguity is at most an arbitrary but fixed N. In particular, there was no known way to determine that there does not exist a number N, such that the level of ambiguity does not exceed N. This condition is called N-inference-observability.

When the legal language is N-inference-observable, the supervisors can be synthesized following Takai and Kumar [TK08].

[1] In their earlier work, Kumar and Takai [KT05] had a different but equivalent notion called *admissibility*. Also note that this is not the notion of validity we recalled in Defn. 2A1.1.

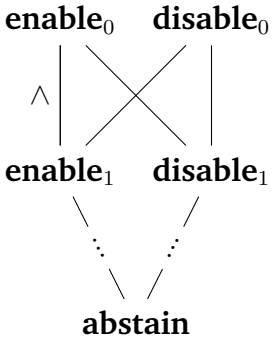

Figure 6.1: The partially ordered set $(\mathcal{CD}, <)$.

6C Visualization

In this section, we illustrate our visualization algorithm step-by-step on the following example: let G be the plant illustrated by Fig. 6.2, where double circled states are those in Q^E. Let $\Sigma_{1,o} = \{\alpha,\ \alpha'\}$, $\Sigma_{2,o} = \{\beta,\ \beta'\}$ and $\Sigma_{1,c} = \Sigma_{2,c} = \{\gamma\}$. This example is derived from Kumar and Takai [KT07, Fig. 1]: we removed half of the plant for compactness

To begin the algorithm, construct the automaton $G' = (\Sigma, Q', \delta', q'_0) := G \times P_1(G) \times \dots P_n(G)$. For our visualization technique, we execute the standard subset construction procedure [HMU06] to represent projections, which results in a computation taking space exponential to the number of states of G and to the number of agents. This automaton has a few nice properties. First, although G' is not necessarily isomorphic to G, we have $L(G') = L(G)$, hence one can always assume, without loss of generality, that the plant is actually implemented as G' instead of G. Then, for all states $q' = (q_G, q_1^{obs}, \dots, q_n^{obs}) \in Q'$ (assuming accessibility of G'), it is always the case that $q \in q_i^{obs}$ for $i \in \mathcal{N}$; conversely, for all $q \in q_i^{obs}$ there always exists a state q' such that $q' = (q_G, \dots, q_i^{obs}, \dots)$. That is, the states Q' record both the plant's

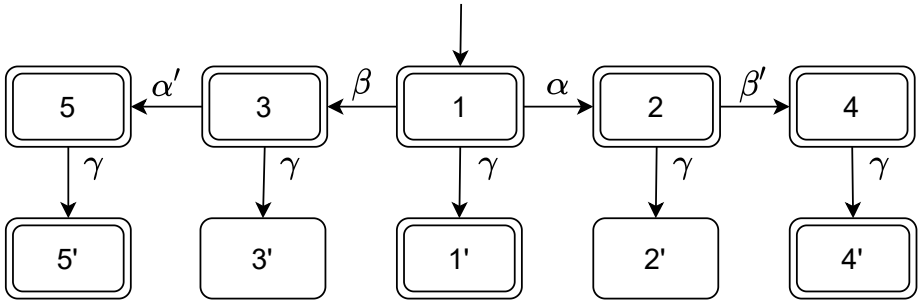

Figure 6.2: Plant G

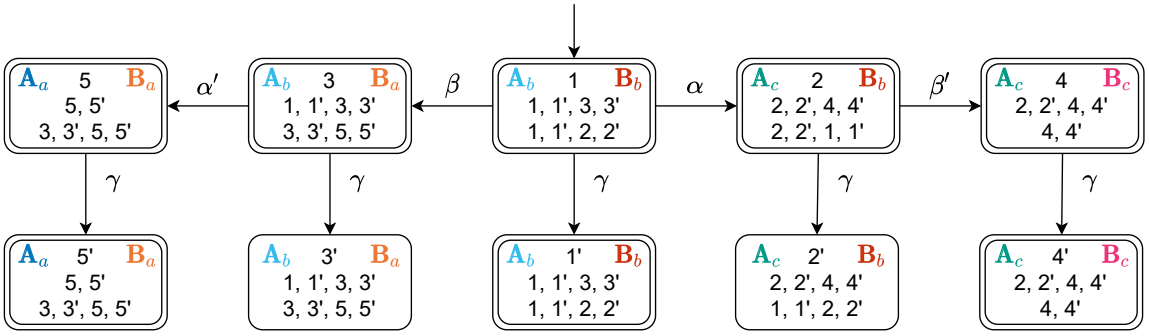

Figure 6.3: Automaton G'. A state $(q_G, q_1^{obs}, q_2^{obs})$ is represented in the figure with q, q_1^{obs}, q_2^{obs} stacked vertically in that order. States are also labelled by their equivalence classes $\ker \sim_1 = \{\mathbf{A_a}, \mathbf{A_b}, \mathbf{A_c}\}$ (resp., $\ker \sim_2 = \{\mathbf{B_a}, \mathbf{B_b}, \mathbf{B_c}\}$) in the upper left (resp., upper right) corner.

actual state, and each supervisor's estimation of the plant's state.

Next, form the (partial) equivalence relations $\{\sim_i\}_{i\in\mathcal{N}}$ over $Q' \subseteq Q \times Q_1^{obs} \times \cdots \times Q_n^{obs}$, so that two states $q' = (q_G, q_1^{obs}, \ldots, q_n^{obs})$ and $p' = (p_G, p_1^{obs}, \ldots, p_n^{obs})$ are related by $\sim_i$ iff $q_i^{obs} = p_i^{obs}$. I.e., $q' \sim_i p'$ whenever q' and p' are indistinguishable to supervisor i. The equivalence class with respect to $\sim_i$ containing a state p is denoted as $[p]_i$, whenever such class exists. We extend the operators $[\cdot]_i$ additively, so that given a set P of states, $[P]_i = \bigcup_{p\in P}[p]_i$.

The automaton G' and the equivalence relations $\sim_i$ are illustrated in Fig. 6.3.

Now we proceed to the actual visualization. For ease of presentation and due to space limitation, we interleave our running example with the formal definition.

Then for each controllable event σ we construct a tabular representation of the equivalence relations $\sim_i$. We dedicate the columns for states in Q', and in each row i indicate in the corresponding columns the equivalence classes the states are in. The table is thus a Venn diagram, displaying overlappings between equivalence classes as vertical adjacencies.

With Fig. 6.3 as the example, the table for event γ is shown in Fig. 6.4. In the tabular diagram, we use different colours to distinguish different equivalence classes. For illustration purposes, we append, two additional rows as follows. The third row indicates the corresponding states for the columns. Since for this particular example, the automaton G' is isomorphic to G, we thus label the columns with states in Q^G instead of Q' for compactness. In the fourth row we indicate whether we desire c to be **enable**'d (as it leads to a legal state), which we indicate with **E**; or **disable**'d (as it leads to an illegal state), indicated with **D**. States at which the event c is not physically possible are indicated by the absence of tokens in the bottom row. For example, states 3, $3'$, 5, $5'$ are indistinguishable by supervisor 2, being in the equivalence class B_a, and we desire that the event c be **enable**'d (resp., **disable**'d) at state 5 (resp., 3), whereas since c is physically impossible at states $3'$ and $5'$, any decision is permitted.

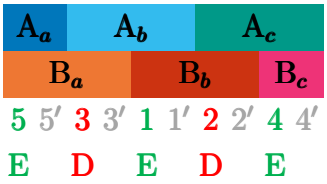

Figure 6.4: Tabular representation of G' in Fig. 6.3

Note that in this example, we enjoy the nice consequence of G' being isomorphic

to G, hence we can label the columns by $q_G \in Q^G$, instead of $(q_G, q_1^{obs}, q_2^{obs}) \in Q^G \times Q_1^{obs} \times Q_2^{obs}$.

For aesthetic purposes, we intentionally arranged the table so that columns of the same equivalence class are adjacent. Note that whether such an arrangement is possible has no implication for problem solvability.

We thus see that the benefit of this tabular representation is to compact information we need to construct the control policy while discard irrelevant information such as transitions.

For states at which the event is physically impossible, since any decision is permitted, we can put off the consideration of these states until the end of the algorithm, and consider only states at which the event is physically possible, thus decisions for the event are pending at those states. Similarly, since illegal states would not be reachable should correct control has been enforced along the way, we only have to consider decisions at legal states. Therefore, as step 0 of the algorithm, we compute $D_0(\sigma)$, the set of states where the desired fused decision should be **disable** and $E_0(\sigma)$, the set of states where the desired fused decision should be **enable**:

$$D_0(\sigma) := \{\, q \in Q' \mid \delta(q_G, \sigma)! \wedge \delta(q_G, \sigma) \notin Q^E \,\}$$
$$E_0(\sigma) := \{\, q \in Q' \mid \delta(q_G, \sigma)! \wedge \delta(q_G, \sigma) \in Q^E \,\}$$

Notice that $D_0(\sigma)$ and $E_0(\sigma)$ are disjoint.

Let $U_0(\sigma) := D_0(\sigma) \cup E_0(\sigma)$ be the collection of states at which local control decisions are yet undetermined. Then, we restrict the partial equivalence relations $\sim_i$ to $U_0(\sigma)$. We then obtain a more compact table. To illustrate this "preprocessing" step, Fig. 6.4 becomes Fig. 6.5 with the irrelevant states removed.

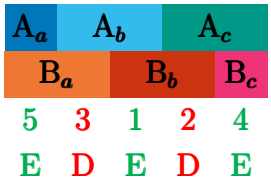

Figure 6.5: Fig. 6.4 after the preprocessing to remove irrelevant states, i.e., step 0 of the algorithm on the example.

Step 0 corresponds to the computation of the following two languages in [KT07;

TK08][2]:

$$\begin{aligned} D'_0(\sigma) &:= \{\, s \in L(E) \mid s\sigma \in L(G) - L(E) \,\} \\ E'_0(\sigma) &:= \{\, s \in L(E) \mid s\sigma \in L(E) \,\} \end{aligned} \tag{6.1}$$

Then we iteratively remove states in the table according to the following rule, until the rule no longer applies. At step $k+1$, consider all equivalence classes of all supervisors. If all states of a class are marked identically, say **E**, then remove the columns corresponding to these states from the table, and let the corresponding supervisor of the class issue the decision **enable**$_k$ for all states in that class, including those removed in the previous steps. Do the same thing for the mark **D**, with the decision **disable**$_k$ instead.

Formally, compute the following set:

$$\begin{aligned} D_{k+1}(\sigma) &:= D_k(\sigma) \cap \left(\bigcap_{i \in \mathcal{N}_\sigma} [E_k(\sigma)]_i \right) \\ &= D_k(\sigma) - \left[D_k(\sigma) - \left(\bigcap_{i \in \mathcal{N}_\sigma} [E_k(\sigma)]_i \right) \right] \\ &= D_k(\sigma) - \left[\bigcup_{i \in \mathcal{N}_\sigma} (D_k(\sigma) - [E_k(\sigma)]_i) \right] \\ &= D_k(\sigma) - \left[\bigcup_{i \in \mathcal{N}_\sigma} (U_k(\sigma) - [E_k(\sigma)]_i) \right] \end{aligned} \tag{6.2.1}$$

where the set $U_k(\sigma) - [E_k(\sigma)]_i$ contains all pending states which are not confused with any state in $E_k(\sigma)$ as perceived by some supervisor i. At these states, supervisor i unambiguously knows that σ has to be disabled. Therefore, a decision can be chosen, and we remove these states from the pending set. Similarly, compute the set

$$\begin{aligned} E_{k+1}(\sigma) &:= E_k(\sigma) \cap \left(\bigcap_{i \in \mathcal{N}_\sigma} [D_k(\sigma)]_i \right) \\ &= E_k(\sigma) - \left[\bigcup_{i \in \mathcal{N}_\sigma} (U_k(\sigma) - [D_k(\sigma)]_i) \right] \end{aligned} \tag{6.2.2}$$

Notice that $D_{k+1}(\sigma)$ and $E_{k+1}(\sigma)$ are disjoint.

Then, let supervisor i issue the decision **disable**$_k$ at all states in the set $U_k(\sigma) - [E_k(\sigma)]_i$, and **enable**$_k$ at all states in the set $U_k(\sigma) - [D_k(\sigma)]_i$.

[2]The notation by Kumar and Takai does not contain prime symbols. We use prime symbols when referring to their languages to distinguish them from our sets of states.

Finally, we restrict the partial equivalence relations $\sim_i$ to $U_{k+1}(\sigma) = D_{k+1}(\sigma) \cup E_{k+1}(\sigma)$.

One important observation of this process is that for a state removed at step i, the minimal control decision issued at that state is unique, and can be denoted (cd, i). Furthermore, cd is exactly the desired fused decision.

The expressions (6.2.1) and (6.2.2) of the sets $D_{k+1}(\sigma)$ and $E_{k+1}(\sigma)$ correspond to the following languages in Kumar and Takai [KT07] and Takai and Kumar [TK08]:

$$\begin{aligned} D'_{k+1}(\sigma) &:= D'_k(\sigma) \cap \left(\bigcap_{i \in \mathcal{N}_\sigma} P_i^{-1} P_i(E'_k(\sigma)) \right) \\ E'_{k+1}(\sigma) &:= E'_k(\sigma) \cap \left(\bigcap_{i \in \mathcal{N}_\sigma} P_i^{-1} P_i(D'_k(\sigma)) \right) \end{aligned} \tag{6.3}$$

However, the correspondence is not exact: the computation of $D_k(\sigma)$ and $E_k(\sigma)$ eventually converge while the computation of $D'_k(\sigma)$ and $E'_k(\sigma)$ does not always converge. To not interrupt the current discussion, we continue illustrating the algorithm with the example and come back to the issue of convergence at the end of this section.

To illustrate the algorithm with the example, consider the equivalence class $\mathbf{A}_a$ in step 0, Fig. 6.5. Since all states in $\mathbf{A}_a$ (in this case, just a single state 5) are marked **E**, let the supervisor 1, from whose equivalence relation $\sim_1$ the equivalence class $\mathbf{A}_a$ was created, issue the control decision $\mathbf{enable}_0$ for the event c at all states in $\mathbf{A}_a$ (viz. just state 5). Then we remove the column for state 5. We use a similar strategy to remove the column for state 4 with supervisor 2 and equivalence class $\mathbf{B}_c$ being the analogues of supervisor 1 and $\mathbf{A}_a$ in the foregoing argument.

This brings us to step 1, as illustrated in Fig. 6.6. Now consider state 3, which has not been removed in step 0, since it was in the same equivalence class $\mathbf{B}_a$ as state 5, and they are marked differently. However with state 5 removed, the state 3 is no longer ambiguous to supervisor 2, since all states in the class $\mathbf{A}_a$ are marked identically as 3.

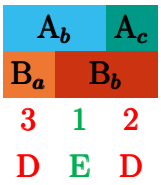

Figure 6.6: Step 1 of the algorithm on the example.

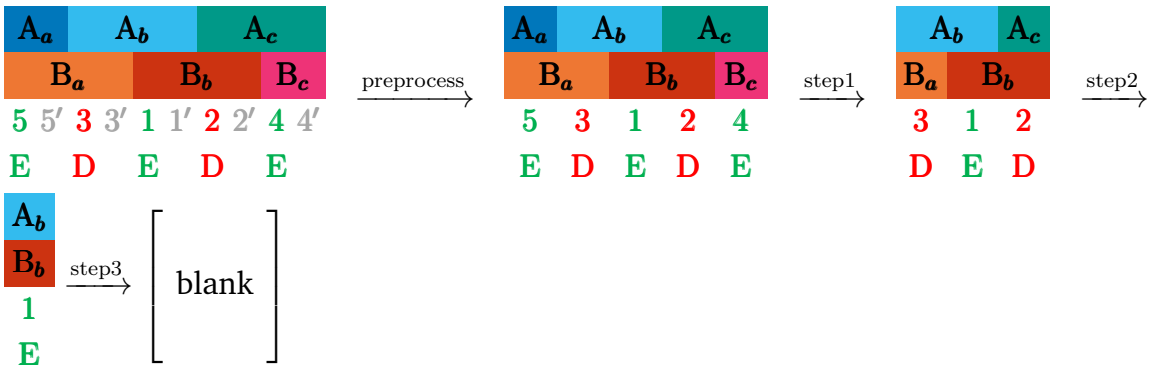

Figure 6.7: Complete trace of the algorithm running on the example.

From supervisor 2's point of view, it "knows" that supervisor 1 has the intention to enforce the desired control requirement at state 5, with the control decision **enable**$_0$, hence supervisor 2 no longer has to worry about state 5, and can focus on the remaining states in B_a. Since the states remaining in B_a are all marked D, hence supervisor 2 now can unambiguously realize that a **disable** decision is in order. But supervisor 2 could not have resolved the prior ambiguity if supervisor 2 did not know that the correct control decision would be enforced at state 5 by supervisor 1 with the control decision **enable**$_0$.

Hence in order not to step on the toes of supervisor 1, supervisor 2 should issue a **disable** command weaker than **enable**$_0$. While any number larger than 0 could be a candidate, supervisor 2 sees that just as it relied on supervisor 1's **enable**$_0$ decision, which is determined at step 0, its decision at the current step, step 1, might also be relied on in later steps. Hence, supervisor 2 should put its decisions under priority 1, and therefore would issue the control decision **disable**$_1$ at all states in B_a, including the state 5 removed in the previous step.

With this example, we conclude the following principle determining the appropriate priority of a control decision: to save us the problem of having to keep tracking the priorities of decisions determined at every step i, we can simply issue decisions with priority i, as the example has demonstrated that doing so is adequate.

Coming back to the example, with taking the global point of view, since state 5 is removed in step 0, a control decision with lower level of ambiguity is issued by supervisor 1 (namely, **enable**$_0$), hence supervisor 2's decision **disable**$_1$ is overridden and not effective. That decision of supervisor 2 is guaranteed to be effective, however, at state 3, since no lower-level decision will be issued at that state, as otherwise state 3 would have already been removed prior to step 1.

Continuing the algorithm on the example until termination, we obtain Fig. 6.7, and all states have been removed, meaning that at least a supervisor can make a non-

abstain decision at every state. For each $(\text{supervisor}, \text{state})$ pair to which we have not explicitly assigned a decision, assign **abstain**, so that the fused decisions remain as intended.

Neither our example (Fig. 6.2) nor the original example by Kumar and Takai [KT07, Fig. 1] illustrates a situation in which a supervisor needs to issue an **abstain** decision. As a supplement, we add, for the sake of conciseness, not a complete example with the plant and observable/controllable events specified, but only a snapshot of the algorithm, to illustrate when an **abstain** decision would be used.

Consider Fig. 6.8. Since both states 1 and 2 are eligible for removal at this point, after executing one step of the algorithm, the equivalence class A_a becomes empty. What makes the situation special is that the last removal making a class empty is not due to the unambiguity of the corresponding supervisor of that class. In the previous example (Fig. 6.4), the set B_a becomes empty when state 3 is removed, and the removal of that state is due to the fact that all remaining states in B_a (namely, state 3), are marked consistently (namely, by D). In contrast, the last removal resulting in the set A_a becoming empty in Fig. 6.8 removes states 1 and 2, however, they are marked differently to supervisor 1.

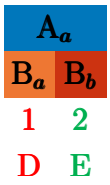

Figure 6.8: A situation where the control decision **abstain** is issued.

Since both states 1 and 2 in Fig. 6.8 are removed due to supervisor 2's confidence, this step of the algorithm assigns non-**abstain** decisions to supervisor 2 at those two states. On the other hand, since the set A_a is also effectively removed, we have to assign some decision at states in the set A_a (including not only states 1 and 2 but also states removed from A_a in the previous steps). By our previous argument, assigning to supervisor 1 either decision **enable** or **disable** with an index smaller than the number of the current step is an adequate choice, since either way supervisor 1's decision will have no effect on shaping the fused decisions. Hence strictly speaking we do not need a distinct **abstain** decision. Still, it is beneficial to use the **abstain** decision to distinguish such special cases from the cases where the regular **enable** and **disable** decisions are needed.

Therefore, we see, whenever the algorithm terminates, either all states have been removed, in which case the problem is solvable, and in fact a solution is produced; or some states are left but all remain ambiguous, in which case our algorithm

produces no solution for the given implementation of G.

At this point, we can informally summarize, that if the algorithm terminates with all states removed, then at every state, a unique minimal non-**abstain** decision is issued, and hence the synthesized supervision must — by construction — be admissible.

Now we turn back to and elaborate on the difference between our sets of states $D_k(\sigma)$, $E_k(\sigma)$ and the languages $D'_k(\sigma)$, $E'_k(\sigma)$ [KT07; TK08]. Since both $D_0(\sigma)$ and $E_0(\sigma)$ are finite, our computation eventually converges. On the other hand, with $D'_0(\sigma)$ and $E'_0(\sigma)$ potentially being infinite, the computation does not always converge. This difference is due to the fact that our algorithm does not exploit the ability to unfold cycles in the plant automaton, whereas when the algorithm of Kumar and Takai does [KT07; TK08], its computation may not converge in a finite number of steps. Recall that given a specific plant specification G, our algorithm constructs the automaton $G' = G \times P_1(G) \times \cdots \times P_n(G)$ and the computation of $D_k(\sigma)$ and $E_k(\sigma)$ can be thought as removing all strings leading into an eligible state of G' at once, by gradually restricting to subsets of the states of G', whereas the computation of $D'_k(\sigma)$ and $E'_k(\sigma)$ can remove fewer strings at once, and thus can be regarded as unfolding cycles in the plant during the computation.

Indeed, our algorithm could be augmented with the capability of unfolding cycles, by iteratively computing the automata $G^{k+1} = G^k \times P_1(G^k) \times \cdots \times P_n(G^k)$, where $G^0 = G$, until convergence (which does not always happen), and then proceed to the described computations of $D_k(\sigma)$ and $E_k(\sigma)$. We then face two questions. Does unfolding cycles in this way allow our algorithm to solve more problems? Furthermore, is it ever necessary to unfold cycles? The answer to these questions is unknown. Having the answer "No" to the second question would be ideal, and a sufficient condition would be if our algorithm is specification-independent, i.e., given two behaviourally equivalent plant specifications G and H (so that $L(G) = L(H)$), if our algorithm terminates with a solution for G, it also terminates with a solution for H. Moreover, if our algorithm is indeed specification-independent, then it is equivalent to the original algorithm of Kumar and Takai [KT07; TK08] (in the sense that when one gives a solution, the other also does), which means we can avoid diverging computations entirely. However, whether our algorithm is specification-independent is unknown.

6D Conclusions

This chapter presents a visualization of the synthesis of supervisors in the inference-based decentralized supervisory architecture [KT07; TK08]. The visualization provides an alternative interpretation to the number N in the "N-inferencing archi-

tecture". Instead of being the highest level of ambiguity one would like to permit when solving a decentralized supervisory problem, the number N denotes the highest level of ambiguity inherently present in the system, so that if one desires to solve the problem at all, one must permit at least N levels of ambiguities during the inference.

The visualization provides more accessible intuition of the nature of "inferencing". We are now able to discuss the supervision informally with phrases such as "supervisor 1 knows that supervisor 2 knows . . . ". This notion of reasoning about knowledge can be formalized in the same way as done in our earlier work [RR23].

References

[HMU06] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. 3rd ed. USA: Addison-Wesley Longman Publishing Co., Inc., 2006. [cit. on p. 88].

[KT05] R. Kumar and S. Takai. "Inference-based Ambiguity Management in Decentralized Decision-Making: Decentralized Control of Discrete Event Systems". In: *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005. DOI: 10.1109/CDC.2005.1582701. [cit. on p. 87].

[KT07] R. Kumar and S. Takai. "Inference-Based Ambiguity Management in Decentralized Decision-Making: Decentralized Control of Discrete Event Systems". In: *IEEE Transactions on Automatic Control* 52.10 (Oct. 2007), pp. 1783–1794. DOI: 10.1109/TAC.2007.906158. [cit. on p. 85. 86. 87. 88. 90. 91. 92. 94. 95].

[PKK97] J. H. Prosser, M. Kam, and H. G. Kwatny. "Decision fusion and supervisor synthesis in decentralized discrete-event systems". In: *Proceedings of the American Control Conference*. IEEE, 1997. DOI: 10.1109/ACC.1997.608978. [cit. on p. 85].

[RR23] K. Ritsuka and K. Rudie. *Do What You Know: Coupling Knowledge with Action in Discrete-Event Systems*. Submitted for publication. 2023. [cit. on p.].

[RW92] K. Rudie and W. M. Wonham. "Think globally, act locally: decentralized supervisory control". In: *IEEE Transactions on Automatic Control* 37.11 (1992), pp. 1692–1708. DOI: 10.1109/9.173140. [cit. on p. 85].

[TK08] Shigemasa Takai and Ratnesh Kumar. "Synthesis of Inference-Based Decentralized Control for Discrete Event Systems". In: *IEEE Transactions on Automatic Control* 53.2 (Mar. 2008), pp. 522–534. DOI: 10.1109/tac.2007.915171. [cit. on p. 86. 87. 91. 92. 95].

[YL02] T.-S. Yoo and Stéphane Lafortune. “A General Architecture for Decentralized Supervisory Control of Discrete-Event Systems”. In: *Discrete Event Dynamic Systems* 12.3 (2002), pp. 335–377. DOI: 10.1023/a:1015625600613. [cit. on p. 85].

7 Equivalence of Decentralized Observation, Diagnosis, and Control Problems in Discrete-event Systems

This chapter demonstrates an equivalence between observation problems, control problems (with partial observation), and diagnosis problems of decentralized discrete-event systems, namely, the three classes of problems are Turing equivalent, as one class Turing reduces to another.

The equivalence allows decomposition of a control problem into a collection of simpler control sub-problems, which are each equivalent to an observation problem; and similarly allows converting a diagnosis problem to a formally simpler observation problem. Since observation problems in their most general formulation have been shown to be undecidable in previous work, the equivalence produced here demonstrates that control problems are also undecidable; whereas the undecidability of diagnosis problems is a known result.

7A Introduction

Most research in discrete-event systems (DES) falls into two categories: those concerning closed-loop systems such as control problems, and those concerning open-loop systems, such as observation problems and diagnosis problems.

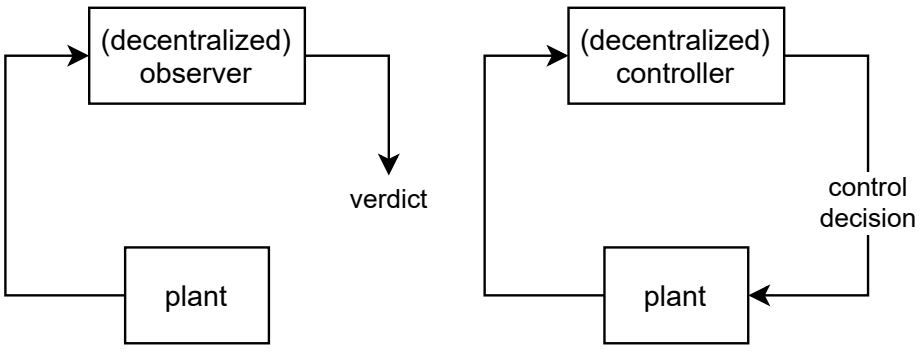

Figure 7.1: Left: open-loop systems. Right: closed-loop systems. Adapted from [Tri].

For a discrete-event system plant, a closed-loop system is formed by imposing supervisory control over the plant. A control problem asks for a supervisory control policy so that the closed-loop system meets some prescribed properties. The scheme of control problems is illustrated in Fig. 7.1.

Studies of control problems began with the seminal work of Ramadge and Wonham [RW87]. Partial observations [LW88] and decentralized supervision [Cie+88; RW92] were introduced in subsequent studies. Cieslak et al. [Cie+88] and Rudie and Wonham [RW92] initially introduced decentralized supervision under a constraint of available local control decisions and how overall control decisions are fused from the local ones. That constraint has been gradually relaxed [PKK97; YL02; YL04; KT05; CK11] over the past few decades.

On the other hand, open-loop systems take different forms. A concrete example is diagnosis problems [Sam+95; DLT00; ST02; QK06; WYL07] that seek distinguishing strings contain "faulty" events within a bounded delay of the occurrence of the faulty events. On the other hand, a more abstract example is observation problems that seek distinguishing strings from a prescribed collection. The earliest formalization of observation problems, as known to the author, is by Tripakis [Tri04]. The scheme of open-loop systems is illustrated in Fig. 7.1.

The three classes of problems, control, diagnosis, and observation, seem to be unrelated. Control problems concern closed-loop behaviour, whereas diagnosis problems allow a delay for the correct verdicts to be made, but observation problems do not. Therefore, the three classes of problems are usually studied separately.

However, results for one of the classes of problems have often been adopted to one of the other classes of problems. This suggests that there is a mutual connection between the three classes of problems. This document is intended to establish such connection as an equivalence between the three classes of problems.

7B Observation Problem

An observation problem seeks to distinguish strings in a set K, where $K \subseteq L$, from strings in $L - K$. Formally, an observation problem is specified as follows. Given alphabet Σ and subalphabets $\Sigma_{i,o} \subseteq \Sigma$ called the observed alphabets, natural projections $P_i \colon \Sigma^* \to \Sigma_{i,o}$, for agents $i \in \mathcal{N} = \{1, \ldots, n\}$, and given languages $K \subseteq L \subseteq \Sigma^*$, the observation problem is to construct observers f_i and a fusion rule

f, such that

$$\begin{aligned} &\forall\, s \in L\,. \\ &\quad s \in K \quad\;\; \Rightarrow f(f_1 P_1(s), \ldots, f_n P_n(s)) = 1 \\ &\wedge\, s \in L - K \Rightarrow f(f_1 P_1(s), \ldots, f_n P_n(s)) = 0 \end{aligned} \tag{7.1}$$

An instance of the observation problem, Obs, is denoted by $\mathrm{O}(L, K, \{\, \Sigma_{i,o} \,\}_{i \in \mathcal{N}})$ or more simply, $\mathrm{O}(L, K, \Sigma_{i,o})$.

If the fusion rule f is given as part of the problem, then the instance is denoted by $\mathrm{O}(f, L, K, \Sigma_{i,o})$. Such problems are instance of the f-observation problem, or f-Obs.

Solvability of observation problems is known to be undecidable [Tri04].

We will show that diagnosis problems and control problems are both equivalent to observation problems.

7C Diagnosis Problem

The diagnosis problems were first studied in the centralized case by Sampath et al. [Sam+95], and extended to the decentralized cases [DLT00; ST02; QK06; WYL07].

A diagnosis problem seeks to identify strings containing special events, known as "faulty events", within a bounded delay of time. Formally, a diagnosis problem is specified as follows. Given alphabet Σ and subalphabets $\Sigma_{i,o} \subseteq \Sigma$ called the observed alphabets, natural projections $P_i \colon \Sigma^* \to \Sigma_{i,o}$, for agents $i \in \mathcal{N} = \{\, 1, \ldots, n \,\}$. For a fault alphabet $\Sigma_f \subseteq \Sigma_{uo} = \Sigma - \bigcup_i \Sigma_{io}$, a language L, say a string $s \in L$ is positive (faulty) if s contains at least one symbol from Σ_f, and otherwise is negative. We may assume that there is a single fault event σ_f as this assumption is inconsequential to the hardness of the problem.

For a faulty string s, if $s = \pi\sigma_f\tau$ for some strings π and τ, where $|\tau| \geq m$, we say that s is faulty for at least m steps. In other words, a string st is faulty for at least m steps if s is faulty and $|t| \geq m$.

Then the diagnosis problem is, given an upper bound of delay as an integer m,

construct observers f_i and a fusion rule f, such that

$$\begin{aligned} &\forall\, s \in L\,. \\ &\quad s \text{ is positive for at least } m \text{ steps} \Rightarrow f(f_1P_1(s), \ldots, f_nP_n(s)) = 1 \\ &\quad \wedge\, s \text{ is negative} \qquad\qquad\quad \Rightarrow f(f_1P_1(s), \ldots, f_nP_n(s)) = 0 \end{aligned} \tag{7.2}$$

That is, faulty strings are diagnosed after at most m steps of the fault.

The problem statement above is a simplification: A subtlety in the problem statement is that there may exist a faulty string $s \in L$ that is positive for less than m steps but has no extension in L which is positive for at least m steps. Since we nonetheless want to diagnose such faulty strings, the phrase "s is positive for at least m steps" should be augmented to include such strings.

An instance of diagnosis problem, Dx, is denoted by $\mathrm{D}(L, \{\,\Sigma_{i,o}\,\}_{i\in\mathcal{N}}, \sigma_f, m)$ or more simply, $\mathrm{D}(L, \Sigma_{i,o}, \sigma_f, m)$.

If the fusion rule f is given as part of the problem, then the instance is denoted by $\mathrm{D}(f, L, \Sigma_{i,o}, \sigma_f, m)$. Such problems are instance of the f-diagnosis problem, or f-Dx.

7C1 Equivalence of Diagnosis Problems and Observation Problems

Theorem 7C1.1
Given a fusion rule f, the class of f-diagnosis problems — f-Dx — reduces to the class of f-observation problems — f-Obs.

Proof. For a given f-diagnosis problem $\mathrm{D}(f, L, \Sigma_{i,o}, \sigma_f, m)$, construct the following f-observation problem $\mathrm{O}(f, L, K, \Sigma_{i,o})$, where $K = \{\, s \in L \mid s \text{ is positive for at least } m \text{ steps} \,\}$.

By construction, (7.1) and (7.2) coincide. □

Theorem 7C1.2
Given a fusion rule f, f-Obs reduces to f-Dx.

Proof. For a given f-observation problem $\mathrm{O}(f, L, K, \Sigma_{i,o})$, construct the following f-diagnosis problem $\mathrm{D}(f, L', \Sigma_{i,o}, \sigma_f, 0)$, where $L' = (L - K) \cup \{\, s\sigma_f \mid s \in K \,\}$ and where we have chosen $m = 0$.

Notice that negative strings in L' are exactly strings in $L - K$, and a string in L' that is positive (for at least 0 steps) — i.e., one in $\{\, s\sigma_f \mid s \in K \,\}$ — uniquely corresponds

to a string $s \in K$ and satisfies $s' = s\sigma_f$, hence $P_i(s) = P_i(s'\sigma_f^n) = P_i(s')$. Thus, by construction, (7.1) and (7.2) coincide. □

Theorem 7C1.3
The classes of problems f-Obs and f-Dx are equivalent. Moreover, Obs and Dx are equivalent.

Proof. By Thms. 7C1.1 and 7C1.2. □

It is known that solvability of diagnosis problems is undecidable [ST02]. The reduction Thm. 7C1.1 offers an alternative route to proving that undecidability. Namely, we showed that observation problems reduce to diagnosis problems, and from Tripakis [Tri04] we know that observation problems are undecidable.

7D Control Problem

Recall that the control problem is to construct controllers f_i^σ and fusion rules f^σ, for each event $\sigma \in \Sigma_c$, such that

$$\begin{aligned} &\forall\, s \in K\,. \\ &\quad s\sigma \in K \qquad \Rightarrow f^\sigma(f_1^\sigma P_1(s), \ldots, f_{n_\sigma}^\sigma P_{n_\sigma}(s)) = 1 \\ &\wedge\, s\sigma \in L - K \Rightarrow f^\sigma(f_1^\sigma P_1(s), \ldots, f_{n_\sigma}^\sigma P_{n_\sigma}(s)) = 0 \end{aligned} \tag{7.3}$$

To avoid trivial unsolvable instances, we assume that an instance is always controllable.

An instance of control problem, Con, is denoted by $\mathrm{C}(L, K, \{\,\Sigma_{i,o}\,\}_{i\in\mathcal{N}}, \{\,\Sigma_{i,c}\,\}_{i\in\mathcal{N}})$, or more simply, $\mathrm{C}(L, K, \Sigma_{i,o}, \Sigma_{i,c})$.

7D1 Equivalence of Control Problems and Observation Problems

We first revise the problem specification of the control problems.

Theorem 7D1.1
Define the following two languages

$$\begin{aligned} L_\sigma &= \{\, s \in K \mid s\sigma \in L \,\} \\ K_\sigma &= \{\, s \in K \mid s\sigma \in K \,\}. \end{aligned} \tag{7.4}$$

Then (7.3) is equivalent to

$$\begin{aligned} &\forall\, \sigma \in \Sigma_c, s \in L_\sigma\,. \\ &\quad s \in K_\sigma \quad\;\; \Rightarrow f(f_1^\sigma P_1(s), \ldots, f_{n_\sigma}^\sigma P_{n_\sigma}(s)) = 1 \\ &\wedge\, s \in L_\sigma - K_\sigma \Rightarrow f(f_1^\sigma P_1(s), \ldots, f_{n_\sigma}^\sigma P_{n_\sigma}(s)) = 0. \end{aligned} \tag{7.5}$$

Proof. By definition, for all $s \in L_\sigma$,

$$s\sigma \in L - K \Leftrightarrow s \in L_\sigma - K_\sigma.$$

This concludes the proof. □

Theorem 7D1.2
The classes of problems Con reduces to Obs

Proof. For a given control problem $\mathrm{C}(L, K, \Sigma_{i,o}, \Sigma_{i,c})$, construct the following observation problems

$$\{\, \mathrm{O}(L_\sigma, K_\sigma, \{\, \Sigma_{i,o} \,\}_{i \in \mathcal{N}_\sigma}) \,\}_{\sigma \in \Sigma_c}.$$

By construction, (7.1) and (7.5) coincide. □

From the proof we can see that it is appropriate to decompose a control problem into a collection of individual (control) sub-problems, each one dealing with a specific event.

Theorem 7D1.3
The class of problems Obs reduces to Con′.

Proof. For a given observation problem $\mathrm{O}(L, K, \Sigma_{i,o})$, construct a control problem as follows. First add to the alphabet a distinguished letter γ, and let $\Sigma_{i,c} = \{\, \gamma \,\}$ for all $i \in \mathcal{N}$. Henceforth, let $\mathrm{pr}(M)$ stands for the prefix-closure of language M. Now let

$$\begin{aligned} L' &:= \mathrm{pr}(L\gamma) \\ &= \mathrm{pr}(L) \cup L\gamma \\ K' &:= \mathrm{pr}(K\gamma \cup L) \\ &= \mathrm{pr}(K) \cup K\gamma \cup \mathrm{pr}(L). \end{aligned}$$

The control problem is then

$$\mathrm{C}(L', K', \Sigma_{i,o}, \Sigma_{i,c}).$$

It should be verified that the control problem is well-posed. First, it is clear that L' and K' are indeed prefix-closed. To verify controllability, let Σ be the alphabet of L, and hence $\Sigma_{uc} = \Sigma$. Then, for any $\sigma \in \Sigma_{uc}$ and string $s \in K'$, suppose that $s\sigma \in L'$. If $s\sigma \in \mathrm{pr}(L)$, $s\sigma \in K'$ as desired. If $s\sigma \in L\gamma$, then $\sigma = \gamma$, which contradicts the fact that γ is a controllable event.

Now compute the languages in (7.4). First, we have

$$\begin{aligned} L'_\gamma &= \{\, s \in K' \mid s\gamma \in L' \,\} \\ &= \{\, s \in K' \mid s\gamma \in \mathrm{pr}(L) \cup L\gamma \,\} \\ &= \{\, s \in K' \mid s \in L \,\} \\ &= L \end{aligned}$$

where the third line is due to γ being a distinguished letter that is not in L, and consequently not in $\mathrm{pr}(L)$; the fourth line is due to the facts that $L \subseteq K\gamma \cup L \subseteq \mathrm{pr}(K\gamma \cup L) = K'$. Similarly, we have

$$\begin{aligned} K'_\gamma &= \{\, s \in K' \mid s\gamma \in K' \,\} \\ &= \{\, s \in K' \mid s\gamma \in \mathrm{pr}(K) \cup K\gamma \cup \mathrm{pr}(L) \,\} \\ &= \{\, s \in K' \mid s \in K \,\} \\ &= K \end{aligned}$$

where the third line is due to γ being a distinguished letter that is not in L, and also K being a subset of L. The last line is due to $K \subseteq \mathrm{pr}(K) \subseteq K'$. Then (7.5) coincides with (7.1). □

Theorem 7D1.4
The classes of problems Obs and Con are equivalent.

Proof. By Thms. 7D1.2 and 7D1.3. □

The approach of Lin and Wonham [LW88] in dealing with centralized control problems under partial observation can be interpreted as a special case of the reduction of control problems to observation problems (i.e., $\mathrm{CON} \leq_T \mathrm{OBS}$).

Corollary 7D1.5
Solvability of control problems are undecidable in general.

Proof. We have just shown that the observation problems reduces to control problems, whereas Tripakis demonstrated that solvability of observation problems is undecidable [Tri04]. □

Corollary 7D1.5 only states the undecidability of control problems when *no restriction* is placed on the fusion rule. However, in special cases when the fusion rule is restricted, such as for the architecture given by Cieslak et al. [Cie+88] and Rudie and Wonham [RW92], solvability can still be decided [RW95].

References

[Cie+88] R. Cieslak, C. Desclaux, A. S. Fawaz, and P. Varaiya. "Supervisory control of discrete-event processes with partial observations". In: *IEEE Transactions on Automatic Control* 33.3 (Mar. 1988), pp. 249–260. DOI: 10.1109/9.402. [cit. on p. 100].

[CK11] H. Chakib and A. Khoumsi. "Multi-Decision Supervisory Control: Parallel Decentralized Architectures Cooperating for Controlling Discrete Event Systems". In: *IEEE Transactions on Automatic Control* 56.11 (Nov. 2011), pp. 2608–2622. DOI: 10.1109/tac.2011.2128730. [cit. on p. 100].

[DLT00] Rami Debouk, Stéphane Lafortune, and Demosthenis Teneketzis. In: *Discrete Event Dynamic Systems* 10.1/2 (2000), pp. 33–86. DOI: 10.1023/a:1008335115538. [cit. on p. 100. 101].

[KT05] R. Kumar and S. Takai. "Inference-based Ambiguity Management in Decentralized Decision-Making: Decentralized Control of Discrete Event Systems". In: *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005. DOI: 10.1109/CDC.2005.1582701. [cit. on p. 100].

[LW88] F. Lin and W. M. Wonham. "On observability of discrete-event systems". In: *Information Sciences* 44.3 (Apr. 1988), pp. 173–198. DOI: 10.1016/0020-0255(88)90001-1. [cit. on p. 100. 105].

[PKK97] J. H. Prosser, M. Kam, and H. G. Kwatny. "Decision fusion and supervisor synthesis in decentralized discrete-event systems". In: *Proceedings of the American Control Conference*. IEEE, 1997. DOI: 10.1109/ACC.1997.608978. [cit. on p. 100].

[QK06] Wenbin Qiu and R. Kumar. "Decentralized failure diagnosis of discrete event systems". In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 36.2 (Mar. 2006), pp. 384–395. DOI: 10.1109/tsmca.2005.853503. [cit. on p. 100. 101].

[RW87] P. J. Ramadge and W. M. Wonham. "Supervisory Control of a Class of Discrete Event Processes". In: *SIAM Journal on Control and Optimization* 25.1 (Jan. 1987), pp. 206–230. DOI: 10.1137/0325013. [cit. on p. 100].

[RW92] K. Rudie and W. M. Wonham. "Think globally, act locally: decentralized supervisory control". In: *IEEE Transactions on Automatic Control* 37.11 (1992), pp. 1692–1708. DOI: 10.1109/9.173140. [cit. on p. 100. 105].

[RW95] K. Rudie and J. C. Willems. "The computational complexity of decentralized discrete-event control problems". In: *IEEE Transactions on Automatic Control* 40.7 (July 1995), pp. 1313–1319. DOI: 10.1109/9.400469. [cit. on p. 105].

[Sam+95] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. "Diagnosability of discrete-event systems". In: *IEEE Transactions on Automatic Control* 40.9 (1995), pp. 1555–1575. DOI: 10.1109/9.412626. [cit. on p. 100. 101].

[ST02] R. Sengupta and S. Tripakis. "Decentralized diagnosability of regular languages is undecidable". In: *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.* IEEE, 2002. DOI: 10.1109/cdc.2002.1184531. [cit. on p. 100. 101. 103].

[Tri] Stavros Tripakis. *Distributed Synthesis*. Virtual Lightning Tutorial Series on Discrete Event Systems 2021. [Accessed May 30, 2022]. URL: https://owncloud.mpi-sws.org/index.php/s/Q3RKiiQ67GPJJw4/download?path=%5C%2F%5C&files=04_STripakis_Slides.pdf. [cit. on p. 99].

[Tri04] Stavros Tripakis. "Undecidable problems of decentralized observation and control on regular languages". In: *Information Processing Letters* 90.1 (Apr. 2004), pp. 21–28. DOI: 10.1016/j.ipl.2004.01.004. [cit. on p. 100. 101. 103. 105].

[WYL07] Yin Wang, Tae-Sic Yoo, and Stéphane Lafortune. "Diagnosis of Discrete Event Systems Using Decentralized Architectures". In: *Discrete Event Dynamic Systems* 17.2 (Jan. 2007), pp. 233–263. DOI: 10.1007/s10626-006-0006-8. [cit. on p. 100. 101].

[YL02] T.-S. Yoo and Stéphane Lafortune. "A General Architecture for Decentralized Supervisory Control of Discrete-Event Systems". In: *Discrete Event Dynamic Systems* 12.3 (2002), pp. 335–377. DOI: 10.1023/a:1015625600613. [cit. on p. 100].

[YL04] T.-S. Yoo and S. Lafortune. "Decentralized Supervisory Control With Conditional Decisions: Supervisor Existence". In: *IEEE Transactions on Automatic Control* 49.11 (Nov. 2004), pp. 1886–1904. DOI: 10.1109/tac.2004.837595. [cit. on p. 100].

8 A Uniform Treatment of Architectures in Decentralized Discrete-Event Systems

Solutions to decentralized discrete-event systems problems are characterized by the way local decisions are fused to yield a global decision. A fusion rule is colloquially called an *architecture*. This chapter provides a uniform treatment of architectures in decentralized discrete-event systems. Current approaches neither provide a direct way to determine problem solvability conditions under one architecture, nor a way to compare existing architectures. Determining whether a new architecture is more general than an existing known architecture relies on producing examples *ad hoc* and on individual inspiration that puts the conditions for solvability in each architecture into some form that admits comparison. From these research efforts, a method based on morphisms between graphs has been extracted to yield a uniform approach to decentralized discrete-event system architectures and their attendant fusion rules. This treatment provides an easy and direct way to compare the fusion rules — and hence to compare the strength or generality of the corresponding architectures.

8A Introduction

Many solutions of varying levels of strengths exist for decentralized supervisory control problems [Cie+88; RW92; PKK97; YL02; YL04; KT05; CK11; RM13; RR22c]. The solutions are given under various architectures, whereby an architecture is characterized by the way decentralized decisions are combined according to some mathematical function called a *fusion rule*. The breadth of an architecture is essentially represented by the class of problems solvable under that architecture. One architecture is more general than the other if the class of problems solvable under the former is greater than that solvable under the latter. Taking control problems as an example, since a common goal of control problems is to produce solutions that generate as large a set of behaviours as possible (or that are minimally restrictive), DES research in decentralized systems has as one of its aims to investigate novel architectures that are more general than existing ones.

In a few simpler settings what distinguishes one architecture from another can be interpreted as how conflicting decisions are resolved. In a broader set of settings

epistemic interpretations are given [RR00; RR07; RR21; RR22c]. However there does not appear to be a uniform interpretation of the differences.

The traditional approach to investigating new, perhaps more general, architectures always proceeds in a similar manner. First, a novel architecture is proposed, often by augmenting or modifying an existing one. Then, a characterization for decentralized problems to have solutions under the novel architecture is given. If it is intended to demonstrate that the novel architecture is possibly more general to some existing architecture, one shows that the problem solvability characterization of the novel architecture logically entails that of the existing architecture. To demonstrate that the new architecture is not superfluous, one provides an example that shows that the class of problems solvable under the new architecture is strictly larger than that solvable under the existing architecture. Sometimes the novel architecture turns out to be incomparable to existing ones. An example of this would be the disjunctive architecture [PKK97], which was shown by Yoo and Lafortune [YL02] to be incomparable to the prior conjunctive architecture [RW92] first used in decentralized DES control problems.

The traditional approach is complicated. First, the problem solvability characterization is usually presented with little insight provided to indicate how it was derived. The same remark applies to the presentation of the example problem. Moreover, the approach is indirect and can be laborious.

This chapter gives a uniform interpretation for decentralized architectures, and proposes a uniform and simple approach that lends itself easily to direct comparison of decentralized architectures.

8B Decentralized Problems

For simplicity of discussion, we shall concern ourselves with the following kind of decentralized observation problems, since it has been shown that the observation problems and the seemingly more complicated control problems are in fact equivalent [RR22a]. Hence focusing on just the observation problems simplifies discussions.

Problem 8B.1 (Decentralized Discrete-Event Systems Observation Problem) Given alphabet Σ and subalphabets $\Sigma_{i,o} \subseteq \Sigma$ called the observed alphabets, natural projections $P_i \colon \Sigma^* \to \Sigma_{i,o}^*$, for agents $i \in \mathcal{N} = \{1, \ldots, n\}$, and given languages $K \subseteq L \subseteq \Sigma^*$, the observation problem is to construct *local decision functions* f_i (also

informally called *observers/agents*) and a *fusion rule* f, such that

$$\begin{aligned}\forall\, s \in L\,.&\\ s \in K \quad &\Rightarrow f(f_1P_1(s), \ldots, f_nP_n(s)) = 1\\ \wedge\, s \in L - K &\Rightarrow f(f_1P_1(s), \ldots, f_nP_n(s)) = 0.\end{aligned}$$

Informally, the quintessential decentralized observation problem is about producing local decisions that ensure that when those decisions are fused, strings that are in some prescribed subset K of L can be distinguished from strings that are not in K.

Throughout this chapter, we will need the following notations to handle functions on n-tuples.

Definition 8B.2
For n functions $g_1, \ldots, g_n \colon A \to B$, define the *broadcasting application* (as broadcasting $x \in A$ to each g_i) as

$$\begin{aligned}\langle g_1, \ldots, g_n\rangle \colon A &\to B^n\\ = \qquad\qquad x &\mapsto (g_1(x\ ), \ldots, g_n(x\ ))\end{aligned}$$

and the *element-wise application* (of x_i and g_i) as

$$\begin{aligned}(g_1, \ldots, g_n) \colon A^n &\to B^n\\ = \ (x_1, \ldots, x_n) &\mapsto (g_1(x_1), \ldots, g_n(x_n))\end{aligned}$$

For compactness, we may write $\langle g_1, \ldots, g_n\rangle(x)$ as $\langle g_i\rangle x$ and $(g_1, \ldots, g_n)(x)$ as $(g_i)x$.

Since we frequently need to consider when two strings are "indistinguishable", we make use of the following definitions.

Definition 8B.3 (Kernel of a Function)
The kernel of a function f, written $\ker f$, is an equivalence relation over the domain of f, such that $(x, y) \in \ker f$ wherever $f(x) = f(y)$, i.e., $\ker f$ relates elements indistinguishable by f.

Definition 8B.4 (Partition of a Set)
We say that a set of sets π_i is a partition of a set S whenever $\bigcup \pi_i = S$, i.e., π_i covers S, and $\bigcap \pi_i = \varnothing$, i.e., sets in π_i are mutually disjoint. Consequently, every element of S is in exactly one set of π_i.

Clearly, kernels of functions over a set S correspond one-to-one with partitions of S. Henceforth, we will identify kernels and partitions.

Definition 8B.5
An equivalence relation R refines an equivalence relation R, written $R \leq S$, whenever $(x, y) \in R \Rightarrow (x, y) \in S$.

Speaking in terms of partitions, a partition π_i refines a partition τ_j, written $\pi_i \leq \tau_j$ if whenever $x, y \in \pi_i$ for some i, there is some j such that $x, y \in \tau_j$. Recall that sets in π_i are disjoint, and similarly for τ_j. Hence, in other words, $\pi_i \leq \tau_j$ iff for each i, $\pi_i \subseteq \tau_j$ for some j.

For two strings s_1 and s_2 in L such that $P_i(s_1) = P_i(s_2)$, necessarily $f_i P_i(s_1) = f_i P_i(s_2)$ for all $i \in \mathcal{N}$. We call this fact *feasibility*. Feasibility can be described in terms of refinement of function kernels: $\ker(P_1, \ldots, P_n) \leq \ker(f_1 P_1, \ldots, f_n P_n)$, i.e., the first kernel refines the second.

8C A Uniform Approach to Derive Problem Solvability Characterization from a Given Fusion Rule

We aim at deriving a uniform approach to compare decentralized architectures directly. We do so by first giving a uniform way to derive problem solvability characterization, from which we will then derive our direct approach for comparing architectures. In our approach, we describe a decentralized problem as an *observation graph* and a solution based on a fusion rule f as a *decision graph*. Then the problem can be expressed as finding a way of folding the observation graph into the decision graph, which will be formally expressed in terms of graph morphism. Then the problem solvability condition can be thought as determining if the decision graph has the capacity to embed the observation graph.

Definition 8C.1 (Observation Graph)
For each subset $N \subseteq \mathcal{N}$, define the symmetric relations $\sim_N$ on L, so that $s \sim_N t$ if and only if the two tuples $\langle P_i \rangle s = (P_1(s), \ldots, P_n(s))$ and $\langle P_i \rangle t = (P_1(t), \ldots, P_n(t))$ differ by exactly the components indexed by N. Formally,

$$\sim_N = \{\, (s, t) \in L \times L \mid P_i(s) \neq P_i(t) \Leftrightarrow i \in N \,\}.$$

The relations $\sim_N$ reflect that exactly the agents in N have changed observation. We may consider L and $\sim_N$ to form an undirected graph $(L, \sim)$, which we will call an *observation graph*. We denote the observation graph also with L. We consider the graph as a complete graph where edges are coloured by subsets of $\mathcal{N}$. We also colour a node s by the truth value of $s \in K$.

The equivalence relation $\sim_\varnothing$ is exactly the kernel $\ker\langle P_i\rangle$.

We provide an example of the observation graph.

Example 8C.2
Consider the observation problem with two agents where $L = \{a, b, ab, bb\}$, $K = b$, $\Sigma_{1,o} = \{a\}$ and $\Sigma_{2,o} = \{b\}$. This example is derived from [RW92, Fig. 1]. We depict the observation graph as in Fig. 8.1. Each node is labelled by strings $s \in L$, $P_1(s)$, and $P_2(s)$, vertically stacked in that order. Vertical/blue/dotted lines denote relation $\sim_1$; horizontal/red/dashed lines denote relation $\sim_2$; and diagonal/purple/solid lines denote relation $\sim_{1,2}$. The relation $\sim_\varnothing$ happens to be the identity relation for this example and is omitted from the graph. Red/singly-bordered nodes indicate strings in $L - K$, and green/doubly-bordered nodes indicate string in K.

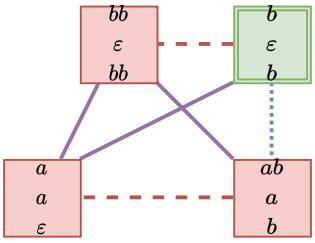

Figure 8.1: Observation graph for the observation problem in Example 8C.2.

The observation graphs are essentially a more compact alternative to the Kripke structures used in the works employing epistemic logic interpretations for decentralized problems [RR00; RR07; RR21; RR22c]. The relations $\sim_N$ capture various notions of group knowledge, e.g., what is expressed by distributed knowledge and by the "everybody knows" operators in epistemic logic [Fag+04].

Without loss of generality, suppose that the local decision functions f_i all have codomain D, for otherwise we can simply take $D = \bigcup_{i\in\mathcal{N}} \mathrm{cdm}(f_i)$, where $\mathrm{cdm}(f_i)$ stands for the codomain of f_i. Hence, the domain of f is a subset of $D \times \cdots \times D$ (n times).

Suppose that f is only defined over a certain collection $\mathcal{D}$ of combinations of local decisions$(d_1, \ldots, d_n) \in D \times \cdots \times D$, i.e., $\mathcal{D} = \mathrm{dom}(f) \subseteq D \times \cdots \times D$. The size of $\mathcal{D}$ roughly reflects the capacity of f, so that if $|\mathcal{D}| = 1$, f is a constant 1 or 0, and if $\mathcal{D}$ is large enough for a problem at hand, f is virtually unconstrained.

The size of $\mathcal{D}$ alone does not fully capture the capacity of the fusion rule. What we need additionally is the following.

Definition 8C.3 (Decision Graph)
For each subset $N \subseteq \mathcal{N}$, define symmetric relations $\sim_N$ on $\mathcal{D}$, so that $(d_1, \ldots, d_n) \sim_N (d'_1, \ldots, d'_n)$ exactly when the two tuples differ by exactly the components indexed by N. Formally,

$$\sim_N = \{\, ((d_1, \ldots, d_n), (d'_1, \ldots, d'_n)) \in \mathcal{D} \times \mathcal{D} \mid d_i \neq d'_i \Leftrightarrow i \in N \,\}$$

The relations $\sim_N$ reflect that exactly the agents in N have changed their decisions due to their change of observation. We may consider $\mathcal{D}$ and $\sim_N$ to form an undirected graph $(\mathcal{D}, \sim)$, which we will call a *decision graph*. We denote the decision graph also with $\mathcal{D}$. We consider the graph as a complete graph where edges are coloured by subsets of $\mathcal{N}$. We also colour nodes by the values of f at the nodes (0 or 1).

We provide an example of the decision graph.

Example 8C.4
The traditional way of describing the conjunctive architecture is by taking $D = \{\, 1, 0 \,\}$ and $f = \wedge$ [RW92]. For a problem with 2 agents, the decision graph can be depicted as in Fig. 8.2. Similar to how we depicted the observation graph in Example 8C.2, vertical/blue/dotted lines denote relation $\sim_1$; horizontal/red/dashed lines denote relation $\sim_2$; and diagonal/purple/solid lines denote relation $\sim_{1,2}$. The relation $\sim_\varnothing$ is the identity relation and is omitted from the graph. Red/singly-bordered nodes indicate fused decision being 0 and green/doubly-bordered nodes indicate fused decision being 1.

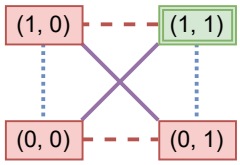

Figure 8.2: Decision graph for the conjunctive architecture.

We can regard the decision graph as reflecting the capacity of the architecture. The notion of capacity arises from the fact that solving an observation problem is essentially finding a way to fold the observation graph into the decision graph. We proceed to describe such a folding formally as a graph morphism.

The local decision functions f_i can be seen as a mapping from $P_i(L)$ to $\mathcal{D}$, subject to the following requirement. For any strings s and s' in L, let N be the (unique) set such that

$$s \sim_N s'.$$

If

$$\begin{aligned}(P_1(s)\,,\ldots,P_n(s)\,) &\xmapsto{(f_i)} (\,f_1P_1(s),\ldots,f_nP_n(s)\,)\\ &= (\quad d_1 \quad,\ldots,\quad d_n \quad)\\ (P_1(s'),\ldots,P_n(s')) &\xmapsto{(f_i)} (f_1P_1(s'),\ldots,f_nP_n(s'))\\ &= (\quad d'_1 \quad,\ldots,\quad d'_n \quad),\end{aligned}$$

then, with letting N' be the set such that $(d_1,\ldots,d_n) \sim_{N'} (d'_1,\ldots,d'_n)$, the mapping $g = \langle f_iP_i\rangle = s \mapsto (f_1P_1(s),\ldots,f_nP_n(s))$ must satisfy the following graph morphism conditions GM:

GM-1: Node-Colour Preserving

> The mapping g preserves node colouring, that is, g achieves the desired fused decision.

GM-2: Edge-Colour Intensive

> $N \supseteq N'$, i.e., only agents with changed observation can change decisions, though they do not necessarily have to. In other words, the mapping g may drop some edge colours, but may not add any. This property captures feasibility.

Conversely, a morphism satisfying the two conditions above gives a solution to the problem.

We capture the foregoing in the following theorem.

Theorem 8C.5
An observation problem is solvable in a given architecture if and only if there exists a morphism from the observation graph to the decision graph (representing the architecture's fusion rule) satisfying the morphism conditions GM.

Proof. ($\Rightarrow$): By the discussion preceding the theorem, g satisfies GM, as that is how the definition of GM was motivated.

($\Leftarrow$): Suppose that there is a morphism g satisfying the morphism conditions GM. Then a solution can be constructed as follows. For each string $s \in L$, let $(d_1,\ldots,d_n) = g(s)$, and let $f_iP_i(s) = d_i$ for $i \in \mathcal{N}$. Since g is edge-colour intensive (GM-2), the functions f_i are well-defined, i.e., if there were s and s' such that $P_i(s) = P_i(s')$, then $f_iP_i(s) = f_iP_i(s') = d_i$. Since g preserves node colours (GM-1), f_i solves the problem. In other words, there must exist f_i such that $g = \langle f_iP_i\rangle$. □

We provide an example illustrating Thm. 8C.5.

Example 8C.6
The problem in Example 8C.2 is solvable in the conjunctive architecture (Example 8C.4), as we can construct the morphism depicted in Fig. 8.3.

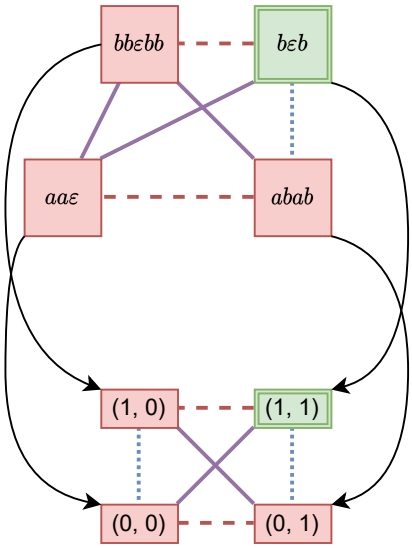

Figure 8.3: Graph morphism from the observation graph in Fig. 8.1 to the decision graph in Fig. 8.2.

Notice how the leftmost diagonal/purple/solid edge in the graph on top loses its redness/horizontalness and become a vertical/blue/dotted edge in the graph on the bottom. All other node/edge colours do not change through the morphism.

The morphism gives a solution to the problem. ∎

The following aspect distinguishes observation graphs from decision graphs. The equivalence relation $\sim_\varnothing$ on L is $\ker\langle P_i\rangle$, whereas $\sim_\varnothing$ on $\mathcal{D}$ is an identity relation. I.e., two distinct strings in L can be related by $\sim_\varnothing$ because they have identical projections; in contrast, the only way for two decisions in $\mathcal{D}$ to be related by $\sim_\varnothing$ is if they are identical. However, this distinction can be removed: in the spirit of Thm. 8C.5 we can collapse $\ker\langle P_i\rangle$ into an identity relation in advance. Then we revise the observation graphs to be based on $L/\ker\langle P_i\rangle$ instead of L.

Definition 8C.7 (Pre-folded Observation Graph)
Let $[s]$ be the set of strings whose projections are the same as s. Formally, let $[s] = \{\, t \in L \mid \langle P_i\rangle t = \langle P_i\rangle s \,\}$, i.e., $[s]$ is the equivalence class of s with respect to $\ker\langle P_i\rangle$. Define the relations $\sim_N$ on $L/\ker\langle P_i\rangle$ instead of directly on L, so that $[s] \sim_N [t]$ when the two tuples $\langle P_i\rangle s = (P_1(s), \ldots, P_n(s))$ and $\langle P_i\rangle t = (P_1(t), \ldots, P_n(t))$ differ by exactly the components indexed by N.

Moreover, since there is a natural bijection between $L/\ker\langle P_i\rangle$ and $\langle P_i\rangle L$, the pre-folded observation graph can also be defined in terms of $\langle P_i\rangle L$. In other words, for two strings that are indistinguishable to any agent, nothing is lost by aggregating the nodes representing the two strings.

Notice that the relations $\sim_N$ are well-defined, as the definition does not depend on the specific elements of the equivalence classes used in the definition. Also, the node colouring of the graph $L/\ker\langle P_i\rangle$ is well-defined when $\ker\langle P_i\rangle$ refines $\{ K, L-K \}$, i.e., for strings s and t in L with identical projections $P_i(s) = P_i(t)$ for all i, s and t must either be both from K or both not, and thus have the same colouring in the observation graph L. Hence, whenever we discuss the graph $L/\ker\langle P_i\rangle$, we implicitly assume that $\ker\langle P_i\rangle$ refines $\{ K, L-K \}$. Recall that $\ker\langle P_i\rangle$ refining $\{ K, L-K \}$ is necessary for any observation problem to be solvable.

The following remark is more easily seen if we take $\langle P_i\rangle L$ instead of $L/\ker\langle P_i\rangle$ as the observation graph: recall that the morphism from the observation graph L gives $g = \langle f_i P_i\rangle$. Pre-folding L through $\langle P_i\rangle$ into $\langle P_i\rangle L$ allows us to take advantage of the feasibility condition and factor out f_i from g. The forgoing is formally expressed in the following theorem and its attendant proof.

Theorem 8C.8 (restating Thm. 8C.5 in terms of pre-folded observation graph) An observation problem is solvable in a given architecture if and only if the following holds:

1. The node colouring of the graph $L/\ker\langle P_i\rangle$ is well-defined, namely, $\ker\langle P_i\rangle \leq \{ K, L-K \}$, i.e., $\forall\, s,t \in L.\ (\forall\, i \in \mathcal{N}.\ P_i(s) = P_i(t)) \Rightarrow \neg(s \in K \wedge t \in L-K)$.

2. There exists a morphism from the pre-folded observation graph $L/\ker\langle P_i\rangle$ to the decision graph (representing the architecture's fusion rule) satisfying the morphism conditions GM.

Proof. Recall that since $\langle P_i\rangle L$ is bijective to $L/\ker\langle P_i\rangle$, we can define the pre-folded observation graph on $\langle P_i\rangle L$ instead. Then the morphism g is precisely (f_i) (contrasting to $g = \langle f_i P_i\rangle$ in Thm. 8C.5). □

Recall that our motivation in defining the pre-folded observation graphs is to eliminate their distinction from the decision graphs. This provides us a mean of choosing an architecture for the solution. We have the following result.

Theorem 8C.9
An observation problem is solvable if and only if the node colouring of the graph $\langle P_i\rangle L$ is well-defined, i.e., $\ker\langle P_i\rangle \leq \{ K, L-K \}$.

The proof will be constructive by giving the architecture and local agents.

Proof. By Thm. 8C.8, it suffices to take $\langle P_i \rangle L$ as the decision graph. By the motivation of its construction, $\langle P_i \rangle L$ is a legitimate decision graph. Then the required morphism is the identity mapping. □

The condition that

$$\ker\langle P_i \rangle \leq \{ K, L - K \},$$

or more explicitly,

$$\forall\, s, t \in L\,.\, (\forall\, i \in N\,.\, P_i(s) = P_i(t)) \Rightarrow \neg(s \in K \wedge t \in L - K),$$

is equivalent to

$$\forall\, s \in K, t \in L - K\,.\, \exists\, i \in N\,.\, P_i(s) \neq P_i(t),$$

by contraposition. The last expression is called Joint Observability [Tri04]. Hence, we may call the architecture whose fusion rule is represented by $L/\ker\langle P_i \rangle$ as the joint architecture. As a consequence of Thm. 8C.9, the joint architecture is the most general architecture. However, as shown by Tripakis [Tri04], the problem solvability condition—Joint Observability—is undecidable. Nonetheless, in the cases where Joint Observability can be asserted (for example, by a mathematical proof), the solution can be trivially found as stated in the proof of Thm. 8C.9.

8D A Uniform Approach to Compare Fusion Rules

The traditional way to compare two fusion rules is by first obtaining a characterization of problem solvability with each of the fusion rules, and then demonstrate whether one characterization logically entails the other. In light of the discussion in the previous section, we can obtain a more direct way to compare fusion rules without deriving characterizations of problem solvability first.

Recall from the previous section, there is no formal distinction between observation graphs and decision graphs. Consequently, we have the following result.

Theorem 8D.1
Given fusion rules f, f' and their respective decision graphs $\mathcal{D}$, $\mathcal{D}'$, the fusion rule f' is more general than f if and only if there is a graph morphism from $\mathcal{D}$ to $\mathcal{D}'$ satisfying the graph morphism conditions GM.

Proof. ($\Leftarrow$): Suppose there is a morphism $g\colon \mathcal{D} \to \mathcal{D}'$ satisfying the morphism conditions GM. Consider an arbitrary observation problem solvable with the fusion rule f. By Thm. 8C.5, there is a morphism $h\colon L/\ker\langle P_i \rangle \to \mathcal{D}$ from the observation graph $L/\ker\langle P_i \rangle$ to the decision graph $\mathcal{D}$ satisfying the morphism conditions GM.

Then $g \circ h \colon L/\ker\langle P_i\rangle \to \mathcal{D}'$ is a morphism from the observation graph $L/\ker\langle P_i\rangle$ to the decision graph $\mathcal{D}'$, which clearly satisfies the morphism conditions GM, and therefore, by Thm. 8C.5, solves the observation problem. Thus, all problems solvable with f are also solvable with f'.

($\Rightarrow$): Suppose that all observation problems solvable with the fusion rule f are solvable with f'. A morphism from $\mathcal{D}$ to $\mathcal{D}'$ can be given as follows. Take $\mathcal{D}$ as isomorphically equivalent to the observation graph of some observation problem. Clearly the observation problem is solvable as the identity morphism over $\mathcal{D}$ satisfied GM. Then the problem is also solvable with the fusion rule f' by assumption. By Thm. 8C.5, there must be a morphism h from $\mathcal{D}$ to $\mathcal{D}'$ satisfying the morphism conditions GM. The morphism h is what we wanted.

To see why we can consider $\mathcal{D}$ as an observation graph, we construct an observation problem whose observation graph is isomorphic to $\mathcal{D}$. Recall that $\mathcal{D} \subseteq D \times \cdots \times D$. Construct the following problem. Take $\Sigma = \{\, (d,i) \mid d \in D \wedge i \in \mathcal{N} \,\}$ as our alphabet, where each symbol consists a decision d, tagged by an agent i, where (d,i) is alternatively written as d^i. To enforce the desired observability, take $\Sigma_{i,o} = \{\, d^i \mid d \in D \,\}$. Associate to each node $v = (d_1, \cdots, d_n)$ in $\mathcal{D}$ the string $s_v = d_1^1 \cdot \cdots \cdot d_n^n$, so that $P_i(s_v) = d_i^i$ as desired. Let $s_v \in K$ if v is coloured green/doubly-bordered, and $s_v \in L - K$ if v is coloured red/singly-bordered. The association gives an isomorphism from the observation graph to the decision graph $\mathcal{D}$ satisfying GM, where by "isomorphism" we mean that the morphism is bijective and preserves edge colouring. Note that, precisely in the case when the set of available decisions D is countably infinite, the alphabet is countably infinite.

The need for an infinite alphabet can be eliminated, as we can encode symbols in an infinite alphabet in terms of a finite alphabet. Using a finite alphabet instead however requires more sophistication in specifying the desired observability. Since we have assumed that the decision set is enumerable, let function $\llcorner\cdot\lrcorner : D \to \mathbb{N}$ be the enumeration of decisions in natural numbers. This enumeration function allows us to speak of the "j-th" decision in the set D. First take $\Sigma = \bigcup_{i\in\mathcal{N}} \{\, 0_i, 1_i \,\}$ and $\Sigma_{i,o} = \{\, 0_i, 1_i \,\}$. Then associate to each node $v = (d_1, \cdots, d_n)$ in $\mathcal{D}$ the string $s_v = 0_1^{\llcorner d_1 \lrcorner} 1_1 \cdots 0_n^{\llcorner d_n \lrcorner} 1_n$, so that $P_i(s_v) = 0_i^{\llcorner d_i \lrcorner} 1_i$. The intention of the encoding is that 0 enumerates decisions in unary notation, 1 marks the endings of code words, and subscripts impose observabilities. In other words, $0_i^{\llcorner d_i \lrcorner}$ means a string of 0's of length $\llcorner d_i \lrcorner$. The idea is that if d_i is the j'th decision in the set D, then it gets encoded by j 0's followed by 1.

Since the enumeration $\llcorner\cdot\lrcorner$ is injective, the encoding $d_i \mapsto 0_i^{\llcorner d_i \lrcorner} 1_i$ is also injective. Moreover, the encoding is prefix-free and hence instantaneously and uniquely decodable, i.e., the association to v of s_v is one-to-one. □

We illustrate the methodology in the proof of Thm. 8D.1 on the following example. The example uses a finite decision set, so that we can display the observation graph of our example however, the methodology is the same for infinite but countable sets D.

Example 8D.2
The decision graph of the conjunctive architecture in Example 8C.4 can be seen as the observation graph of the following problem. Let the enumeration function $\llcorner\cdot\lrcorner$ send the symbol 0 to the number 0 and the symbol 1 to the number 1. With $\mathcal{N} = \{1,\ 2\}$, let $\Sigma = \{0_1,\ 1_1,\ 0_2,\ 1_2\}$, $\Sigma_{1,o} = \{0_1,\ 1_1\}$, and $\Sigma_{2,o} = \{0_2,\ 1_2\}$. Let $L = \{1_1 1_2,\ 0_1 1_1 1_2,\ 1_1 0_2 1_2,\ 0_1 1_1 0_2 1_2\}$ and $K = \{0_1 1_1 0_2 1_2\}$. Then the observation graph is depicted in Fig. 8.4.

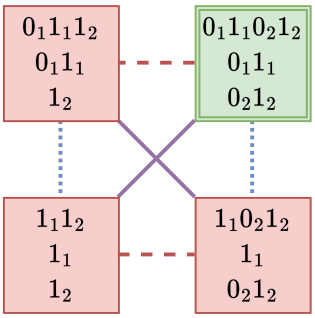

Figure 8.4: An observation graph that is isomorphic to the decision graph in Fig. 8.2.

We now illustrate how two architectures can be directly compared with our approach. We first show how one architecture can be determined to be strictly more general than another.

Example 8D.3
Consider the architecture in which the local decisions available are $\{0, 1, \mathrm{dk}\}$, where dk stands for "don't know". The associated fusion rule outputs 0 whenever a 0 local decision is present, and 1 whenever a 1 local decision is present, and is undefined when either there are conflicting local decisions (both 0 and 1 are present), or all supervisors don't know (all supervisors are confused).

This architecture is termed the C&P∧D&A architecture by Ritsuka and Rudie [RR22c] to correspond to the C&P (conjunctive and permissive) architecture [RW92] and the D&A architecture (disjunctive and anti-permissive) [PKK97].

The decision graph is depicted in Fig. 8.5, where grey/dash-bordered nodes indicate disallowed combination of local decisions.

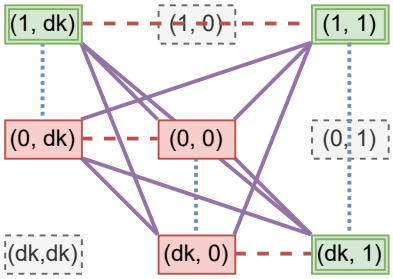

Figure 8.5: Decision graph for the C&P∧D&A architecture.

The C&P∧D&A architecture is known to be weaker than the C&P architecture (which we have been calling the "conjunctive architecture"). This fact can be readily demonstrated by giving a decision graph morphism. Fig. 8.6 depicts such a morphism, where for representation purpose we no longer make use of horizontalness/verticalness to denote edge colouring.

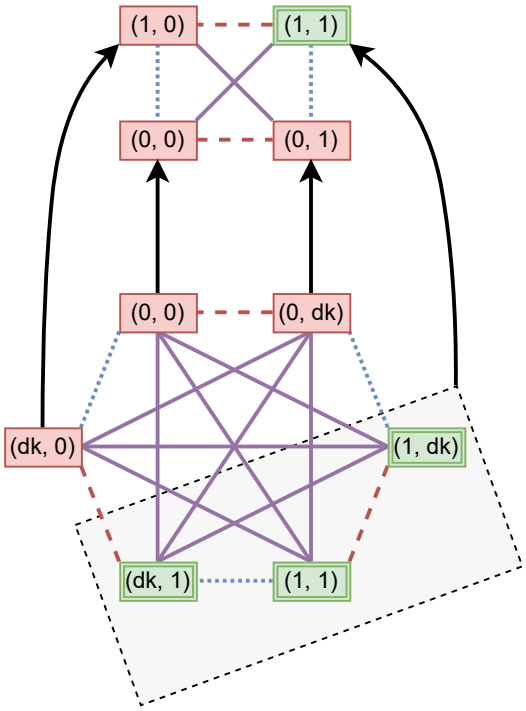

Figure 8.6: Graph morphism from the decision graph for the C&P∧D&A architecture (bottom) to the decision graph for the C&P architecture (top).

One can also see that there can be no morphism going in the other direction, as there is no green/doubly-bordered node in the bottom graph having both red/dashed and

blue/dotted edges to red/singly-bordered nodes, which is necessary for the node $(1, 1)$ in the top graph.

The following example shows how two seemingly different architectures can be determined to be equivalent.

Example 8D.4
An alternative way to describe the conjunctive architecture is by using three decisions $\{ 0, 1, \mathrm{cd} \}$, where cd is to be interpreted as a conditional decision, so that the fusion rule outputs 1 when only the conditional decision is present, and otherwise behaves identically to the fusion rule in the C&P∧D&A architecture as given in Example 8D.3 (although we renamed the decision dk to cd). The decision graph is depicted in Fig. 8.7 without edges for compactness.

Figure 8.7: Alternative decision graph for the conjunctive architecture.

It is easy to check that this architecture is indeed equivalent to the conjunctive architecture. Since the graph would be too complex to draw, we describe the morphisms verbally. The morphism h to the conjunctive architecture is like the morphism from the C&P∧D&A architecture to the conjunctive architecture as given in the previous example, where all green/doubly-bordered nodes are sent to $(1, 1)$. Unlike the C&P∧D&A architecture, now we have a morphism g from the conjunctive architecture: red/singly-bordered nodes are mapped by reversing h, where the only green/doubly-bordered node $(1, 1)$ is mapped to $(\mathrm{cd}, \mathrm{cd})$. That is, in the C&P architecture, the decision 1 can be interpreted as a conditional decision, which aligns with the interpretation in [RR22c].

The foregoing shows that there may exist two architectures that are equivalent in the sense of having morphisms in both directions, for example, the two architectures depicted in Figs. 8.2 and 8.7. However, although the architecture in Fig. 8.7 has more nodes than that of Fig. 8.2, the redundancy in this case serves a purpose: the original formulation of the conjunctive architecture by Rudie and Wonham [RW92] essentially forced the decision 1 (enable) to stand for both an agent actively enabling an event because the agent knew the event was legal, and passively enabling the

event when the agent didn't know if the event was legal. To understand the meaning behind agents' behaviours, it is useful to separate out the roles played by a decision, which is exactly what the architecture in Fig. 8.7 does.

The following example shows how two architectures can be determined to be incomparable.

Example 8D.5
Recall the decision graph for the conjunctive architecture in the left part of Fig. 8.8. Compare it with the disjunctive architecture [PKK97], also known as the D&A architecture, whose decision graph is depicted in the right part of Fig. 8.8.

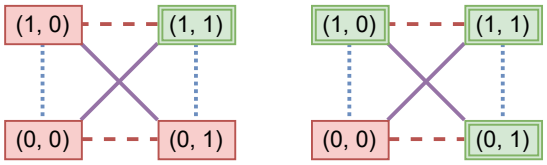

Figure 8.8: Decision graph for the conjunctive architecture recalled in the left, with the decision graph for the disjunctive architecture in the right.

One can see that there can be no morphism from left to right, as there is no green/doubly-bordered node in the right graph having both red/dashed and blue/dotted edges to red/singly-bordered nodes, which is necessary for the node $(1, 1)$ in the left graph. By a similar argument over the node $(0, 0)$ in the right graph, one can see that there can be no morphism from right to left either. This is sufficient to determine that the conjunctive architecture and the disjunctive architecture are incomparable. This confirms the result by Yoo and Lafortune [YL02].

8E Conclusion

We proposed two useful tools in studying decentralized observation problems: observation graphs and decision graphs. The decision graphs alone provide a systematic approach to directly compare decentralized architectures. Together with observation graphs, we have systematic approaches to derive problem solvabilities and solutions.

As we can see in the development of decentralized observation problems, the earlier works propose verifiable characterizations for problem solvability and computable

algorithms to construct solutions [Cie+88; RW92; PKK97; YL02; YL04], but subsequent works can no longer provide computable solvability characterizations, let alone algorithms to construct solutions [KT05; CK11]. Said differently, finding a graph morphism may be hard, but verifying a witness could be easier. Specifically, when the graphs involved are finite, the problem can be solved in nondeterministic polynomial time, but has proofs verifiable in polynomial time. When the graphs are infinite, the problem can be undecidable, but proofs can be verified. This suggests that in a situation where the solvability characterization becomes undecidable, one should attempt to prove the characterization instead. Moreover, when a solution is "finite" in some sense, e.g., the solution is described by finite state automata, it remains verifiable.

In summary, fusion rules with unbounded numbers of decisions present challenges for finding graph morphisms. Nonetheless, for fusion rules with finite, bounded numbers of decisions, our work provides a direct and easy approach to compare the corresponding architectures.

References

[Cie+88] R. Cieslak, C. Desclaux, A. S. Fawaz, and P. Varaiya. "Supervisory control of discrete-event processes with partial observations". In: *IEEE Transactions on Automatic Control* 33.3 (Mar. 1988), pp. 249–260. DOI: 10.1109/9.402. [cit. on p. 109].

[CK11] H. Chakib and A. Khoumsi. "Multi-Decision Supervisory Control: Parallel Decentralized Architectures Cooperating for Controlling Discrete Event Systems". In: *IEEE Transactions on Automatic Control* 56.11 (Nov. 2011), pp. 2608–2622. DOI: 10.1109/tac.2011.2128730. [cit. on p. 109].

[Fag+04] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Vardi. *Reasoning About Knowledge*. The MIT Press, 2004. DOI: 10.7551/mitpress/5803.001.0001. [cit. on p. 113].

[KT05] R. Kumar and S. Takai. "Inference-based Ambiguity Management in Decentralized Decision-Making: Decentralized Control of Discrete Event Systems". In: *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005. DOI: 10.1109/CDC.2005.1582701. [cit. on p. 109].

[PKK97] J. H. Prosser, M. Kam, and H. G. Kwatny. "Decision fusion and supervisor synthesis in decentralized discrete-event systems". In: *Proceedings of the American Control Conference*. IEEE, 1997. DOI: 10.1109/ACC.1997.608978. [cit. on p. 109. 110. 120. 123].

[RM13] S. L. Ricker and H. Marchand. “A parity-based architecture for decentralized discrete-event control”. In: *2013 American Control Conference*. IEEE, June 2013. DOI: 10.1109/acc.2013.6580727. [cit. on p. 109].

[RR00] S. L. Ricker and K. Rudie. “Know means no: Incorporating knowledge into discrete-event control systems”. In: *IEEE Transactions on Automatic Control* 45.9 (2000), pp. 1656–1668. DOI: 10.1109/9.880616. [cit. on p. 110. 113].

[RR07] S. L. Ricker and K. Rudie. “Knowledge Is a Terrible Thing to Waste: Using Inference in Discrete-Event Control Problems”. In: *IEEE Transactions on Automatic Control* 52.3 (Mar. 2007), pp. 428–441. DOI: 10.1109/TAC.2007.892371. [cit. on p. 110. 113].

[RR21] K. Ritsuka and Karen Rudie. “A Visualization of Inference-Based Supervisory Control in Discrete-Event Systems”. In: *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, Dec. 2021. DOI: 10.1109/cdc45484.2021.9683210. [cit. on p. 110. 113].

[RR22a] K. Ritsuka and Karen Rudie. *A correspondence between control and observation problems in decentralized discrete-event systems*. 2022. arXiv: 2204.10792 [eess.SY]. [cit. on p. 110].

[RR22c] K. Ritsuka and Karen Rudie. “Epistemic interpretations of decentralized discrete-event system problems”. In: *Discrete Event Dynamic Systems* 32.3 (June 2022), pp. 359–398. DOI: 10.1007/s10626-022-00363-7. [cit. on p. 109. 110. 113. 120. 122].

[RW92] K. Rudie and W. M. Wonham. “Think globally, act locally: decentralized supervisory control”. In: *IEEE Transactions on Automatic Control* 37.11 (1992), pp. 1692–1708. DOI: 10.1109/9.173140. [cit. on p. 109. 110. 113. 114. 120. 122. 124].

[Tri04] Stavros Tripakis. “Undecidable problems of decentralized observation and control on regular languages”. In: *Information Processing Letters* 90.1 (Apr. 2004), pp. 21–28. DOI: 10.1016/j.ipl.2004.01.004. [cit. on p. 118].

[YL02] T.-S. Yoo and Stéphane Lafortune. “A General Architecture for Decentralized Supervisory Control of Discrete-Event Systems”. In: *Discrete Event Dynamic Systems* 12.3 (2002), pp. 335–377. DOI: 10.1023/a:1015625600613. [cit. on p. 109. 110. 123. 124].

[YL04] T.-S. Yoo and S. Lafortune. “Decentralized Supervisory Control With Conditional Decisions: Supervisor Existence”. In: *IEEE Transactions on Automatic Control* 49.11 (Nov. 2004), pp. 1886–1904. DOI: 10.1109/tac.2004.837595. [cit. on p. 109. 124].

9 Discussion

This chapter first summarizes the achievements of the thesis.

First, due to the equivalence of observation, control, and diagnosis problems as demonstrated in Chapter 7, we will henceforth speak of “decentralized problems” and “decentralized architectures” without further qualifications.

As demonstrated in Chapters 3 to 5, we are able to cast a number of existing decentralized architectures in a unified framework based on epistemic logic. The unified framework provides a more intuitive understanding of these architectures and allows one to derive solvability conditions and solution specifications methodologically for these architectures. Chapter 6 additionally provides a visual alternative to this formalism.

Chapter 8 proposes a graph-theoretic translation of the epistemic logic formalism as a unifying framework. The chapter greatly simplifies the notation-heaviness of the epistemic logic formalism, while preserving its intuitiveness.

The work here opens up the following avenues for future research. First, given the undecidability, one may ask if there is a suitably large decidable subclass of problems whose solvability is decidable. On the other hand, one may also ask if there is a suitably general architecture under which the problem solvability is decidable. Both of the two questions above involve a judgement of what is suitable, i.e., evaluation against real applications. However, to the author’s knowledge, there does not exist a benchmark of realistic problems for evaluating architectures.

Moreover, a specific problem about closed-loop systems is that if the desired behaviour cannot be synthesized (under a prescribed architecture), it is desired to modify the problem requirement and construct a maximally-permissive (or in other words, least-restrictive) control policy (under the same architecture). While the problem has been solved for a few simple architectures, no work seems to have been done for the more complicated architectures, let alone with the use of a uniform approach to solve such problems.

The direct approach for comparing architectures provided in Chapter 8 can aid evaluating an architecture’s generality, so that one does not have to compare architectures indirectly through their problem solvability conditions, where the latter method may require one to come up with a clever example to show that an architecture is strictly more general than another. Then, one may wish to follow the approach demonstrated in Chapter 4 to methodologically derive a problem

solvability condition and synthesize a solution for a chosen architecture. Finally, although there is no definite solution for finding a maximally-permissive control policy, the discussion in Chapter 3 might be a prominent starting point for further research.

Bibliography

[Cie+88] R. Cieslak, C. Desclaux, A. S. Fawaz, and P. Varaiya. “Supervisory control of discrete-event processes with partial observations”. In: *IEEE Transactions on Automatic Control* 33.3 (Mar. 1988), pp. 249–260. DOI: 10.1109/9.402. [cite on Chapter 1 (p. 1), 3 (p. 18), 4 (pp. 59, 60), 7 (pp. 100, 105), 8 (pp. 109, 124)].

[CK08a] Hicham Chakib and Ahmed Khoumsi. “Multi-decision C&P∨D&A architecture for the decentralized control of discrete event systems”. In: *2008 IEEE International Conference on Automation Science and Engineering*. IEEE, Aug. 2008. DOI: 10.1109/COASE.2008.4626526. [cite on Chapter 1 (p. 1)].

[CK08b] Hicham Chakib and Ahmed Khoumsi. “Multi-decision decentralized control of discrete event systems : Application to the C&P architecture”. In: *2008 9th International Workshop on Discrete Event Systems*. IEEE, 2008. DOI: 10.1109/WODES.2008.4605993. [cite on Chapter 1 (p. 1)].

[CK11] H. Chakib and A. Khoumsi. “Multi-Decision Supervisory Control: Parallel Decentralized Architectures Cooperating for Controlling Discrete Event Systems”. In: *IEEE Transactions on Automatic Control* 56.11 (Nov. 2011), pp. 2608–2622. DOI: 10.1109/tac.2011.2128730. [cite on Chapter 1 (p. 1), 7 (p. 100), 8 (pp. 109, 124)].

[CL07] Christos G. Cassandras and Stéphane Lafortune. *Introduction to Discrete Event Systems*. Second. Springer-Verlag GmbH, 2007. 772 pp. [cite on Chapter 2 (pp. 5, 6)].

[DLT00] Rami Debouk, Stéphane Lafortune, and Demosthenis Teneketzis. In: *Discrete Event Dynamic Systems* 10.1/2 (2000), pp. 33–86. DOI: 10.1023/a:1008335115538. [cite on Chapter 7 (pp. 100, 101)].

[Fag+04] Ronald Fagin, Joseph Y. Halpern, Yoram Moses, and Moshe Vardi. *Reasoning About Knowledge*. The MIT Press, 2004. DOI: 10.7551/mitpress/5803.001.0001. [cite on Chapter 2 (p. 10), 8 (p. 113)].

[HM90] Joseph Y. Halpern and Yoram Moses. “Knowledge and common knowledge in a distributed environment”. In: *Journal of the ACM* 37.3 (July 1990), pp. 549–587. DOI: 10.1145/79147.79161. [cite on Chapter 2 (p. 10)].

[HMU06] John E. Hopcroft, Rajeev Motwani, and Jeffrey D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. 3rd ed. USA: Addison-Wesley Longman Publishing Co., Inc., 2006. [cite on Chapter 6 (p. 88)].

[KC18] Ahmed Khoumsi and Hicham Chakib. “Decentralized Supervisory Control of Discrete Event Systems: An Arborescent Architecture to Realize Inference-Based Control”. In: *IEEE Transactions on Automatic Control* 63.12 (Dec. 2018), pp. 4278–4285. DOI: 10.1109/tac.2018.2811785. [cite on Chapter 1 (p. 1)].

[KT05] R. Kumar and S. Takai. “Inference-based Ambiguity Management in Decentralized Decision-Making: Decentralized Control of Discrete Event Systems”. In: *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005. DOI: 10.1109/CDC.2005.1582701. [cite on Chapter 1 (p. 1), 3 (p. 39), 6 (p. 87), 7 (p. 100), 8 (pp. 109, 124)].

[KT07] R. Kumar and S. Takai. “Inference-Based Ambiguity Management in Decentralized Decision-Making: Decentralized Control of Discrete Event Systems”. In: *IEEE Transactions on Automatic Control* 52.10 (Oct. 2007), pp. 1783–1794. DOI: 10.1109/TAC.2007.906158. [cite on Chapter 4 (pp. 61, 72, 75, 76), 5 (pp. 79, 81), 6 (pp. 85, 86, 87, 88, 90, 91, 92, 94, 95)].

[LW88] F. Lin and W. M. Wonham. “On observability of discrete-event systems”. In: *Information Sciences* 44.3 (Apr. 1988), pp. 173–198. DOI: 10.1016/0020-0255(88)90001-1. [cite on Chapter 3 (p. 39), 7 (pp. 100, 105)].

[PKK97] J. H. Prosser, M. Kam, and H. G. Kwatny. “Decision fusion and supervisor synthesis in decentralized discrete-event systems”. In: *Proceedings of the American Control Conference*. IEEE, 1997. DOI: 10.1109/ACC.1997.608978. [cite on Chapter 1 (p. 1), 2 (pp. 6, 9), 3 (pp. 17, 18, 19, 25, 37), 4 (pp. 59, 60), 5 (p. 82), 6 (p. 85), 7 (p. 100), 8 (pp. 109, 110, 120, 123, 124)].

[QK06] Wenbin Qiu and R. Kumar. “Decentralized failure diagnosis of discrete event systems”. In: *IEEE Transactions on Systems, Man, and Cybernetics - Part A: Systems and Humans* 36.2 (Mar. 2006), pp. 384–395. DOI: 10.1109/tsmca.2005.853503. [cite on Chapter 7 (pp. 100, 101)].

[RM13] S. L. Ricker and H. Marchand. “A parity-based architecture for decentralized discrete-event control”. In: *2013 American Control Conference*. IEEE, June 2013. DOI: 10.1109/acc.2013.6580727. [cite on Chapter 8 (p. 109)].

[RR00] S. L. Ricker and K. Rudie. “Know means no: Incorporating knowledge into discrete-event control systems”. In: *IEEE Transactions on Automatic Control* 45.9 (2000), pp. 1656–1668. DOI: 10.1109/9.880616. [cite on Chapter 1 (p. 1), 2 (p. 10), 3 (pp. 23, 24, 47), 4 (pp. 75, 76), 8 (pp. 110, 113)].

[RR07] S. L. Ricker and K. Rudie. “Knowledge Is a Terrible Thing to Waste: Using Inference in Discrete-Event Control Problems”. In: *IEEE Transactions on Automatic Control* 52.3 (Mar. 2007), pp. 428–441. DOI:

10.1109/TAC.2007.892371. [cite on Chapter 1 (p. 1), 2 (pp. 10, 12, 13, 14), 3 (pp. 24, 39, 47), 4 (pp. 60, 64, 72, 75), 5 (p. 82), 8 (pp. 110, 113)].

[RR21] K. Ritsuka and Karen Rudie. "A Visualization of Inference-Based Supervisory Control in Discrete-Event Systems". In: *2021 60th IEEE Conference on Decision and Control (CDC)*. IEEE, Dec. 2021. DOI: 10.1109/cdc45484.2021.9683210. [cite on Chapter 1 (p. 3), 3 (p. 39), 4 (p. 72), 8 (pp. 110, 113)].

[RR22a] K. Ritsuka and Karen Rudie. *A correspondence between control and observation problems in decentralized discrete-event systems*. 2022. arXiv: 2204.10792 [eess.SY]. [cite on Chapter 1 (p. 3), 4 (p. 66), 8 (p. 110)].

[RR22b] K. Ritsuka and Karen Rudie. *A Uniform Treatment of Architectures and Fusion Rules in Decentralized Discrete-Event Systems*. 2022. arXiv: 2210.16511 [eess.SY]. [cite on Chapter 1 (p. 3)].

[RR22c] K. Ritsuka and Karen Rudie. "Epistemic interpretations of decentralized discrete-event system problems". In: *Discrete Event Dynamic Systems* 32.3 (June 2022), pp. 359–398. DOI: 10.1007/s10626-022-00363-7. [cite on Chapter 1 (p. 3), 4 (pp. 60, 61), 5 (p. 82), 8 (pp. 109, 110, 113, 120, 122)].

[RR23] K. Ritsuka and K. Rudie. *Do What You Know: Coupling Knowledge with Action in Discrete-Event Systems*. Submitted for publication. 2023. [cite on Chapter 1 (p. 3), 3 (pp. 17, 23, 26, 31, 32, 33, 34, 39), 6 (p. 96)].

[RW87] P. J. Ramadge and W. M. Wonham. "Supervisory Control of a Class of Discrete Event Processes". In: *SIAM Journal on Control and Optimization* 25.1 (Jan. 1987), pp. 206–230. DOI: 10.1137/0325013. [cite on Chapter 3 (p. 23), 7 (p. 100)].

[RW90] Karen Rudie and W. Murray Wonham. "The infimal prefix-closed and observable superlanguange of a given language". In: *Systems & Control Letters* 15.5 (Dec. 1990), pp. 361–371. DOI: 10.1016/0167-6911(90)90059-4. [cite on Chapter 3 (pp. 37, 43)].

[RW92] K. Rudie and W. M. Wonham. "Think globally, act locally: decentralized supervisory control". In: *IEEE Transactions on Automatic Control* 37.11 (1992), pp. 1692–1708. DOI: 10.1109/9.173140. [cite on Chapter 1 (p. 1), 2 (pp. 9, 12), 3 (pp. 17, 18, 19, 20, 37), 4 (pp. 59, 60), 5 (p. 82), 6 (p. 85), 7 (pp. 100, 105), 8 (pp. 109, 110, 113, 114, 120, 122, 124)].

[RW95] K. Rudie and J. C. Willems. "The computational complexity of decentralized discrete-event control problems". In: *IEEE Transactions on Automatic Control* 40.7 (July 1995), pp. 1313–1319. DOI: 10.1109/9.400469. [cite on Chapter 7 (p. 105)].

[Sam+95] M. Sampath, R. Sengupta, S. Lafortune, K. Sinnamohideen, and D. Teneketzis. “Diagnosability of discrete-event systems”. In: *IEEE Transactions on Automatic Control* 40.9 (1995), pp. 1555–1575. DOI: 10.1109/9.412626. [cite on Chapter 7 (pp. 100, 101)].

[ST02] R. Sengupta and S. Tripakis. “Decentralized diagnosability of regular languages is undecidable”. In: *Proceedings of the 41st IEEE Conference on Decision and Control, 2002.* IEEE, 2002. DOI: 10.1109/cdc.2002.1184531. [cite on Chapter 7 (pp. 100, 101, 103)].

[TK08] Shigemasa Takai and Ratnesh Kumar. “Synthesis of Inference-Based Decentralized Control for Discrete Event Systems”. In: *IEEE Transactions on Automatic Control* 53.2 (Mar. 2008), pp. 522–534. DOI: 10.1109/tac.2007.915171. [cite on Chapter 4 (p. 72), 6 (pp. 86, 87, 91, 92, 95)].

[TKU05] S. Takai, R. Kumar, and T. Ushio. “Characterization of co-observable languages and formulas for their super/sublanguages”. In: *IEEE Transactions on Automatic Control* 50.4 (Apr. 2005), pp. 434–447. DOI: 10.1109/tac.2005.844724. [cite on Chapter 2 (p. 9), 3 (pp. 17, 18, 19, 27, 29, 40, 41, 42, 43)].

[Tri] Stavros Tripakis. *Distributed Synthesis*. Virtual Lightning Tutorial Series on Discrete Event Systems 2021. [Accessed May 30, 2022]. URL: https://owncloud.mpi-sws.org/index.php/s/Q3RKiiQ67GPJJw4/download?path=%5C%2F%5C&files=04_STripakis_Slides.pdf. [cite on Chapter 7 (p. 99)].

[Tri04] Stavros Tripakis. “Undecidable problems of decentralized observation and control on regular languages”. In: *Information Processing Letters* 90.1 (Apr. 2004), pp. 21–28. DOI: 10.1016/j.ipl.2004.01.004. [cite on Chapter 7 (pp. 100, 101, 103, 105), 8 (p. 118)].

[TU01] S. Takai and T. Ushio. “Strong co-observability conditions for decentralized supervisory control of discrete event systems”. In: *Proceedings of the 40th IEEE Conference on Decision and Control*. IEEE, 2001. DOI: 10.1109/cdc.2001.980821. [cite on Chapter 2 (p. 9), 3 (pp. 26, 29, 47)].

[TU02] Shigemasa Takai and Toshimitsu Ushio. “A modified normality condition for decentralized supervisory control of discrete event systems”. In: *Automatica* 38.1 (Jan. 2002), pp. 185–189. DOI: 10.1016/s0005-1098(01)00187-x. [cite on Chapter 3 (p. 53)].

[WC18] W. Murray Wonham and Kai Cai. *Supervisory Control of Discrete-Event Systems*. Springer-Verlag GmbH, 2018. 487 pp. [cite on Chapter 2 (p. 5)].

[WYL07] Yin Wang, Tae-Sic Yoo, and Stéphane Lafortune. “Diagnosis of Discrete Event Systems Using Decentralized Architectures”. In: *Discrete Event Dynamic Systems* 17.2 (Jan. 2007), pp. 233–263. DOI: 10.1007/s10626-006-0006-8. [cite on Chapter 7 (pp. 100, 101)].

[YL02] T.-S. Yoo and Stéphane Lafortune. “A General Architecture for Decentralized Supervisory Control of Discrete-Event Systems”. In: *Discrete Event Dynamic Systems* 12.3 (2002), pp. 335–377. DOI: 10.1023/a:1015625600613. [cite on Chapter 1 (p. 1), 2 (p. 6), 3 (pp. 18, 19, 31), 4 (pp. 59, 60), 5 (p. 82), 6 (p. 85), 7 (p. 100), 8 (pp. 109, 110, 123, 124)].

[YL04] T.-S. Yoo and S. Lafortune. “Decentralized Supervisory Control With Conditional Decisions: Supervisor Existence”. In: *IEEE Transactions on Automatic Control* 49.11 (Nov. 2004), pp. 1886–1904. DOI: 10.1109/tac.2004.837595. [cite on Chapter 1 (p. 1), 2 (pp. 7, 12), 3 (pp. 37, 39), 4 (pp. 59, 61), 5 (p. 82), 7 (p. 100), 8 (pp. 109, 124)].

[YL05] Tae-Sic Yoo and S. Lafortune. “Decentralized supervisory control with conditional decisions: supervisor realization”. In: *IEEE Transactions on Automatic Control* 50.8 (Aug. 2005), pp. 1205–1211. DOI: 10.1109/tac.2005.852556. [cite on Chapter 4 (pp. 59, 75)].