SCUOLA DI DOTTORATO
UNIVERSITÀ DEGLI STUDI DI MILANO-BICOCCA

Department of Computer Science

PhD program in Computer Science Cycle XXXI

Algebraic Structures for the Analysis of Distributability of Elementary Systems and their Processes

Surname: Puerto Aubel Name: Adrián

Registration number: 810595

Tutor: Gianluca Della Vedova

Supervisor: Lucia Pomello
Co-supervior: Luca Bernardinello

Coordinator: Stefania Bandini

ACADEMIC YEAR 2018/2019

Abstract

This work studies systems, and the processes they execute, in the way they can be distributed. To this aim, the central notion is that when a system is distributed, a remote observation requires an exchange of information from the different locations of the system.

The chosen formalisms are taken in the framework of Petri net theory. Elementary net systems, and condition/event net systems provide specifications for the systems. Causal nets and partially ordered sets allow for modelling processes. With these last formalisations, the state of the art provides a notion of subprocesses that can be structured so as to carry information on how a process can be distributed. This structure is formalised as an orthomodular lattice. This work shows that the minimal non trivial elements of this lattice, the minimal subprocesses, can be ordered so as to provide an abstraction of the process. The nature of this notion of subprocess permits to show that this abstraction depicts the localities of the process, parts of the process which can run independently from each other.

The behaviour of elementary, and condition/event net systems, is modelled with labelled transition systems. This work adheres to an interpretation of the set of elementary regions, as the one of locally observable properties of the system, motivated by elementary net synthesis. According to this interpretation, elementary regions represent a suitable specification of the available infrastructure on which to distribute a system. The state of the art shows that the set of regions of an elementary, or condition/event system, forms an orthomodular poset, thus providing a way to retrieve a canonical labelled transition system such that all regions of the orthomodular poset are also regions of it. The question of whether this canonical transition system has more regions than the specified ones is an open problem. The canonical transition system is the largest one can obtain from an orthomodular poset, in the sense that systems complying with the specification, can be found as subsystems of it. However, not all its subsystems display the same regional structure. This work presents a sufficient condition for this to happen. This is achieved by providing a structure to the set of events, or labels, of the canonical system, which reflects concurrency.

An orthomodular poset is called stable when it is isomorphic to the set of regions of its canonical transition system. The state of the art shows that when the first poset is of a given class, it embeds in the second. It is conjectured that all posets that arise as the sets of elementary regions of an elementary system, regional posets, are stable. This work provides a condition necessary for an orthomodular poset to be regional, and shows that when it holds, the embedding is strong. Not every embedding is strong, but all isomorphisms are, in particular, strong embeddings. This result implies that the embedding maps minimal regions to minimal regions.

Contents

Chapter 1

Introduction

A Distributed System consists of several computing components, each of which is located at a different point in space. A network of computers distributed around the globe is an example of such a system.

Different models allow for description and specification of such systems, among which Labelled Transition Systems and Petri Nets are widespread paradigms. The two are very closely related in that Labelled Transition Systems are commonly used to represent the behaviour of Petri Net Systems. Indeed, the former capture statically, in one single snapshot, all possible behaviours of the system, whereas the latter rely on a dynamic expression of behaviour to more succinctly represent the system.

Petri Net models are, in this sense, closer to the implementation level, and they are, for instance, used in the design of digital circuits, or logistic networks. Furthermore, the Petri Net paradigm allows for representation of the processes a system is able to execute.

The Elementary setting refers to subclasses of either Labelled Transition Systems and Petri Nets, the evolutions of which are guided strictly by Boolean conditions. The execution of a possible action relies solely on the true or false value of series of variables, or flags. In this sense, elementary systems are bare of any data carrying, or counting capacity. However, this restriction on the expressive power of the models permits a formal analysis of concurrency of the systems in logical terms.

Indeed, in the elementary framework, given the specification of either a system or a process in one of the aforementioned models, concurrency features can be distilled into a particular class of algebraic structures. These structures carry the relevant information on the extent to which the system, or process, can be distributed in space, allowing for exploitation of the computational

power of as many components as possible. It is in this sense, that the way a system, or process, can be distributed is referred to as its distributability.

This first chapter is structured in four sections, the first of which is a brief presentation of the field of distributed systems. The second section deals with the mathematical formalisations of these, and motivates the choice of the models at stake in this work. The third section presents the main principles that will be followed when analysing how a system can be distributed. Finally, the fourth section presents the structure of the thesis, underlining its main contributions.

1.1 Distributed Systems

Distributed Systems have the characteristic that components do not need to share memory or clocks, hence all communication is performed by message passing [25]. In the words of F. Mattern: "A distributed system can be characterised by the fact that the global state is distributed and that a common time base does not exist" [41]. When a system is composed of one single component, namely one locality, and one clock, it is said to be fully sequential, its computations are totally ordered sequences of states. When composed of several sequential components, a system is distributed, and is able to perform actions corresponding to each of its components asynchronously, in parallel.

The study of systems, distributed or sequential, is, apart from the obvious theoretical interest, of great use in practice. Indeed, when designing a system, such as a microprocessor, or a network protocol, one will certainly want to verify that the chosen implementation complies with a given specification. For instance, a CPU should never reach a deadlock state, or full stop of the system, since in this situation, a user could not get any service from it, thus violating the requirements of the specification. Model-checking is a field of computer science devoted to the verification of such a compliance of the system to the specification (see for example [24]). In general, specifications are provided in some formal language, suitable to express behaviours of the system. Temporal logics are commonly used in this frame. The converse problem of model-checking is the synthesis. In this case, one would like to design a procedure, which, given a suitable specification, would automatically generate a model of a system compliant with it. These two approaches are well settled, and an extensive literature exists presenting techniques and solutions. However, most of them regard fully sequential systems, and in general do not scale very well to the distributed case.

The reasons for this lack of scalability are of different natures. First, ei-

ther model-checking, or synthesis techniques need, in the distributed case, to face the so called "state explosion problem". This problem is due to the fact that independent components evolve independently, unless communication is set to synchronise them. In such a situation, a global state of the system will represent a combination of the local states of each component. Since each such component is able to change its state independently, the number of global states, as combinations of local states, grows exponentially with the number of components. This exponential increment presents a great barrier in practical application, in particular for large systems, and has led to the development of a set of alternatives such as structural analysis, and the lately very popular modular approach. This last approach focuses on seeing the whole system as the composition of smaller modules, such that the properties to be checked hold on the whole system whenever they hold on each of the modules. Under these circumstances, it would be sufficient to check the properties in the possibly logarithmically smaller modules, to assert that the property holds in the system. In the case of the synthesis, the limitations are even greater. For instance, although it is known that the reactive synthesis problem, also known as Churchs solvability problem, admits a solution in 2-EXP time in the sequential case [49], it was shown to be undecidable in the general distributed case [50]. Late advances in this field have however provided decidable subclasses of the problem, dealing still with a very high complexity [32].

Another strong limitation for the analysis of distributed systems is the principle of locality itself. When reasoning about testable properties of such a system, it will be assumed that an observer can not be at two different localities simultaneously. Hence, his knowledge about the global state of the system is restricted to what he can learn by observing a given sequential component. Note that he might be able to infer information about other components of the system, from the way they interface with the one at its location. However, this information remains incomplete. Indeed, the observation of a distant component is subject to a form of uncertainty principle: to receive information about the remote system, this one will have to send a message, thus changing its state. Furthermore, the lack of a global clock prevents the observer from gaining information about previous states of the system, for it would require to know the state of several components at a given instant. In his seminal paper *Time, Clocks, and the Ordering of Events in a Distributed System*, L. Lamport [40] addressed the question of comparing the delay in message transmission with the time-lapse between events of a single sequential process. When this delay is not negligible with respect to execution times in the process, the notion of simultaneity only has sense at actual interfaces.

This principle justifies a mathematical link between this work and quantum

mechanics. The model with which these observable properties of distributed systems are represented was mainly developed and studied by Garrett Birkhoff and John von Neumann as early as 1936 [19], in an attempt to axiomatically formalise the set of testable properties of a quantum system. A rich theory has been developed in this topic ever since. In this work, results from that field will be extensively used. In particular, the notion of quantum logic, formalised as an orthomodular partial order, will be presented as a suitable algebraic structure to represent the logic of observable properties of distributed systems. This approach was introduced in [8], and further developed in [7].

Although the principle of uncertainty allows for the analogy with quantum mechanics, the systems studied in this work are of a very different nature than those of quantum physics. So even though the mathematical models developed for the quantum theory are a suitable frame for the present study, focus will be put on a particular subclass of such models, that differs from the cases of interest in that field. As a matter of fact, the subclass of interest will arise very naturally from the study of classical models of distributed systems.

1.2 Models of Computation

Distributed computation admits different kinds of formalisation. As models of computation, they evolve around the central notion of Automaton. An automaton is a mathematical model of a computing device, or system. It consists of a finite set of states, usually depicted as points, or vertices of a graph, and a set of labelled arcs linking them, represented as labelled arrows between the corresponding nodes. The labels on the arcs represent actions that, when performed by the system, lead this one from one state to another. A particular state is set as initial, meaning that the system starts in that state, and remains there until reading any kind of input. Sequences of actions which correspond to a path from the initial state on the automaton are interpreted as a computation. In the classical setting, some states are marked as accepting, indicating when reached, that the corresponding computation terminates, and is thus a valid computation for such a model. The main drawback of such a simple representation is that a given automaton can only recognise a restricted set of computations, those corresponding to a regular expression. Classical theory deals with more general models so as to include memory, yielding stack automata, and Turing machines. The latter represent the main paradigm of general computation, due to its versatility, and its capacity to adapt to the input. Indeed, it represents the main model of programmable machines, in the sense that the input, encoded as a string of characters in the memory,

can alter the way the input itself will be treated. As a matter of fact, the well-known Church-Turing hypothesis conjectures that any computation can be performed by a suitable Turing machine.

All these paradigms, however, deal with a finite notion of computation. Indeed, in such models, a computation is a sequence of actions that terminates. In some cases however, this assumption is very limiting. One can easily think of a program, such as an operating system, that after performing a given computation would remain idle until further instructions are given by the user. Reactive systems are a set of mathematical models that deal with this possibility. In most cases, the set of accepting states is simply substituted by a set of accepting conditions, that can vary depending on the requirements. These can be simply reaching a given set of states, or requiring that they be visited infinitely often. When abstracting from accepting conditions, one simply considers a set of states, labelled arcs between them, and in some cases an initial state. Such a model is called a *labelled transition system*, and will be the focus of an important part of the study presented in this work. The choice of such model is motivated by its generality, in the sense that it can be further interpreted by deciding suitable accepting conditions, or extended with different memory formalisations. On the other hand, it is a suitable frame for the study of concurrency, since one can easily express features of concurrent systems on it. Indeed, when interpreted as a distributed system, a labelled transition system depicts concurrency by means of the so called interleaving semantics. In this frame, states of the labelled transition system represent global states of the system. Actions, in fact, are assumed to be bound to a locality, so that an action belonging to different components of the system can only represent a synchronisation among these. In the classes of labelled transition systems that will be considered in this work, one can further determine if two actions are concurrent, whenever at a given global state, executing them leads the system to a same single state, independently of the order in which they are executed. In this case, one may say the two actions commute. Note that in such a situation, the mid-states, in which one of the actions has been executed, but yet not the other, do not coincide. The concurrent occurrence of two actions will hence be characterised by such four states: the initial one, two mid-states in which one action has occurred but not the other, and the single state in which both have been performed. Such a configuration is called a diamond. This is the simple case of two concurrent actions, but the notion of diamond can be naturally extended to arbitrary sets of concurrent actions, each to be performed by a sequential component of the system. Note that the number of global mid-states grows exponentially with the number of concurrent actions. Indeed, a diamond involving three

actions will consist of 8 global states, whereas a diamond involving 4 actions will count 16 states. This exponential number of mid-states is actually what was earlier introduced as the state space explosion problem. It represents the main drawback of such models, be it for the analysis or the design of systems.

A great effort has been put by the community in order to overcome this problem. It has lead to a wide range of models, all meant to deal with concurrency while trying to avoid expressing mid-states of diamonds. Each of such models present specific advantages, and the intended use should motivate their choice. For instance, asynchronous automata [63] are suitable for the study of formal languages in a concurrent setting, whereas interface automata [28] are best suited for the analysis of the interactions of the different components of a system. This work however, will focus on the paradigm of Petri nets [56]. Petri nets present the advantage of explicitly depicting concurrency. Indeed in this model the localities are explicitly represented, and actions are not simple labels on edges, but rather nodes of their own, that interact directly with the localities of the system.

1.3 Approach to the Analysis of Distributability

In this work, Petri net models are taken as the basic representation of a distributed system. The reason for this choice is that, in this family of mathematical models, concurrency is explicit. A large literature has tackled the analysis of concurrency in them, leading to several notions of distributability. Most of these rely on the notion of state machine. Indeed, structural analysis is a well-known set of techniques that allow, in particular, for the identification of the state machines of a net. A state machine is a part of the net, or subnet, which is fully sequential, it depicts no concurrency. In this last sense, it can be identified with a subclass of finite state automata. A natural approach for studying how the system can be distributed is to try to allocate each of these state machines to a different spatial location. The way state machines are interconnected would then depict the way they communicate. A different approach for allocating parts of the system to spatial locations is developed in [3], or [16]. However, that line of research presents, in the view of the author, the drawback that the mode of communication among the components is limited. Indeed, parts of the system assigned to different locations can only communicate by exchanging messages. Although accurately representing the reality, this approach imposes limitations when designing a system.

The approach adopted in this work is different. Instead of trying to assign parts of the system to spatial locations, the notion of observation is exploited.

The systems will be decomposed in possibly overlapping parts, that will be called sequential components. A sequential component is a part of the system which is observable from one spatial locality. Hence, a sequential component not only includes the set of local states and instructions allocated to a point in space, it extends to the parts of the system it is able to obtain information from. In this sense, two sequential components are allowed to overlap, on a part of the system they agree upon. This shared part of the system is to be interpreted as their communication protocol, and in fact, this approach permits to interpret distributability so as to consider a wide range of communication modes. The building blocks for these modes will be handshaking, the simultaneous execution of an action from several parts, and communication channels, implemented as shared memory. By composing these basic features, more complex protocols can be implemented. In this setting, a wider range of designs can be interpreted as distributable. This approach is characterised by the fact that the notion of observation, and observability, are central.

The notion of observation, and observability, is an important matter of study, in particular, in physics, where the design and interpretation of experiments is crucial. In particular, in quantum mechanics, these notions are formalised in terms of logic. A property of a physical system is, as in computer science, a statement about the system which can hold at a given instant, or not. This is formalised as a Boolean variable that can take the value true or false. The nature of the system then imposes restrictions on the dependencies among these variables, generating structures that express these dependencies. A property is said to be observable if it can be observed in the system without interfering with its behaviour. This idea will be motivated and developed in Section 4.1.2.

In order to apply this notion of observability, this work restricts its focus within the class of Petri net models to the elementary and condition/event case. Indeed, the behaviour of these models is guided by Boolean conditions, variables that take the value true or false. In these models such variables represent the local states of the system, in such a way that the value of a collection of local states determines the global state the system is at. This allows one to interpret observable properties as potential local states, states that if not present, could be added to the system without altering its behaviour. The theory of Petri net synthesis (see for example [2]) allows for identifying all such redundant states.

Hence, this work is guided by two main principles. The first one comes from the field of Petri nets, and was postulated by Petri himself [48]. An event is a component of the model which represents an action of the system. The principle of extensionality states that the events of a system are characterised

by their effect. An event is only observable by the way it modifies the system, and so two events applying the same modifications on it should be considered the same. The second principle that serves as a guideline in this work, is what is here called the principle of locality. It postulates that remote observations can only be performed through communication. From a given location, the acquisition of information about a remote part of the system can only be performed by communicating with it, thus imposing that the remote part alters its state. Hence, the observations performed by a sequential component on another must be integrated in the part of the system which is shared by both, their communication protocol.

1.4 Outline of the Thesis and its Main Contributions

This work is structured as follows. Chapter 2 will introduce the basic mathematical formalism regarding the models at stake. After briefly introducing some general notions of Petri net theory, the focus is put on elementary net systems, and condition/event net systems. The nonsequential processes that these systems can run will be formalised as causal nets. Labelled transition systems will then be introduced as interleaving models for the behaviour of such net systems, so a particular attention will be put on elementary and condition/event transition systems. Finally, orthomodular partial orders, and orthomodular lattices will be introduced as the main tools for analysing these systems, and their processes, respectively.

Chapter 3 will deal with process distributability. After motivating the formalisation of processes as causal nets, it will present the mathematical instrument introduced in [10]. It is a tool that permits to analyse a process by endowing it with an algebraic structure. This structure is composed of subsets of the causal net, interpreted as subprocesses, together with a set of relations among them, and forms an *orthomodular lattice*. This lattice carries information about concurrency of the process, and the causal dependences among its parts.

The first contributions of this work will then be presented. The properties of this orthomodular lattice are exploited so as to extract a canonical abstraction of the process which carries all information regarding its concurrency and its causal dependences. In particular, the concurrency relation among the minimal non-trivial elements of this lattice is shown to be sufficient to determine the whole structure. As a consequence, the relation between the subprocesses corresponding to these elements are sufficient to determine how

the process can be distributed. This novel canonical abstraction omits the information regarding the parts of the process for which the causal dependences described in its specification impose that they belong to the same spatial locality. The remaining information it depicts regards the way the process can be distributed in space, and the way information flows among the distinct parts. The introduction of this abstraction represents the contribution of this work regarding processes, most of which can be found in [1].

Chapters 4 and 5 will be concerned with system distributability. In Chapter 4, the notion of observable property of a system will be motivated. To this aim, the main results of elementary net synthesis are first presented, thus introducing the notion of region. These results are then used to justify that to every observable property of the system, there corresponds an elementary region. Since the observability of these properties is subject to the principle of locality, the structure of elementary regions, as introduced in [7], is presented as a suitable tool for analysing the distributability of the system. This structure, known as *quantum logic*, is formalised as an *orthomodular partial order*. The authors of that work described a synthesis procedure that provides a model of all possible behaviour implementable from a given specification of the properties of the system that should be observable, the so called *saturated transition system*. They further conjectured that no additional property can be observable on a system with a behaviour consistent with the one described in the synthesised model. In other words, each region of the synthesised system should correspond to an element of the orthomodular poset it was built from.

The construction of this saturated transition system will be presented, and its analysis will lead to the original contributions of this chapter. Based on the concurrency of its events, the saturated transition system will be endowed with a structure. Indeed, it will be shown that these events can be partially ordered, and an independence relation among them will be defined, based on their structural properties. It will be shown that this independence relation coincides with their pairwise commutativity, and can thus be interpreted as concurrency. This additional structure, provided to the synthesised system, allows to select a subset of events which is sufficient to depict all concurrency of the system. As a consequence, a system having this subset as events, will have the same regions as the saturated one. Hence, the contribution permits to further exploit the results of [7]. Most of the contribution in this line of research can be found in [15]

In Chapter 5, the problem of whether no additional property is observable on the saturated transition system is tackled. The problem is first discussed, and stated: an orthomodular poset is called stable when it is isomorphic to

the structured set of regions of its saturated transition system. The state of the art shows that when an orthomodular poset verifies a given collection of properties, then it embeds into the aforementioned set of regions. It is conjectured that, when it arises as the collection of regions of some elementary system, then this embedding is an isomorphism. In other words, regional orthomodular posets are stable.

Although the conjecture remains an open problem in the general case, it will be proven as an original contribution of this work, that the formalisation of observable properties must comply with an additional axiom, which is not required in the general case. This additional axiom is found to be a property of every regional orthomodular poset.

It is further shown, with this additional requirement, that the embedding of the orthomodular poset into the regions of its saturated transition system is in fact strong. This represents a step forward in proving the conjecture. Indeed, strong embeddings are closer to being isomorphisms in that their lack of surjectivity is narrowed down. In fact, under this assumption, the embedding must map each minimal element to a minimal region, so that no property whose extension is properly contained in such a region is observable on the saturated transition system.

Finally, it is shown for a few subclasses of structures of observable properties, that the conjecture is in fact true. Most of the results which constitute the contribution of this work in this chapter can be found in [13], and [14].

Chapter 2

Elementary Systems

In the analysis of a distributed system, it is assumed that a specification of the system, or of its behaviour, is provided in terms of some formal model. The fact that the system is distributed, imposes on such a model to express concurrency. Labelled Transition Systems, and Petri Net Systems, generalise Finite State Automata in very different ways. Indeed, the two models differ in that they represent concurrency with different formalisms.

Hence, the main difference between a Petri net and a finite state automaton is that actions, rather than labelling the arcs of the underlying graph, become vertices of their own, allowing one to represent transitions between the states in a much richer way. Indeed, arcs of a sequential automaton transit from one state to another, whereas actions in a Petri Net are allowed to have several states as input and several output states. The main consequence of this difference, is that states in a Petri net acquire a local nature. In some sense, the richer relational structure between actions and vertices spread global states of the system across vertices of the model. Given a Petri net system, a model of the processes it can run can be derived rather naturally. Actions executed along such a process are represented with their causal dependencies, which can be formalised as a partial order. A representation of the local states visited along the process allows to determine whether lack of causal dependency corresponds to concurrent occurrence, or mutual exclusion. Labelled transition systems as a formalism, remain closer to finite state automata. Different arcs carrying the same label are considered as different occurrences of the same action. In this case, concurrency of actions is not explicit, but is expressed by means of the so called interleaving semantics. In the elementary framework, two actions are considered concurrent whenever they commute. The order in which two concurrent actions occur is not relevant, in the sense that the two

corresponding sequences of occurrences will lead to the same state.

The evolution of an elementary system is guided strictly by Boolean conditions. This classical restriction on models of concurrent systems [57], will allow for a logical analysis of concurrency. Two main consequences of this restriction are to be considered. Regarding Petri net systems, the elementary framework imposes that local states adopt a Boolean nature, becoming variables that can only take values true or false. The set of global states in which a given local state holds the value true is its extension. The local states of a labelled transition systems will be uniquely characterised by their extension. As such they will be identifiable with the observable properties of the system.

This, in turn, implies that the processes executed by such a system are representable in such a way that the intermediary state between the occurrence of two causally dependent actions can be interpreted as a set of flags raised by the first action which allow for the occurrence of the second. The elementary setting, in particular, provides a logical interpretation both of local states of a system, and of a particular class of sub-processes. With the appropriate structure, there arise orthomodular partial orders. These models of logical propositions have been studied in certain fields of physics, in which, like in this work, the principle of locality plays an important role.

2.1 Elementary Net Systems, and Net Processes

Petri Nets are named after Carl Adam Petri, who introduced them as a formal tool for a theory of systems, and of information flow [46]. They are nowadays commonly used in fields ranging from industrial engineering to hardware design, and of course, theoretical computer science [58]. To suit the requirements of each field of application, several variants of the original formalism have been developed. Starting from the so called low-level nets, which keep the model to its simplest expression, several features can be added to handle clocks and delays, data types, and even variables and values by the means of guards. [44] provides a survey on the various types of formalisms, and [57] covers them in more detail. These extended models, the so called high-level nets, each representing an interesting matter of study on its own, widely exceed, however, the topic of this work. The focus will be therefore put on the simplest classes of Petri Nets, the so called Elementary net systems.

2.1.1 Petri Nets

Petri Nets have proven to be a powerful tool in the field of Computer Science. Indeed they allow for representation of both distributed systems, and the

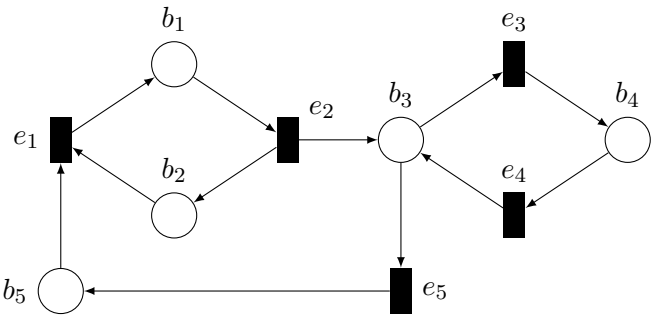

Figure 2.1: A Petri Net $N = (B, E, \mathcal{F})$. Conditions, in B, are represented by circles, and events, in E, by rectangles. The flow relation is represented by arrows going from conditions to events, or from events to conditions.

processes these are able to carry out. Rather than a model, they are a class of models with the common characteristic that the elements of the system which encode its state, its localities, are represented at the same level as the elements which modify its state, its actions. The usual terminology refers to these as conditions, and events respectively.

Definition 2.1.1 (Petri Net)**.** *A* Petri Net *is a tuple* $(B, E, \mathcal{F})$[1]*, where*
B *is the set of its* conditions
E *is the set of its* event
$B \cap E = \emptyset$*, no element is both a condition and an event, and*
$\mathcal{F} \subseteq (B \times E) \cup (E \times B)$ *is the* flow relation *expressing the causal dependencies between conditions and events.*

One advantage of Petri Nets is that they admit an intuitive graphical representation, that makes the model rather popular among engineers and system designers.

As depicted in Figure 2.1, conditions are commonly depicted by circles, whereas events are drawn as rectangles or squares. The flow relation is then simply a set of arcs linking conditions to events, or events to conditions. The flow relation only relates conditions to events, and events to conditions. The causal dependencies between events must be expressed by the means of localities, and events are the only allowed link between localities.

Given an event, the flow relation dictates to which conditions it is bound. The flow being oriented, one can distinguish the conditions it depends on from

[1] B stands for "Bedingungen", the German term for Conditions, and E stands for "Ereignisse", German for Events.

the ones that depend on it. Analogously, one can consider the sets of events which depend on a given condition, or which the condition depends on.

Definition 2.1.2 (Pre-set, Post-set)**.** *Given an event $e \in E$, its* pre-set*, or set of* pre-conditions *is the set of conditions which precede e according to the flow relation.*

$${}^\bullet e := \{b \in B \mid (b, e) \in \mathcal{F}\} \subseteq B$$

Symmetrically, the post-set*, or set of* post-conditions*, of e is the set of conditions which depend on e.*

$$e^\bullet := \{b \in B \mid (e, b) \in \mathcal{F}\} \subseteq B$$

Given a condition $b \in B$, its pre- and post-sets are defined analogously.

${}^\bullet b := \{e \in E \mid (e, b) \in \mathcal{F}\} \subseteq E$ *are the* pre-events *of b, and*

$b^\bullet := \{e \in E \mid (b, e) \in \mathcal{F}\} \subseteq E$ *are its* post-events*.*

$\forall x \in B \cup E : \nu(x) := {}^\bullet x \cup x^\bullet$ *is called its* neighbourhood

In the original view of Petri, elements of a net are characterised by their extension [48]. This notion, called *principle of extensionality*, states that an event in itself is not observable, only its effect on a system is. The effect of an event is, as it will be seen in the next section, formalised by means of its neighbouring conditions. According to this principle, two events with the same pre-, and post-conditions could not be distinguished. Conversely, a condition should be characterised by its contribution to the states of the system. If two conditions have the same pre-, and post-events, then whatever effect of this that applies to one, will also apply to the other. Then these two conditions will always provide the same information regarding the states of the system. Again, according to the principle of extensionality, they should be the same element.

Although in many Petri net models, this principle has been dropped for the sake of versatility, it is required to hold in the elementary setting. As a matter of fact, the principle of extensionality plays an important role in the development of the theory which is the focus of this work, and the reader will encounter it throughout the chapters. The consequences that concern this section are of structural type. In particular, the Petri nets to be considered in this work will fulfil the following properties.

Definition 2.1.3 (Simple Petri Net)**.** *A Petri net $(B, E, \mathcal{F})$ is said to be* simple*, whenever the flow relation distinguishes any two different elements.*

$$\forall x, y \in B \cup E : ({}^\bullet x = {}^\bullet y \text{ and } x^\bullet = y^\bullet) \rightarrow x = y$$

A condition $b \in B$ and an event $e \in E$ are said to form a self-loop, whenever both $(b, e) \in \mathcal{F}$ and $(e, b) \in \mathcal{F}$. A Petri net without self-loops is said to be *pure*. This notion can be formalised as follows.

Definition 2.1.4 (Pure Petri Net)**.** *A Petri net* $(B, E, \mathcal{F})$ *is said to be* pure*, whenever any condition, or event has disjoint pre- and post-sets.*

$$\forall x \in B \cup E : {}^\bullet x \cap x^\bullet = \emptyset$$

Pureness follows, in the elementary frame, from the extensionality principle, in that an event involved in a self-loop would have no observable effect.

The most interesting characteristic of these models in the analysis of either systems or processes, is that the causal dependencies between the possible actions are strictly expressed by the means of conditions. If an action must be carried out before another, then there must be a condition expressing this dependency. Such a dependence relation is commonly interpreted as a flow of information, that should be stored at a point in space. Symmetrically, if in the execution, two conditions are required to exchange information, then there must be an action expressing this exchange.

2.1.2 Elementary Net Systems

Elementary Net Systems [57], or ENS, represent a suitable paradigm to study the logical structure of observable properties of distributed systems. As a class of Petri Nets, they explicitly represent concurrency, and as elementary systems, their local states are interpretable as Boolean variables, they take values true or false. They have indeed shown to be useful in the design of digital circuits, and hardware in general [23].

Definition 2.1.5 (Elementary Net System)**.** *An* Elementary Net System *is a tuple* $(B, E, \mathcal{F}, m_0)$
such that
$(B, E, \mathcal{F})$ *is a pure and simple Petri net, and*
$m_0 : B \to \{0, 1\}$ *is the* initial marking*.*
A marking *of the system is any map of the kind* $m : B \to \{0, 1\}$*.*

Markings are represented graphically, by assigning tokens to conditions. Given a marking m, a condition b of the net, represented by a circle, will contain a token if $m(b) = 1$, and none when $m(b) = 0$, as in Figure 2.2. Markings represent a global state of the system, and will often be be referred to as states.

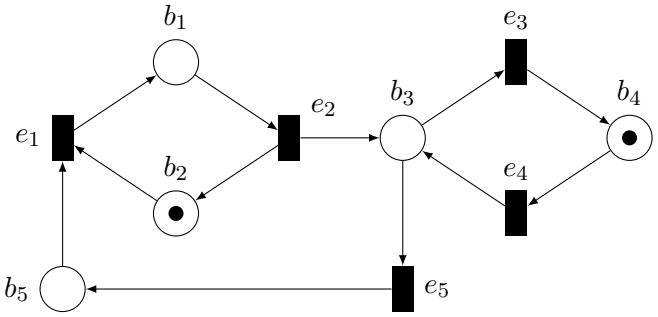

Figure 2.2: An elementary net system with the underlying net of figure 2.1. The represented marking assigns 1 to b_2 and b_4, and 0 to all other conditions. It can be seen as the set $\{b_2, b_4\}$.

Remark 2.1.1. *A binary function $f : X \to \{0,1\}$ can be seen as a subset of X. Indeed, its support $\boldsymbol{supp}(f) := \{x \in X \mid f(x) \neq 0\} \subseteq X$, corresponds precisely to the elements of X it assigns 1 to. Given an arbitrary subset S of X, one can define its characteristic function $f_S : X \to \{0,1\}$ as:*

$$f_S(x) := \begin{cases} 0 & x \notin S \\ 1 & x \in S \end{cases} \tag{2.1}$$

Then trivially $S = \boldsymbol{supp}(f_S)$, and $f \equiv f_{\boldsymbol{supp}(f)}$. This duality will be exploited throughout this work, and binary functions will be commonly referred to as their supports.

In particular, a marking of an elementary net system will often be viewed as the subset of conditions it assigns the value true to. Equivalently a state will be identified with the set of conditions that hold at it.

Events represent the actions of the system. Conditions represent, when they are marked, local states of the system. Conditions are to be understood as localities, which participate in a global state, when the corresponding marking contains it.

At a given marking, some events are allowed to *fire*. The firing of an event represents the execution of the corresponding action, which leads the system from one state to another. The firing of events describes the dynamics of the system. However, these dynamics are guided by the marking of conditions, that dictate which events can fire at a given state. When a marking allows for the firing of an event, it said to enable it. In order to do so, it must assign a token to each of its pre-conditions, and no tokens to any of its post-conditions.

Definition 2.1.6 (Enabled Event)**.** $(B, E, \mathcal{F}, m_0)$ *be an elementary net system. A marking* m enables *an event* e*, denoted* $m[e\rangle$*, whenever*

$$\forall b \in {}^\bullet e : m(b) = 1 \text{ and } \forall b \in e^\bullet : m(b) = 0$$

A situation in which $\forall b \in {}^\bullet e : m(b) = 1$, but $\exists b \in e^\bullet : m(b) = 1$ is called a *contact*. Contacts will be relevant when considering processes of a system, they are characterised by the fact that an event is not enabled, although all its pre-conditions are true.

When an event is enabled, it is allowed to fire, yielding a new marking m'

Definition 2.1.7 (Elementary Firing Rule)**.** *The* elementary firing rule *allows to compute a new marking* m' *obtained from firing an enabled event* e*, at a marking* m*.*

$$m[e\rangle m' \Leftrightarrow m'(b) := \begin{cases} 0 & b \in {}^\bullet e \\ 1 & b \in e^\bullet \\ m(b) & \textit{otherwise} \end{cases}$$

In this setting, a condition will be said to hold, its value to be true, at a given state, whenever the corresponding marking assigns a token to it. A state is then simply a truth assignment to the conditions of the system. In this way, an event will be enabled whenever all conditions in its pre- set hold, and none of the conditions in its post-set do. Upon the firing of an event, all its pre-conditions cease to hold, and all its post-conditions become true. According to the extensionality principle, this is the observable effect of an event, meaning that each event e should be identified with the pair of sets $\langle {}^\bullet e, e^\bullet \rangle$. Also, according to the firing rule, if there is a condition belonging both to the pre- and post-sets of an event, then that event can never fire. Indeed, the condition, either marked or unmarked, prevents the event from firing. Such an event is said to be *dead*, providing no behaviour to the system. As a matter of fact, such an event would have no observable effect. This justifies, following the extensionality principle, the requirement that the underlying Petri nets of elementary net systems are pure and simple. In this work, no dead event will be considered a part of the system.

Net systems are a dynamic model, in the sense that only one marking is represented at a time. In order to see the full behaviour, one must let it run. Graphically, this makes sense, since a marking is determined by distributing tokens in the corresponding conditions. Therefore, the behaviour of the system can only be analysed by letting it run. This is done by iteratively selecting enabled events, or events, and letting them fire, leading the system from one

state to another. Such a sequence of occurrences of events is called a *firing sequence.*

Definition 2.1.8 (Firing Sequence)**.** *A* firing sequence *is a finite sequence* $(e_i)_{i\leq n}$, $\forall i \leq n : e_i \in E$, *such that for every* i *there are two markings* m *and* m' *satisfying* $m[e_i\rangle m'$ *and* $m'[e_{i+1}\rangle$ *When* $m_0[e_1\rangle$, *the firing sequence is said to be initial.*

In a firing sequence, a given event can be fired several times, according to the previous definition, there could be $i < j \leq n$ such that $e_i = e_j$. This is why it is considered, rather than a sequence of events, as a sequence of their occurrences. Indeed, an event is a structural element of the system, it is its occurrence, or firing, which has a dynamic effect. Thus, conditions and events are the structural elements of the systems, whereas its dynamics will be guided by markings, and firings. A state, or marking, is said to be reachable, if the system can eventually visit it along its run.

Definition 2.1.9 (Reachable Marking)**.** *A* marking is reachable *if there exists a firing sequence that leads to it. Formally, a marking* m_n *is reachable from* m *whenever there exist a sequence of markings* $(m_i)_{1\leq i\leq n}$, *and a firing sequence* $(e_i)_{i\leq n}$ *such that*

$$m[e_1\rangle, \text{ and } \forall i \leq n : m_{i-1}[e_i\rangle m_i \tag{2.2}$$

An event is said to be dead *at a marking* m *if no marking reachable from* m *enables it. Throughout this work, elementary systems will be assumed to have no dead events at the initial marking.*

Given a marking m the set of all reachable markings from m is denoted by M. One can consider the reachability relation $R := \{(m, m') \in (\{0,1\}^B) \times (\{0,1\}^B) \mid m' \text{ is reachable from } m\}$. Note that if three markings m_1, m_2, m_3 are such that m_2 is reachable from m_1, and m_3 is reachable from m_2, then there must be a firing sequence leading from m_1 to m_3. Hence, R is a transitive relation.

It is worth noting at this point, that the initial marking is key to determine the evolution of the system. Indeed a same Petri Net $(B, E, \mathcal{F})$ can lead to different net systems, whenever the initial state m_0 is chosen differently. In fact, different initial markings usually lead to different sets of reachable states M.

Example 2.1.1. *Figure 2.2 represents a system* $N_1 = (B, E, \mathcal{F}, m_0)$, *with the underlying net* N *of Figure 2.1. Its initial marking is* $m_0 = \{b_2, b_4\}$. *The event* e_4 *is enabled at* m_0, *since* ${}^\bullet e_4 = \{b_4\} \subseteq m_0$, *and* $e_4^\bullet \cap m_0 = \{b_3\} \cap m_0 = \emptyset$.

Then system could follow the execution:
$m_0[e_4\rangle\{b_2, b_3\}[e_5\rangle\{b_2, b_5\}[e_1\rangle\{b_1\}[e_2\rangle\{b_2, b_3\}[e_3\rangle\{b_2, b_4\}$. *Then*
$(e_4, e_5, e_1, e_2, e_3)$ *is a firing sequence, and* $\{b_2, b_3\}$, $\{b_2, b_5\}$, $\{b_1\}$, *and* $\{b_2, b_3\}$ *are reachable markings.*

Consider the elementary net system consisting of the Petri net of Figure 2.1, with initial marking $m'_0 = \{b_5, b_2, b_4\}$, $N_2 = (B, E, \mathcal{F}, m'_0)$. *In this system,* $(e_4, e_5, e_1, e_2, e_3)$ *is not a firing sequence, since after firing* e_4, *the marking* $\{b_5, b_2, b_3\}$ *contains a post condition of* e_5, *and so it is not enabled. This system is different from* N_1.

However, two different initial markings could eventually lead to the same state. This state would belong to the intersection of the sets of markings, reachable from each initial state. In order to characterise the possible behaviours bound to the structure of a Petri net, it would be suitable to consider disjoint sets of reachable markings.

This leads to a different class of systems, in which backward firing is also considered.

Definition 2.1.10 (Backward Enabled Event)**.** $(B, E, \mathcal{F}, m_0)$ *be an elementary net system. An event* e *is* backward enabled *by a marking* m, *denoted* $[e\rangle m$, *when* m *marks all post-conditions of* e, *and none of its pre-conditions.*

$$\forall e \in E : [e\rangle m \Leftrightarrow (\forall b \in {}^\bullet e : m(b) = 0) \wedge (\forall b \in e^\bullet : m(b) = 1)$$

Definition 2.1.11 (Backward Firing)**.** *The* backward firing *of an event leads to another marking* m'

$$m'[e\rangle m \Leftrightarrow m'(b) := \begin{cases} 0 & b \in e^\bullet \\ 1 & b \in {}^\bullet e \\ m(b) & otherwise \end{cases}$$

Backward firing allows to properly classify system behaviour in terms of sets of reachable markings. Indeed, when one considers both backward and forward reachability, the corresponding relation on markings is not only transitive, but also symmetric. By assuming every state to be reachable from itself, one thus obtains an equivalence relation of reachability. The classes of equivalence for this relation, form a partition of the set of all possible markings $\{0, 1\}^B$. Each marking belongs to exactly one class.

Remark 2.1.2. *Consider a set* X. *An equivalence relation on* X *is a binary relation* $\sim \subseteq X \times X$ *which is*

1. *reflexive,* $\forall x \in X : (x, x) \in \sim$
2. *symmetric,* $\forall x, y \in X : (x, y) \in \sim \rightarrow (y, x) \in \sim$
3. *transitive,* $\forall x, y, z \in X : ((x, y) \in \sim \wedge (y, z) \in \sim) \rightarrow (x, z) \in \sim$

The sets $[x]_\sim := \{y \in X \mid (x, y) \in \sim\}$ *are called* equivalence classes *of* $\sim$*. They form a partition of* X*, every* $x \in X$ *is in some class of equivalence, and these are pairwise disjoint.*

A partition *of* X *is a collection of subsets* $P \subseteq 2^X$*, such that they are pairwise disjoint, and their union is the whole set, formally*

1. $\forall S_1, S_2 \in P : S_1 \cap S_2 = \emptyset$
2. $\bigcup_{S \in P} S = X$

The set of all equivalence classes is called the quotient $X/\sim$*.*

This consideration allows to define a net system which presents the same advantages as elementary net systems, regarding the scope of this work.

Definition 2.1.12 (Condition/Event Net System)**.** *A* condition/event net system*, is a tuple* $(B, E, \mathcal{F}, M)$*, such that* $(B, E, \mathcal{F})$ *is a pure and simple Petri net, and* M *is a class of markings for the relation of backward and forward reachability. Furthermore,* $\forall e \in E : \exists m \in M : m[e\rangle$*.*

The dynamics of condition/event net systems are guided by both the backward and the forward firing rule.

Condition/event net systems will be useful in the following, since they present the same advantages as elementary net systems, and allow to leave reachability considerations aside. Indeed, by considering the full class of markings, it is ensured that these will be backward, or forward reachable.

As a classes of Petri net systems, both elementary net systems and condition/event net systems present the desired properties regarding concurrency in the system. Indeed, concurrency in this frame can be reduced to a structural property called *independence*.

Definition 2.1.13 (Concurrency)**.** *Two events* e_1 *and* e_2 *are said to be independent whenever they share no pre- or post-conditions, namely*

$$({}^\bullet e_1 \cup e_1^\bullet) \cap ({}^\bullet e_2 \cup e_2^\bullet) = \emptyset \tag{2.3}$$

Whenever two events are independent, and there is a marking m *which enables both of them,* $m[e_1\rangle$ *and* $m[e_2\rangle$*, they are said to be* concurrent*.*

This structural property makes this model rather comfortable for the study of distributed systems.

Note that a marking may enable two events without them being concurrent. The firing of one might lead to a marking which does not enable the other any more. In this case, it is said that the two events are in *conflict*. Of course, for the occurrence of an event to have disable another, either the first consumed a token from the pre-conditions of the second, or it produced a token to one of its post-conditions.

Definition 2.1.14 (Conflict)**.** *Two events e_1 and e_2 are in* conflict *whenever at least one of the following conditions hold:*

1. $^\bullet e_1 \cap {}^\bullet e_2 \neq \emptyset$, *(backward conflict)*

2. $e_1^\bullet \cap e_2^\bullet \neq \emptyset$, *(forward conflict)*

A Petri net without conflicts is called conflict-free*. The class of conflict-free is characterised as the set of Petri nets $(B, E, \mathcal{F})$ which satisfy $\forall b \in B : |^\bullet b| \leq 1 \wedge |b^\bullet| \leq 1$.*

When two events are in conflict, and there is a reachable marking m which enables both $m[e_1\rangle$, and $m[e_2\rangle$, they are said to constitute a choice*.*

Example 2.1.2. *With reference to Figure* **??**, *$^\bullet e_1 = \{b_2, b_5\}$, and $e_1^\bullet = \{b_1\}$, so $\nu(e_1) = \{b_1, b_2, b_5\}$. $\nu(e_4) = \{b_3, b_4\}$, and so e_1 and e_4 are independent. Consider the systems N_1 and N_2 from Example 2.1.1. In N_1, no reachable marking enables both e_1 and e_4. Although they are independent, they are not concurrent in N_1. In N_2, the initial marking $m_0' = \{b_5, b_2, b_4\}$ enables both e_1, and e_4. Hence e_1 and e_4 are concurrent in N_2, (e_1, e_4), and (e_4, e_1) are both valid firing sequences, and they lead to the same marking $\{b_1, b_3\}$. This marking presents a contact. e_2 is not enabled at it, although $^\bullet e_2 \subseteq \{b_1, b_3\}$. The condition b_3 creates structural conflict. e_3 and e_5 are in backward conflict since $^\bullet e_3 \cap {}^\bullet e_5 = \{b_3\} \neq \emptyset$. The marking $\{b_1, b_3\}$ enables both of them, but the firing of one disables the other, by consuming the token in $\{b_3\}$, and so e_3 and e_5 are in choice at $\{b_1, b_3\}$. Similarly, e_2 and e_4 are in forward conflict, $e_2^\bullet \cap e_4^\bullet = \{b_3\} \neq \emptyset$. At the marking $\{b_1, b_4\}$ both can fire, but the firing of one would disable the other by contact.*

One of the main advantages of Petri net systems, is that they allow for structural analysis. In this work, it will be presented with the scope of identifying the sequential components of a system.

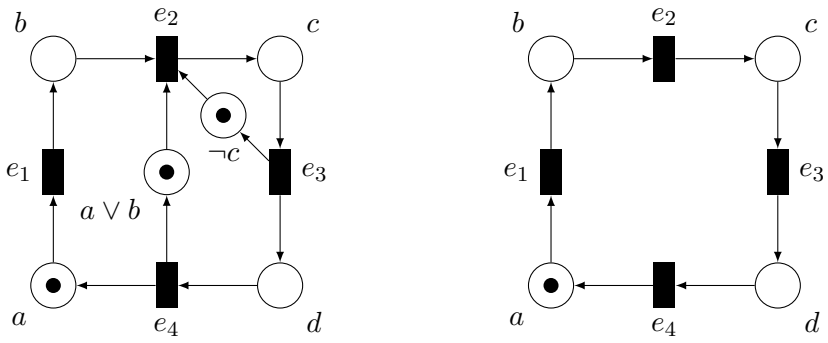

Figure 2.3: To the left, the elementary net system N_3 has sequential behaviour. To the right, N_4 is a subsystem of N_3, which is a state machine.

Definition 2.1.15 (State machine)**.** *A Petri net $N(B, E, \mathcal{F})$ is called a* state machine *whenever it's underlying graph is connected, and*

$$\forall e \in E : |{}^\bullet e| = 1 \text{ and } |e^\bullet| = 1$$

A condition/event net system $N = (B, E, \mathcal{F}, M)$ is called a mono-marked state machine*, when it is a state machine, and its conditions are* mutually exclusive*, namely*

$$\forall m \in M : \exists b \in B : m = \{b\}$$

This definition transfers to elementary systems by considering M as the set of reachable markings. In a mono-marked state machine, all conditions are mutually exclusive such that each marking selects exactly one of them.

A mono-marked state machine is fully sequential. No two of its events can fire concurrently. With the elementary definition, state machines coincide with the class of deterministic finite state automata that have no repeated labels.

This definition will help localising the sequential parts of a system. These parts will be called *sequential components*.

Definition 2.1.16 (Generated Subnet, Subsystem)**.** *Let $N = (B, E, \mathcal{F}, M)$ be a condition/event net system. Given a collection $S \subseteq B$ of conditions, consider the union of their neighbourhoods $E_S := \bigcup_{b \in S} \nu(b)$. Then the* subnet generated by S *is defined as*

$$N(S) = (S, E_S, \mathcal{F}{\restriction}_{(S \times E_S) \cup (E_S \times S)})$$

Consider $M_S = \{m\restriction_S |\ m \in M\}$*, then the* subsystem generated by S *is*

$$N(S) = (S, E_S, \mathcal{F}\restriction_{(S\times E_S)\cup(E_S\times S)}, M_S)$$

A state machine component *of N is a subsystem of N which is a state machine.*

The following is a well known result. It is here restated for the condition/event case.

Proposition 2.1.1 (State Machine Component)**.** *Let $N = (B, E, \mathcal{F}, M)$ be a condition/event net system. Let $S \subseteq B$ be a collection of mutually exclusive conditions, such that*

$$\forall m \in M : |m \cap S| = 1$$

Then the subsystem generated by S is a mono-marked state machine.

Definition 2.1.17 (State Machine Decomposable Net System)**.** *A condition/event, or elementary, net system is said to be* state machine decomposable*, when all its conditions are contained in some subset which generates a state machine subsystem.*

Example 2.1.3. *With reference to Figure 2.3, in the net system N_3 (left), the conditions $\{a, b, c, d\}$ are mutually exclusive, no reachable marking selects more than one. The system N_4 (right), is the subsystem of N_3 generated by $\{a, b, c, d\}$. Each of its events has only one pre-condition, and one post-condition, so it is a state machine. N_4 is a state machine component of N_3. Since each reachable marking has exactly one condition, it is mono-marked.*

The set $\{a \vee b, c, d\}$ generates a subsystem with only $\{e_2, e_3, e_4\}$ as events, it is also a state machine component of N_3.

Finally, $\{c, \neg c\}$ generates another state machine component with events $\{e_2, e_3\}$. Each condition of N_3 is contained in some of its state machine components, so it is state machine decomposable.

N_3 has sequential behaviour, his firing sequences are all of the sort $(e_1, e_2, e_3, e_4, e_1, \ldots)$. It consists of a single sequential component.

Note that a system may be state machine decomposable without being a state machine in itself. An event of a state machine component may have more than one pre-, or post- condition in the ambient system. However, the fact that the conditions of the component are mutually exclusive restricts the behaviour of its events. These can never fire concurrently.

Consequently, state machine components are interpreted as belonging to the same sequential components.

Remark 2.1.3. *The notion of distributability has already been addressed in this formal context. One of the most relevant approaches, to the knowledge of the author, is that of [3]. The notion of distributability proposed in that work, differs from the one presented here, in the following two main aspects.*

1. *Every element of the net model must belong to exactly one location.*
2. *Every conflict must be solved locally: two conflicting events must belong to the same location, as all their pre-conditions.*

Intuitively, this approach interprets all modes of communication as simple message passing. The reader is referred to [16] for further details.

In the view proposed in this work, components are allowed to share elements. This permits to represent basic interactions as handshaking, shared local states (as a model of shared memory). With these fundamental modes of interaction, more complex means of communication can be modelled, such message passing. In this case, the communication channel must be explicitly represented as one component of the system.

The main advantage this work will take of considering elementary and condition/event net systems, is that they allow for a Boolean interpretation. The binary restriction on the markings allows for only two values for each condition, zero or one, these will be intended as false and true. Conditions can be seen as Boolean variables. Hence, this paradigm represents a suitable framework for the study of the testable propositions on a distributed system.

The following example should give an idea of the study that will be carried out.

Example 2.1.4. *In the net system N_3 of Figure 2.3, the condition $a \vee b$ is marked exactly when either a is marked, or b is marked. At each reachable marking m, its truth value is determined by $m(a \vee b) = m(a) \vee m(b)$.*

Analogously, the condition $\neg c$ is marked exactly when c is not, its truth value is determined, at each reachable marking m, by $m(\neg c) = \neg m(c)$.

These two conditions are redundant regarding the behaviour of the system, since N_3, and N_4 admit the same firing sequences

2.1.3 Causal Nets as Partial Orders

A process can be represented by means of a Petri Net [47, 59]. However, in this case, no tokens, and hence no markings, are considered. As a matter of fact, events are not to be considered as actions any more, but rather as occurrences of actions. Indeed, in a net system, an event represents a single action that can

occur several times during the run of the system. When considering processes however, each event represents one single occurrence of the corresponding action, and so to each action there may correspond several events. Conditions in a process are to be interpreted analogously. They represent the occurrence of local states. In this way, the flow relation will now express the causal dependencies between the different occurrences of the elements of the system, and since each new occurrence will lead to a new element, the whole structure will contain no cycles, it forms a partial order [17].

A process corresponds to the record of a possible behaviour of a system. The system might execute one action several times along a process. The sequential execution of a process is a total ordering of the actions that compose it. Just like a string accepted by a finite state automaton takes symbols from an alphabet, the process run by a net system takes symbols from its set of events, and conditions. Thus, symbols are allowed to be repeated along a process. Among the different generalisations of strings accepted by automata, meant to include concurrency ([42],[45, 62]), the present work will focus on causal nets.

Causal nets are a class of Petri nets, which are suitable for representing the processes of Petri net systems, and in particular of elementary systems. In fact, given an elementary net system, its processes can be derived from its execution. This matter will be covered

Strings are total orderings of action occurrences. This order of actions can be thought of as marked by a clock. After each tick, one single action is allowed to occur. In a distributed setting however, the lack of global clock may prevent an observer from knowing which of two remote actions actually occurred first. In causal nets, time is replaced by a weaker notion, causality. Causality, just like time, is formalised by an order relation.

Definition 2.1.18 (Order Relation)**.** *An* order relation *or* ordering *of a set* X *is a binary relation* $\leq \subseteq X \times X$ *which is*

1. *reflexive,* $\forall x \in X : (x, x) \in \leq$
2. *antisymmetric,* $\forall x, y \in X : (x, y), (y, x) \in \leq \rightarrow x = y$
3. *transitive,* $\forall x, y, z \in X : (x, y), (y, z) \in \leq \rightarrow (x, z) \in \leq$

The order is total *whenever* $\forall x, y \in X : x \leq y$ *or* $y \leq x$

In a sequential system, only one action can be performed at a time. Hence, time is a total ordering, each action must happen before, or after, any other

action. Causal dependence is weaker in this sense. Causally independent actions remain unordered, the underlying order is partial.

Definition 2.1.19 (Partially ordered set)**.** *A* partially ordered set*, or* poset*, is a set equipped with an order relation $(P, \leq)$, which need not be a total ordering.*

Given a subset $S \subseteq P$, its up-set is the set of elements of P which greater or equal to some element of S,

$$\uparrow S : \{x \in P \mid \exists y \in S : y \leq x\}$$

Analogously, the down-set of S is the set of elements of P less or equal to some element of S,

$$\downarrow S : \{x \in P \mid \exists y \in S : x \leq y\}$$

In a partial order, the notion of minimum *may not correspond to a single element. It is defined so as to provide a subset of elements which are not greater or equal to any other element. Let $S \subseteq P$, then*

$$\min(S) := \{x \in S \mid \forall y \in S : \neg(y \leq x)\}$$

Analogously, the maximum *is defined as*

$$\max(S) := \{x \in S \mid \forall y \in S : \neg(x \leq y)\}$$

Given two partial orders $(P, \leq)$, and $(P', \leq')$. Then $(P', \leq')$ is a subposet *of $(P, \leq)$ whenever there is an injective morphism $\phi : P' \hookrightarrow P$ such that it preserves the order*

$$\forall x, y \in P' : x \leq' y \to \phi(x) \leq \phi(y)$$

It may be required that the order is also reflected,

$$\forall x, y \in P' : \phi(x) \leq \phi(y) \to x \leq' y$$

In this case, the morphism is called an order embedding*, and P' is called an* induced subposet *of P. An induced subposet can be defined as $P' \subseteq P$, with $\leq' := \leq\restriction_{P'}$.*

Instead of representing the fact that an action occurred before another, causality expresses the fact that for an action to happen, the occurrence of another is required. Hence, this order expresses causal dependency. By convention, an element always depends causally on itself. Causality shares some

properties with time. If one action occurs before another, and this second action before a third, then certainly the first will have occurred before the third. Causality is also taken to be transitive, in the sense that in a chain of three causally dependent occurrences, the third action requires the occurrence of the first one, in order to occur. Note that, if the occurrence of the third action is required for the occurrence of the first one, then none of the chain can effectively happen. Such a behaviour can never be observed, and so, out of the extensionality principle, the causality relation is taken antisymmetric.

When representing causal dependence by means of a Petri net, one must ensure that the flow relation generates an order. The flow relation only represents the direct involvement of conditions in the occurrence of actions, and so one must extend it to represent transitive causality.

Definition 2.1.20 (Transitive Closure)**.** *Let $R \subseteq X \times X$ be an arbitrary binary relation on a set X. The* transitive closure *R^+ of R the smallest binary relation on X such that*

1. $R \subseteq R^+$
2. $\forall x, y, z \in X : (x, y), (y, z) \in R^+ \rightarrow (x, z) \in R^+$

It is the intersection of all transitive relations which contain R.

Conditions express the fact that a given action has occurred, and so the actions which depend on it are allowed to occur themselves. Note that, just like a tick of a clock happens only once, one must distinguish the occurrence of a condition from the condition itself. In a causal net, a condition is not understood as a variable that can take the value true or false. It rather represents the fact that a given variable takes the value true along an execution of the system. It is the occurrence of a condition. If along the execution, a given condition ceases to hold, and becomes true again, the fact that it is true will be represented twice along a process.

In order to interpret the lack of ordering as causal independence, one must pay attention to the following fact. Suppose, in an elementary system, that a marked condition enables two events. The occurrence of any two of the events will lead to a marking which does not contain the condition, thus disabling the other event. It is said they constitute a choice. These two occurrences would not be causally independent, in the sense that the occurrence of one does depend on whether the other has fired or not. However, the flow relation does not represent this dependence. One could consider both occurrences and still obtain an order from the flow. A symmetric situation arises when two events

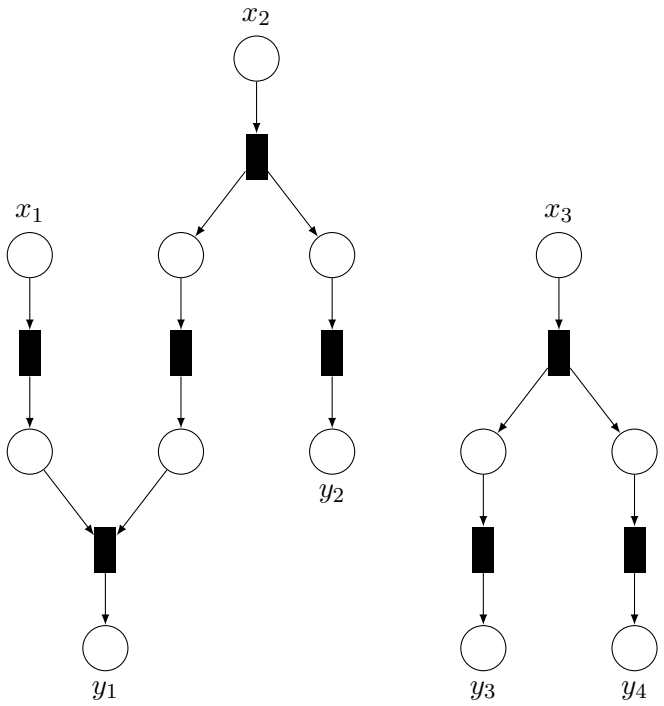

Figure 2.4: A causal net

share a post-condition. Note that a process represents one single execution of the system, and so in such a situation, only one of the two occurrences is to be represented. Thus, the class of causal nets will not allow such situations, they are conflict-free.

Definition 2.1.21 (Causal net)**.** *A* causal net *is a conflict-free Petri net, such that the transitive closure of its flow relation is an order.*

Causal nets will be the basic structures underlying processes. However, some additional restrictions should be added. In particular, processes will formally carry the information on how they relate to the net system they are an execution of. Hence, causal nets do not represent processes in themselves, but as part of their formalisation, they allow for the appropriate interpretation of the following notions regarding partial orders.

Definition 2.1.22 (Relations on a poset)**.** *Let* $(P, \leq)$ *be a partially ordered set. Two elements* $x, y \in P$ *are said to be in* li relation*,* x **li** y*, whenever*

either $x \leq y$ or $y \leq x$. The li relation is the symmetric closure of the order

$$\mathbf{li} = \leq \cup \leq^{-1}$$

Two elements $x, y \in P$ are said to be concurrent*, x **co** y, whenever neither $x \leq y$ nor $y \leq x$. The concurrency relation is the complement of the li relation*

$$\mathbf{co} = P^2 \setminus \mathbf{li}$$

Note that **li** *is reflexive and symmetric, but does not need to be transitive. Similarly,* **co** *is irreflexive, and symmetric, but does not need to be transitive.*

Partial orders will arise in many different settings in this work, and it will sometimes be convenient to distinguish whether a given relation corresponds to one, or another. In these cases, $\leq_P$, $\mathbf{li}_P$, and $\mathbf{co}_P$ will denote that the relations are taken in the poset P.

In what follows, it will be assumed that the partial order $(P, \leq)$ is obtained from a causal net $N' = (B', E', \mathcal{F}')$, as $P = B' \cup E'$ and $\leq = \mathcal{F}^+$. Two elements are in line when there is some causal dependence between them, independently from the orientation of it. Thus, two elements will be concurrent when they are causally independent. It is interesting to extend these notions to subsets of the partial order.

Definition 2.1.23 (Lines and Cuts)**.** *Let $(P, \leq)$ be a partially ordered set.*

A li-set *is a clique of* **li** *in P, namely a subset $S \subseteq P$ which satisfies that $\forall x, y \in S : x$ **li** y.*

A line *is a maximal li-set, namely a li-set S such that adding any element to it would violate the above condition. Formally, this reads as $\forall y \notin S : (\exists x \in S : x$ **co** $y)$*

Analogously, a co-set *is a clique of* **co** *in P, namely a subset $S \subseteq P$ such that $\forall x, y \in S : x \neq y \rightarrow x$ **co** y.*

A cut *is a maximal co-set, namely a co-set S such that $\forall y \notin S : (\exists x \in S : x$ **li** $y)$*

In a causal net, a cut $c \subseteq B'$ is called a B-cut*, and a cut $c' \subseteq E'$, an* E-cut*.*

A line is a maximal totally ordered subset, it will be interpreted as expressing a sequential subprocess. It corresponds to a sequence of actions and conditions that need to occur in that particular order. It is relevant to stress that when order expresses time, a total ordering represents the possibility of the actions happening in that order. When order expresses causality, a total

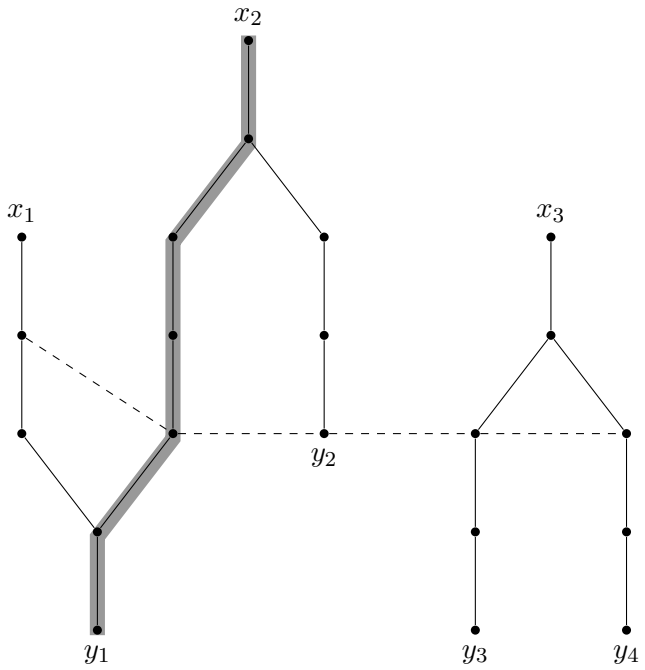

Figure 2.5: A partially ordered set P underlying the causal net of Figure 2.4. Solid lines represent the order, oriented downwards. Such that, for example, $x_3 \leq y_2$, and $x_1 \leq y_1$. The dashed line represents a *cut*. The shaded path is a line. $\min(P) = \{x_1, x_2, x_3\}$, and $\max(P) = \{y_1, y_2, y_3, y_4\}$.

ordering represent the necessity that the occurrences relate in this particular way.

Hence, a line represents a set of occurrences for which causality fully imposes the order of execution.

Symmetrically, a cut represents a set of occurrences that can happen in any time-wise order. Indeed, since no particular ordering is required amongst them, they may occur in any possible order. The lack of causal dependence allows to distribute a cut. In particular, the actions of an E-cut could be executed remotely from one another. This implies that they might as well occur simultaneously. However, the lack of global clock assumption prevents from observing such situation. Under this assumption, one could not effectively observe the time- wise order of occurrences of an E-cut.

An analogous situation is found regarding B-cuts. Any two elements of a B-cut correspond to conditions of an elementary net system that can be

marked simultaneously. Out of maximality, a B-cut corresponds to a marking, which is possibly visited along the execution of the system. Again, this does not mean that the marking is necessarily visited, only that it is reachable.

With this interpretation, the concurrency relation permits to analyse how the process can be distributed. It can be lifted to subsets of a partial order, that will be identified with the parts of the process that can be distributed, separated from one another. These parts will be considered as subprocesses, although their formalisation is weaker than the standard notion of subprocess of Petri net theory. They will be defined, and studied in Chapter 3.

2.2 Interleaving Semantics: Elementary Transition Systems

Elementary Net Systems are useful models in the interpretation of observable properties as logical propositions. Although these models convey a plain intuition about its motivations, the present work will rather focus on Labelled Transition Systems. There are two main reasons for this. First, Labelled Transition Systems are, among the two, the most widespread paradigm in either model-checking, and model design. Indeed, even though structural analysis of Net Systems has proven to be highly efficient, the amount of contributions in this field remains largely smaller than existing literature regarding Labelled Transition Systems. As a matter of fact, most of model checking techniques for Net Systems, require the computation of their Case Graphs, and it is this latter model which is actually checked. The popularity of Transition System Models is justified by the fact that, as automata, they remain closer to classical models of computation. As such, they admit interpretations that allow for widespread model-checking techniques.

2.2.1 Labelled Transitions Systems

The dynamic nature of Net Systems, presents some limitations. Indeed, in order to analyse the behaviour of the system, it would be suitable to handle a model in which all reachable states are depicted in one single snapshot. One can easily, given an elementary net system, obtain an equivalent model with such characteristics.

Definition 2.2.1 (Case Graph)**.** *Given an Elementary Net System* $N = (B, E, \mathcal{F}, m_0)$*, its* Case Graph *is a tuple* $\boldsymbol{CG}(N) = (M, E, T, m_0)$ *where*

M *is the set of reachable markings of* N

$T := \{(m, e, m') \in M \times E \times M \mid m[e\rangle m'\}$ *represents the transitions from states* m *to* m'*, labelled by event* $e \in E$*.*

In order to provide some intuition, the *Case Graph* of an elementary Net System can be defined alternatively as a *graph* [4]. Its vertices are all reachable markings of the system, and each arc is labelled by an event e, when its occurrence leads from one state to another. Namely $\mathbf{CG}(N) = (M, T', \lambda, m_0)$ where $T' \subset M \times M$, and $\lambda : T' \to E$ are such that $m[e\rangle m' \Rightarrow (m, m') \in T'$ with $\lambda(m, m') = e$. Such a definition provides a natural graphical representation, where markings are depicted as points, and there is an arrow labelled with an event that links two points, whenever the occurrence of this event changes the state of the system from one corresponding marking to the other.

The definition of case graph is absolutely analogous in the case of condition/event systems, one only needs to consider backward reachability in the definition of the set T of transitions.

Such a labelled graph is said to depict the *behaviour* of the system. Indeed, behavioural properties, become apparent in this representation. For instance, a *deadlock*, or full stop of the system, will be characterised by the existence of a state, or vertex, with no outgoing arcs. If the system is allowed to iterate part of its execution indefinitely, this will correspond to a circuit among states in the graph. The system is said to be *reversible* if from any state of the system, there is a path leading back to the initial marking.

It will be useful, in what follows, to consider the set of markings in which a given condition holds, as a subset of states of the case graph.

Definition 2.2.2 (Extension of a Condition)**.** *Let* $N = (B, E, \mathcal{F}, m_0)$ *be an elementary net system, and* $\boldsymbol{CG}(N) = (M, E, T, m_0)$*. For each* $b \in B$*, the* extension *of* b *is the set* $\mathbf{ext}(b) := \{m \in M \mid m(b) = 1\}$*.*

A path on the case graph of a net system, or rather the sequence of labels one encounters along it, is a firing sequence.

The formalism in which one expresses the case graph of a net system is in itself a widespread model of reactive computation. In the general case, its instances are simply called *labelled transition systems*. They form a class of automata built on the notion of state and of state transition. In labelled transition systems, transitions are labelled by the elements of an alphabet, here called *events*.

Definition 2.2.3 (Labelled Transition System)**.** *A* labelled transition system *is a structure* $A = (Q, E, T)$*, where* Q *is a set of* states*,* E *is a set of* events *and* $T \subseteq Q \times E \times Q$ *is a set of* transitions *such that*

1. *the underlying graph of the* transition system *is connected;*

2. $\forall (q_1, e, q_2) \in T \quad q_1 \neq q_2$*;*

3. $\forall (q, e_1, q_1)(q, e_2, q_2) \in T \quad q_1 = q_2 \Rightarrow e_1 = e_2$*;*

4. $\forall e \in E \quad \exists\, (q_1, e, q_2) \in T$*.*

It will be sometimes useful to drop the first requirement, in which case the labelled transition system will be called generalised transition system

In many definitions of labelled transition systems, the second axiom is not required. It states the absence of events such that their occurrence does not alter the state of the system. In elementary, or condition/events systems, such an event would have no observable effect, and should, out of the principle of extensionality, not be represented.

Remark 2.2.1. *In this work, only transition systems with a finite set of states, and a finite set of events are considered.*

Example 2.2.1. *Figure 2.6 depicts an elementary net system with initial marking* $\{p_1, r_2\}$*. It is the initial state* q_0 *of* $\boldsymbol{CG}(N)$*. All reachable markings of* N *are represented in* $\boldsymbol{CG}(N)$ *as the states* $\{q_0, q_1, \ldots, q_7\}$*. The four events of* N *are the labels on the arcs of* $\boldsymbol{CG}(N)$*. A transition going from* q_i *to* q_j *carries the label* e_k *whenever* $q_i[e_k\rangle q_j$*. The events* e_1*, and* e_4 *are concurrently enabled at* q_0*, and their firings lead respectively to* $q_1 = \{p_2, r_2\}$*, and* $q_4 = \{p_1, r_1\}$*. Since they are concurrent, after both have fired, the system will reach state* $q_2 = \{p_2, r_1\}$*, independently of the order in which they do. All reachable markings of* N *are states of the system* $\boldsymbol{CG}(N)$*.*

A whole field of study is dedicated to classify Labelled Transition Systems according to the class of Net Systems they can express the behaviour of. Such a study is commonly tagged as Net Synthesis. This work will focus on the Transition Systems which are isomorphic to the Case Graph of some Elementary Net System, namely *Elementary Transition Systems*, as well as on variations of such models.

2.2.2 Elementary Transition Systems

Since the case graph of a net system is a labelled transition system. A natural question to be asked, is whether every transition system is the case graph of some net system. This was shown not to be the case in general, and the

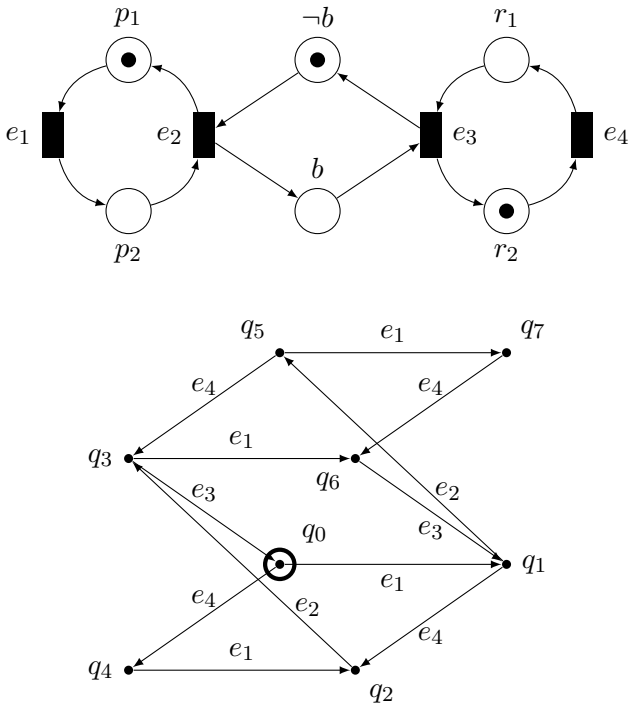

Figure 2.6: An elementary net system N (above), and its case graph $\mathbf{CG}(N)$ (below). The initial marking of N, is the initial state of $\mathbf{CG}(N)$ $q_0 = \{p_1, r_2\}$

characterisation of the subclass of transition systems which are the case graph of an elementary net system has proven to be non-trivial, and has led to a whole field of research, Petri Net Synthesis. The net synthesis problem can be stated as follows.

Given a labelled transition system A, is there a Petri Net System such that its Case graph is isomorphic to A?

The idea to solve the elementary synthesis dates back to [30, 31], and set the basis for solutions to the problem in wider classes of net systems.

Elementary transition systems can be naturally defined as the labelled transition systems which are the case graph of some elementary net system.

However, in their seminal series of papers [30, 31], A. Ehrenfucht and G. Rozenberg provided a full characterisation of this class of transition system. Such a characterisation is here presented as the actual definition. Although in [30, 31] the results were developed using a slightly different formalism, 2-

structures, they are here presented in terms of the models at stake in this work.

The key to the characterisation of the class of elementary transition systems is the notion of region.

A *region* of a transition system is a subset of its states such that each event has a *uniform crossing* relation, entering, leaving or not crossing, with the region itself through each of its occurrences, as formalised in the next definition.

Definition 2.2.4 (Region)**.** *A* region *of a transition system $A = (Q, E, T)$ is a subset r of Q such that every event* crosses r uniformly*, namely:*
$\forall e \in E, \forall (q_1, e, q_2), (q_3, e, q_4) \in T$*:*

1. *($q_1 \in r$ and $q_2 \notin r$) implies ($q_3 \in r$ and $q_4 \notin r$)* and
2. *($q_1 \notin r$ and $q_2 \in r$) implies ($q_3 \notin r$ and $q_4 \in r$).*

This *uniform crossing property* is sufficient for a subset of states to be a region. The intuition behind it is that these are the subsets with respect to which the orientation of transitions is consistent with the labelling. Indeed, as a case graph of a net system, a transition system labels its transitions with events of the net. The uniform crossing property makes sure that all instances of a same label have the same orientation with respect to regions. This in turn allows for identifying a region with the extension of a condition, the set of states which assign the value true to it. In this way, the orientation of a label with respect to a region provides the flow relation between the corresponding event, and condition. Given a transition system A, its set of regions will be denoted by $\mathcal{R}(A)$; given a state $q \in Q$, the set of regions containing q will be denoted by $\mathcal{R}_q(A)$ and, when the transition system that originates the regions is clear from the context, simply by $\mathcal{R}_q$. Note that the set of regions $\mathcal{R}(A)$ of a transition system $A = (Q, E, T)$ can not be empty since at least the whole set of states Q is a region.

For the sake of intuition, it is worth noting that the extension of each condition of a condition/event system is a region of its case graph. The proof of this result is a long known result that can be found, for instance, in [30, 31]. It is here reported to convey intuition about the relation between conditions of a net system, and regions of its case graph.

Proposition 2.2.1. *Let $N = (B, E, \mathcal{F}, M)$ be a condition/event net system, and $\boldsymbol{CG}(N) = (M, E, T)$ be its case graph. For each $b \in B$, consider its extension $\mathbf{ext}(b) := \{m \in M : b \in m\}$. Then $\forall b \in B : \mathbf{ext}(b) \in \mathcal{R}(\boldsymbol{CG}(N))$.*

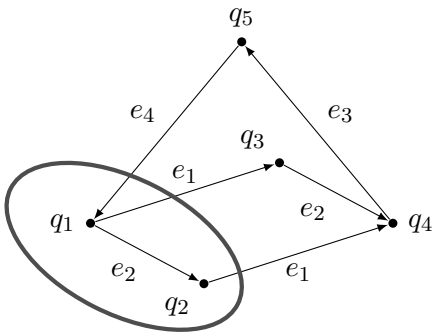

Figure 2.7: A region of an elementary transition system. All the occurrence of e_1 exit the region. The event e_4 has a single entering occurrence. No occurrence of e_2, or e_3, exits or enters the region. The region satisfies the uniform crossing property.

Proof. Consider $b \in B$, and $\mathbf{ext}(b) \subseteq M$. Let $e \in E$, then, since N is pure, either $e \in {}^\bullet b$, or $e \in b^\bullet$, or neither of the two.

Suppose there are two states $m_1 \in \mathbf{ext}\, b$, $m_2 \in M \setminus \mathbf{ext}\, b$, and a transition $(m_1, e, m_2) \in T$. Clearly, $b \in m_1$ and $b \notin m_2$. Then out of the firing rule, it must be ${}^\bullet e = m_1 \setminus m_2$, and $e^\bullet = m_2 \setminus m_1$, then $b \in {}^\bullet e$, and so $(b, e) \in \mathcal{F}$. Then it holds that $\forall m \in M : m[e\rangle \rightarrow b \in m$. This, in turn, implies that $\forall (m_3, e', m_4) \in T : e' = e \rightarrow (b \in m_3$ and $b \notin m_4)$. And so every transition labelled with e will go from a state in the extension of b, to a state outside of it.

Now suppose there are two states $m_1 \in M \setminus \mathbf{ext}\, b$, $m_2 \in \mathbf{ext}\, b$, and a transition $(m_1, e, m_2) \in T$, then $e^\bullet = m_2 \setminus m_1$. An analogous argument shows that every transition labelled with e will go to a state in the extension of b, from a state outside of it.

Finally, consider the following two cases. On one hand, if $m_1 \in M \setminus \mathbf{ext}\, b$, and $m_2 \in M \setminus \mathbf{ext}(b)$, on the other hand $m_1 \in \mathbf{ext}(b)$, and $m_2 \in \mathbf{ext}(b)$. In either case, the existence of a transition $(m_1, e, m_2) \in T$ implies that ${}^\bullet e = m_1 \setminus m_2$, and $e^\bullet = m_2 \setminus m_1$, and so $b \notin ({}^\bullet e \cup e^\bullet)$. Then b is independent from e, neither (b, e), nor (e, b) are in $\mathcal{F}$. Hence, at any marking, the firing of e will not change the value of b, $\forall (m_3, e', m_4) \in T : e' = e \rightarrow (b \in m_3 \leftrightarrow b \in m_4)$. □

This result motivates that the notion of pre- and post-sets can be extended to regions.

Definition 2.2.5 (Pre-sets, Post-sets on Transition Systems)**.** *Let $A = (Q, E, T)$ be a transition system. The* pre-set *and* post-set *operations, denoted respectively by the operators ${}^\bullet(\cdot)$ and $(\cdot)^\bullet$, applied to regions $r \in \mathcal{R}(A)$ and events $e \in E$ are defined by:*

1. ${}^\bullet r = \{e \in E \mid \exists\,(q_1, e, q_2) \in T \text{ such that } q_1 \notin r \text{ and } q_2 \in r\}$;
2. $r^\bullet = \{e \in E \mid \exists\,(q_1, e, q_2) \in T \text{ such that } q_1 \in r \text{ and } q_2 \notin r\}$;
3. ${}^\bullet e = \{r \in \mathcal{R}(A) \mid e \in r^\bullet\}$;
4. $e^\bullet = \{r \in \mathcal{R}(A) \mid e \in {}^\bullet r\}$.

The pre- and post-sets of the extension of a condition, as a region of the case graph of a net system, coincide with its the pre- and post-sets as described by the flow relation of the net. The following is a well known result.

Proposition 2.2.2. *Let $N = (B, E, \mathcal{F}, q_0)$ be an elementary net system, and let $A = \boldsymbol{CG}(N) = (Q, E, T)$ be its case graph. Then $\forall b \in B : {}^\bullet b = {}^\bullet\,\mathbf{ext}(b)$ and $b^\bullet = \mathbf{ext}(b)^\bullet$*

Proof. Let $b \in B$, and $r = \mathbf{ext}(b)$. Consider $e \in {}^\bullet r$ then $\exists (q_1, e, q_2) \in T$ such that $q_1 \notin r$ and $q_2 \in r\}$. Then $q_1(b) = 1$, $q_2(b) = 0$, and $q_1[e\rangle q_2$, and so $b \in {}^\bullet e$. Conversely, if $b \in {}^\bullet e$, then for every pair of markings $q_1, q_2 \in Q$ such that $q_1[e\rangle q_2$, it must hold that $q_1(b) = 1$, and $q_2(b) = 0$. Then $(q_1, e, q_2) \in T$, and so $e \in {}^\bullet r$.

The result for $e^\bullet$ is derived analogously. □

Regarding the synthesis problem, the set of regions of a transition system provides all candidates for conditions of the net system to be found. Whether such a system exists depends, however, on the fact that the system behaviour is guided by the markings of such conditions. Intuitively, there must be enough regions to determine this behaviour.

These situations were characterised in [30, 31] by identifying the so called *separation axioms*.

Definition 2.2.6 (Elementary Separation Axioms)**.**

1. $\forall\, q_1, q_2 \in Q : \mathcal{R}_{q_1} = \mathcal{R}_{q_2} \rightarrow q_1 = q_2$;
2. $\forall q_1 \in Q : \forall e \in E : {}^\bullet e \subseteq \mathcal{R}_{q_1} \rightarrow \exists q_2 \in Q : (q_1, e, q_2) \in T$;

These axioms make sure that a system built with this set of regions as conditions will generate the transition system as its case graph. Note that the key idea is that the elementary separation axioms make sure that the existential quantifiers in Definition 2.2.5 are in fact universal. This, in turn, ensures that the principle of extensionality holds among the set of regions, and the set of events of the system. Indeed, when the separation axioms hold, events are characterised by their orientation with respect to regions. Furthermore, when the graph underlying the transition system is connected, then a region r is characterised by the pair $\langle {}^\bullet r, r^\bullet \rangle$.

However, since elementary net systems depend on the definition of an initial state, one must also be able to identify it on the transition system. An elementary transition system will present a state from which all other states are reachable.

Definition 2.2.7 (Elementary Transition System)**.** *An* elementary transition system *is a tuple* $A = (Q, T, E, q_0)$ *where* $A = (Q, E, T)$ *is a labelled transition system satisfying the elementary separation axioms of Definition 2.2.6, and* $q_0 \in Q$ *is an* initial state *such that each* $q \in Q$ *is reachable from it.*

The consideration of this initial state is rather cumbersome. Reachability being a dynamic property, it leads away from the scope of this work, concerned essentially with structural properties of the systems. It will therefore be suitable to handle condition/event transition systems.

2.2.3 Condition/Event Transition Systems

In a condition/event net system, instead of defining an initial marking, a whole class of reachable markings is considered. However, for such a class to be properly defined, backward reachability needs to be considered. In some sense, a condition/event system allows for several different initial states.

In determining whether a labelled transition system is the case graph of some condition/event net system, the requirement that there exists an initial state is dropped. However, the backward firing of events, in such a model, imposes the consideration of an additional axiom.

Definition 2.2.8 (Condition/Event Separation Axioms)**.** *A* Condition/Event Transition System *is a transition system such that the following conditions are satisfied:*

1. $\forall q_1, q_2 \in Q : \mathcal{R}_{q_1} = \mathcal{R}_{q_2} \rightarrow q_1 = q_2$;
2. $\forall q_1 \in Q : \forall e \in E : ({}^\bullet e \subseteq \mathcal{R}_{q_1}) \rightarrow (\exists q_2 \in Q : (q_1, e, q_2) \in T)$;

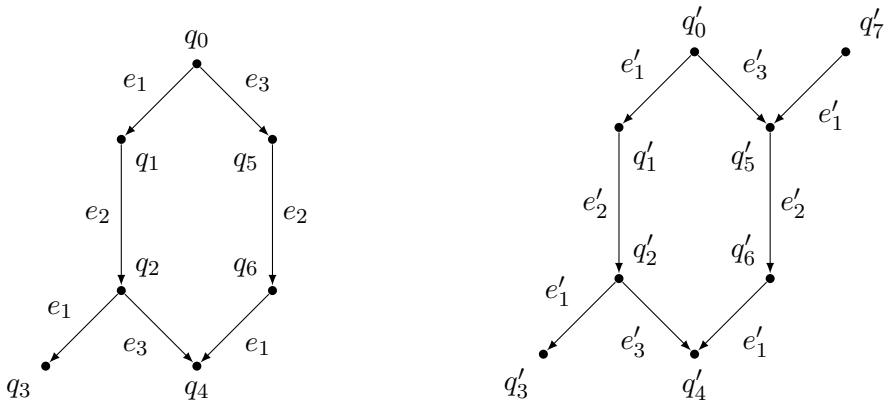

Figure 2.8: To the left, an elementary transition system A_1. To the right, a condition/event transition system A_2

3. $\forall q_1 \in Q : \forall e \in E : (e^\bullet \subseteq \mathcal{R}_{q_1}) \rightarrow (\exists q_2 \in Q : (q_2, e, q_1) \in T)$.

It was shown in [30, 31] that so defined, a condition/event transition system is isomorphic to the case graph of some condition/event net system.

Condition/event transition systems and elementary transition systems form distinct classes of labelled transition systems. These are not comparable. Indeed, the following example shows both a condition/event transition system which is not an elementary, and an elementary transition system which is not condition/event.

Example 2.2.2. *The transition system on the left side of Figure 2.8 is elementary, with initial state q_0. It can be shown that it is not condition/event. Focus on the regions in the post-set of e_1. Let $r \in e_1^\bullet$. Certainly, r must contain q_3, and not q_2. It must also contain q_1, but not q_0. Since it contains q_1 and not q_2, it must be that $r \in {}^\bullet e_2$, and so it must also contain q_5, but not q_6. Now $\forall r \in e_1^\bullet : q_5 \in r$, so in particular $e_1^\bullet \subseteq \mathcal{R}_{q_5}$. Note that e_1 is not backward enabled at q_5, so the third axiom of Definition 2.2.8 fails to hold.*

On the same figure, the transition system to the right is condition/event. In this similar case, e'_1 is backward enabled at q'_5. However, the system is not elementary. Indeed, q'_7 is not forward reachable from q'_0, and vice-versa. No choice of an initial state makes all its states forward reachable.

Note that the definition of region is the same in both models. As one can verify on the previous example, the sets of regions can often be identified.

This is the case, not only when considering the plain set of regions, but also when endowing it with a structure.

The class of algebraic structures that one obtains when ordering regions by set inclusion is highly relevant for the study of distributability of the system. Indeed, one can identify regions with the local states of the system, and their structure as subsets of states will provide insight in the way the system can be distributed. Hence, the formalism with which regions are structured is one of the main focuses of this work. When ordered by inclusion, regions form *orthomodular partial orders*.

2.3 Orthomodular Posets as Logics

When reasoning about the properties that hold in a system, one can identify each of these with its extension. The extension of a property is the set of global states in which it holds, its value is true. Obviously the extension of the negation of any property coincides with the set states in which this property carries the value false, the set complement of its extension. If the extension of a property is contained in the extension of another, the second will be true whenever the first is. Properties are, in this way, endowed with an order that translates the idea of implication.

In the purely sequential case, the lack of concurrency in a labelled transition system, lifts all restrictions on observability. Any subset of states corresponds to a proposition which is observable. Such a full structure of subsets is known to form a Boolean algebra, when ordered by set inclusion.

This ceases to be the case when one considers a labelled transition system depicting concurrent behaviour. In this case, the system is assumed to be distributed and so the principle of locality applies. Some subsets of states will not be observable anymore. Only regions of the system are considered observable, due to their consistency with respect to the effect of the event in the system. Still, one can order the set of regions according to set inclusion, obtaining a slightly weaker structure than a Boolean algebra. Indeed, most of the identities that hold in the latter, also apply in the former, distributivity, however, is replaced by a much weaker notion, orthmodularity.

A similar mathematical formalisation arises when reasoning about concurrency in processes. In this case, one considers subprocesses. A subprocess is understood to be a part of the global process that can be executed as a module. It can be abstracted by a single action, or rather as it will be shown in Chapter 3, as a local state of the system. The scope of the construction that this work presents, is to isolate parts of the process which can run in parallel,

and for the resulting set of subprocesses, set inclusion defines the relation "is a subprocess of". In this case, one could obtain a Boolean algebra, if no subprocesses interact with each other, and can run independently from one another. However, when they depict causal dependencies, the mathematical structure is weakened, replacing once again distributivity by the orthomodular law.

2.3.1 Logics as Partial Orders

Orthomodular posets were introduced by Birkhoff and von Neumann in 1936 as the algebraic structure of linear subspaces of Hilbert spaces, in relation to quantum mechanics [19]. It was originally intended as a formalisation of the propositions which are testable on a quantum mechanical system. This field of physics has been the main engine in the development of the theory of these particular structures, often referred to as *quantum logics*. In order to provide some intuition regarding the logical interpretation of orthomodular posets, and to justify the prominent use of the term *logic* to refer to them, a few well-known concepts are collected.

Classical *propositional logic* can be built from a set of variables $V = \{v_1, v_2, ..., v_n\}$, and two constants $C = \{0, 1\} = \{\mathbf{false}, \mathbf{true}\}$
Formulas of the logic $\phi \in \Phi$ are defined recursively:

1. $C \subset \Phi$, and $V \subset \Phi$
2. $\forall \phi, \psi \in \Phi : (\phi), \neg\phi, \phi \vee \psi, \phi \wedge \psi, \phi \to \psi \ \in \Phi$

A truth assignment on the variables is a function $t : V \to C$. It uniquely determines a truth assignment $\tilde{t} : \Phi \to C$ on all formulas.

One can then identify formulas which take the same value for all truth assignments. To formalise this, define $\phi \sim \psi \Leftrightarrow (\forall t \in C^V : \tilde{t}(\phi) = \tilde{t}(\psi))$. $\sim$ is not only an equivalence relation, but a congruence with respect to $\neg, \vee, \wedge$, and $\to$.

Remark 2.3.1. *Consider a set X with an n-ary operation $o : X^n \to X$. An equivalence relation $\sim$ in X is called a* congruence *with respect to o whenever it is consistent with it, namely*

$$\forall (x_i)_{i \leq n}, (x'_i)_{i \leq n} : (\forall i \leq n : (x_i, x'_i) \in \sim) \to (o(x_i)_{i \leq n}, o(x'_i)_{i \leq n}) \in \sim$$

$\Phi/\sim$ forms a Boolean algebra $\mathcal{B}$, further more, if V is finite, then so is $\Phi/\sim$. Intuitively, a Boolean algebra considers the logical propositions in terms of the truth values they can take, and represents their logical dependencies by the means of a negation operation, and an ordering which defines implication. It can be formalised as a tuple $\mathcal{B} = (B, 0, 1, \leq, \neg)$ such that

1. $B = \Phi/\sim$,

2. $\forall \phi \in \Phi : \neg[\phi]_\sim = [\neg\phi]_\sim$,

3. $\forall [b_1]_\sim, [b_2]_\sim \in B : [b_1]_\sim \leq [b_2]_\sim \Leftrightarrow (\forall \tilde{t} : \tilde{t}(b_1 \to b_2) = 1)$

Then $\forall [b_1]_\sim, [b_2]_\sim \in B$.

1. $[b_1]_\sim \vee [b_2]_\sim$ is the *least upper bound* of $[b_1]_\sim$ and $[b_2]_\sim$,

2. $[b_1]_\sim \wedge [b_2]_\sim$ is the *greatest lower bound* of $[b_1]_\sim$ and $[b_2]_\sim$

The last two observations motivate the use of the symbols in the next definition.

Definition 2.3.1 (Join, Meet)**.** *Consider a set X endowed with an order $\leq$. Let $S \subseteq X$. The* least upper bound*, supremum, or* join *of S, denoted $\bigvee_{s \in S} s$, or simply $\bigvee S$, is an element $x \in X$ such that*

1. $\forall s \in S : s \leq x$

2. $\forall y \in X : (\forall s \in S : s \leq y) \to x \leq y$

Analogously, the greatest lower bound*, or,* infimum*, or* meet *of S, denoted $\bigwedge_{s \in S} s$, or simply $\bigwedge S$, is an element $x \in X$ such that*

1. $\forall s \in S : x \leq s$

2. $\forall y \in X : (\forall s \in S : y \leq s) \to y \leq x$.

In an arbitrary partial order, such elements might not be defined. However, it is inherent to their definition, that if they exist they must be unique. In the general case, this uniqueness is not guaranteed.

When there is risk of confusion, the operations on a partial order P will be tagged with the corresponding name: $\leq_P, \vee^P, \wedge^P$.

Definition 2.3.2 (Lattice, Complete Lattice)**.** *A partial order $(P, \leq)$ is called a* lattice *whenever for any finite subset $S \subseteq P$, $\bigvee S \in P$ and $\bigwedge S \in P$ are well defined.*

It is called a complete lattice *whenever for any subset $S \subseteq P$, $\bigvee S \in P$ and $\bigwedge S \in P$ are well defined.*

When the set S consists of two elements $\{a, b\}$, in-fix notation is commonly used, $a \vee b$, and $a \wedge b$ instead of $\bigvee\{a, b\}$, and $\bigwedge\{a, b\}$, respectively. In this way, $\vee$ and $\wedge$ are seen as binary operations. They are commutative $a \vee b = b \vee a$, and $a \wedge b = b \wedge a$,and associative $a \wedge (b \wedge c) = (a \wedge b) \wedge c$, and $a \vee (b \vee c) = (a \vee b) \vee c$. In general, they do not distribute over each other.

Definition 2.3.3 (Distributive Lattice)**.** *A* lattice $(L, \vee, \wedge)$ *is called* distributive *when* $\forall a, b \in L$ *the following hold*

1. $a \wedge (b \vee c) = (a \wedge b) \vee (a \wedge c)$
2. $a \vee (b \wedge c) = (a \vee b) \wedge (a \vee c)$

When a poset has a minimum element, and a maximal element, it said to be *bounded*. Note that any formula ϕ implies **true**, and is implied by **false**, this motivates the use of 0, and 1, to denote respectively the minimal and maximal elements of a bounded poset.

Definition 2.3.4 (Orthocomplemented Poset)**.** *A poset* $(P, \leq)$ *is said to be* complemented *whenever there are two elements* 0*, and* 1*, and a unary operation* $(\cdot)'$ *such that*

1. $\forall a \in P : 0 \leq a \leq 1$.
2. $\forall a \in P : \exists a' \in P : a \vee a' = 1$ *and* $a \wedge a' = 0$.

a' *is called the* complement *of* a.

$(P, \leq)$ *is said to be* orthocomplemented *if it is complemented, and the following hold*

1. $\forall a \in P : (a')' = a$.
2. $\forall a, b \in P : a \leq b \rightarrow b' \leq a'$.

In this case, $(\cdot)'$ *is called an* involution*. When two elements* a *and* b *of an orthocomplemented poset satisfy* $a \leq b'$*, then clearly* $b \leq a'$*, and they are said to be* orthogonal*, written* $a \perp b$*. Orthogonality is an irreflexive and symmetric relation. It is only transitive in Boolean algebras.*

Note that $\forall x \in P : 0 \leq x'$, and so 0 is orthogonal to all elements of the poset. In a complemented poset, De Morgan's laws always hold. $(a \vee b)' = a' \wedge b'$, and $(a \wedge b)' = a' \vee b'$.

By requiring all these conditions, one obtains a Boolean algebra

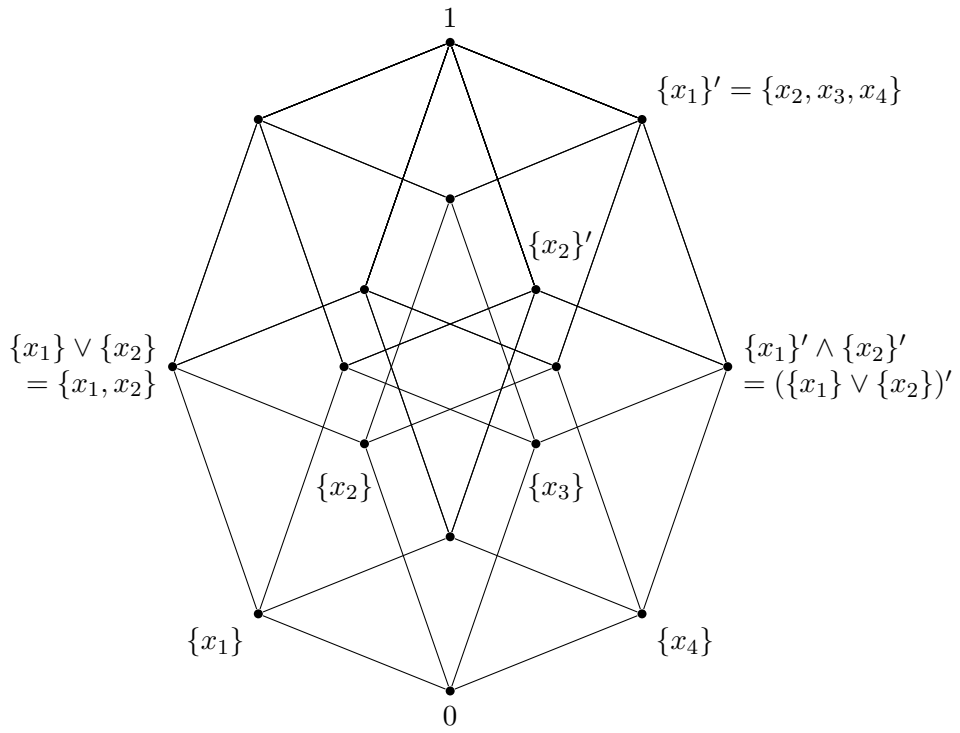

Figure 2.9: Hasse diagram of a Boolean algebra, in which order is represented upwards. It has a minimal element 0, and a maximal element 1. It has 16 elements. It represents the collection of subsets of $X = \{x_1, x_2, x_3, x_4\}$, and set inclusion as ordering. The join of elements is the union, the meet is the intersection, and the orthocomplement corresponds to set complement.

Definition 2.3.5 (Boolean Algebra)**.** *A* Boolean algebra *is an orthocomplemented distributive lattice.*

Orthomodular partial orders are a weaker version of Boolean algebras. Although admitting an analogous logical interpretation, join and meet operations are not defined for all pairs of elements. They are orthocomplemented posets, but in general, not lattices. Still, joins are defined among orthogonal elements, and they satisfy the orthmodular law, which states a weak form of distributivity subordinated to complementation. In this work, notation and terminology follow those of [51], where orthomodular posets are called *quantum logics*, or simply *logics*. The name is motivated by the fact that this notion was first formalised, and further on studied, in an attempt to axioma-

tise quantum mechanics.

Definition 2.3.6 (Orthomodular partial order)**.** *An* orthomodular partial order*,* orthomodular poset*, also* quantum logic $\langle L, \leq, (.)', 0, 1\rangle$ *is a set* L *endowed with a partial order* $\leq$ *and a unary operation* $(\cdot)'$ *(called orthocomplement), such that the following conditions are satisfied:*

1. L *has a least and a greatest element (respectively* 0 *and* 1*) and* $0 \neq 1$*;*
2. $\forall\, x, y \in L \quad x \leq y \Rightarrow y' \leq x'$*;*
3. $\forall\, x \in L \quad (x')' = x$*;*
4. *if* $\{x_i \mid i \in \mathbb{N}\}$ *is a countable subset of* L *such that* $i \neq j \Rightarrow x_i \perp x_j$*, then* $\bigvee_{i\in\mathbb{N}} x_i$ *exists in* L*;*
5. $\forall\, x, y \in L : x \leq y \rightarrow y = x \vee (x' \wedge y)$*.*

The latter condition is the aforementioned orthomodular law*.*

In this work, the term *logic* will often be used as a shorthand for quantum logic, or orthomodular poset, and L instead of $\langle L, \leq, (.)', 0, 1\rangle$, when there can be no ambiguity.

Definition 2.3.7 (Sublogic)**.** *A sublogic of* L *is a subset* $\hat{L}$ *of* L *that remains a logic with respect to the restrictions of the operation* $(.)'$ *and the relation* $\leq$ *to* $\hat{L}$*. In particular,* $x \in \hat{L} \Rightarrow x' \in \hat{L}$ *and, for* $X = \{x_i\}_{i\in\mathbb{N}} \subseteq \hat{L}$ *a sequence of mutually orthogonal elements, the supremum of* X *taken in* L *belongs to* $\hat{L}$*.*

A sublogic is Boolean *if it is a Boolean algebra.*

Morphisms of logics are defined in such a way that they preserve order (and consequently orthogonality) and compatibility.

Definition 2.3.8 (Logic Morphism)**.** *([51]) Let* L_1 *and* L_2 *be logics. A mapping* $f : L_1 \rightarrow L_2$ *is a* morphism of logics *if the following conditions are satisfied:*

1. $f(0) = 0$*;*
2. $\forall x \in L_1\ f(x') = f(x)'$*;*
3. *for any sequence* $\{x_i \mid i \in \mathbb{N}\}$ *of mutually orthogonal elements in* L_1*,* $f(\bigvee_{i\in\mathbb{N}} x_i) = \bigvee_{i\in\mathbb{N}} f(x_i)$*.*

Proposition 2.3.1. *([51]) Morphisms of logics preserve order, and orthogonality.*

A morphism $f : L_1 \to L_2$ is an *isomorphism* if f is injective, maps L_1 onto L_2 and f^{-1} is a morphism. Moreover, f is an *embedding* if $f(L_1)$ is a sublogic of L_2 and $f : L_1 \to f(L_1)$ is an isomorphism.

2.3.2 Partial Order of Regions

Regions of elementary (or condition/event) transition systems were proven to form orthomodular partial orders in [7]. In order to provide some intuition regarding orthomodular posets, an analogy is here presented with the representability of Boolean algebras.

In his seminal paper of 1936 [60], M. H. Stone presented a result according to which the elements of a boolean algebra can be interpreted as subsets of a carrier set X.

Indeed, one can easily check that given a carrier set X, then the collection of all its subsets, called its *power set* $2^X : \{S \subseteq X\}$ can be endowed with a Boolean algebra structure. It is in order to do so, sufficient to take set inclusion as order relation, and set complement as negation. Defined as the infimum, and supremum respectively, meet, and join operations coincide with set intersection, and set union.

As a matter of fact, in what concerns this work, this is the structure of regions one obtains from a system which is purely sequential. In either elementary or condition/event framework, fully sequential systems are characterised by the fact that no pair of transitions carry the same label. Intuitively, two different states are, as markings of a net system, distinguished by at least one condition. If both states enable the same event, then this condition can not be involved in its firing. Such a condition being independent from the event, it must belong in different sequential components, and so the system has at least two sequential components.

Example 2.3.1. *Consider the transition system $A = (X, E, T)$ with set of states $X = \{x_1, x_2, x_3, x_4\}$, set of events $E = \{e_1, e_2, e_3, e_4\}$, and transitions $T = \{(x_1, e_1, x_2), (x_2, e_2, x_3), (x_3, e_3, x_4), (x_4, e_4, x_1)\}$. It has a sequential cyclic behaviour, and is either condition/event, or elementary, if one selects any of its states as initial. Since all transitions have different labels, the uniform crossing property imposes no restrictions, and any subset of X is a region. Then $\mathcal{R}(A) = 2^X$ forms the Boolean algebra of Figure 2.9, when ordered by set inclusion.*

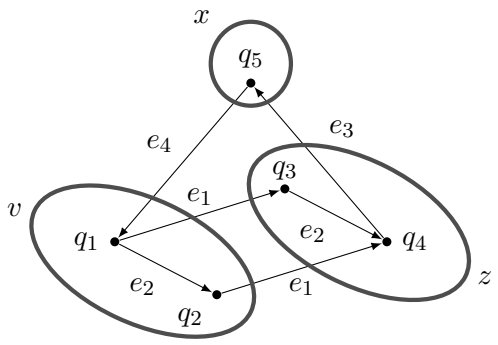

Figure 2.10: The regions of the transition system of Figure 2.7 form partitions of the set of states.

Note that the set of all singletons, subsets consisting of one single element, form a partition of the carrier. Any subset can be retrieved as disjoint unions of these singletons.

When a transition system is sequential, any subset of states is trivially a region. Hence, the structure of regions forms a Boolean algebra. In the general case however, the labelling of transitions prevents some subsets from being regions, thus leading to a weaker structure. Some interesting properties were proven in [5].

Proposition 2.3.2. *Let $A = (Q, E, T)$ be a condition/event transition system, then $\forall r_1, r_2 \in \mathcal{R}(A)$:*

1. $r_1 \cap r_2 = \emptyset \rightarrow r_1 \cup r_2 \in \mathcal{R}(A)$

2. $Q \setminus r_1 \in \mathcal{R}(A)$

The result is analogous for elementary systems. As a consequence, regions can be organised in partitions of the set of states.

Example 2.3.2. *Consider the transition system of Figure 2.10. The regional partition $\{\{q_1, q_2\}, \{q_5\}, \{q_3, q_4\}\}$ is made of disjoint unions of the one depicted in the figure. They correspond to the same sequential component. The set of regions $\{\{q_1, q_3\}, \{q_2, q_4\}, \{q_5\}\}$ forms another partition. It represents a different sequential component. The partitions share the region $\{q_5\}$. This region is a synchronisation of the two sequential components. Intuitively, the two sequential components must agree in order for e_3 to fire.*

Hence, some regions might belong to several partition. The orthomodular law establishes that, whenever the intersection of two regions r_1, and r_2 is a region, then one can find a regional partition of the set of states such that r_1 and r_2 can be retrieved as unions of elements of the partition. This is taken as the main intuitive idea behind orthomodular posets. When defined over a power set, they can be seen as formalising interactions between partitions.

It is relevant to mention the great effort put by the community in understanding the nature of orthomodular posets [39, 37, 27, 18]. Although the main motivation for this development has historically flown from physics, the mathematical subject has become of interest in itself. The author of this work wishes to highlight the contributions of G. Bruns, and J. Harding, among whose work, [22, 35, 36] are particularly enlightening.

The power set of any set forms a Boolean algebra. The acclaimed result from M. H. Stone, however, was the proof of the converse. Every Boolean algebra, can be seen as the collection of subsets of some carrier X. It is said that they admit a *concrete representation*. Extending the analogy with quantum logics in this direction is not straightforward. Indeed, it was shown that not all orthomodular posets can be seen as a collection of subsets [34]. The characterisation of this class is due to S. Gudder, and this subject will be covered extensively in Chapter 4. As a matter of fact, the present work follows this line of thought, and tackles the problem of whether a given quantum logic admits a representation as the set of regions of some elementary (or equivalently, condition/event) system.

The orthomodular poset that arises as the structure of regions gathers information about the sequential components of the system. Indeed, when the extension of two conditions of a net system are such that no regional partition contains both, then the two conditions must belong to different sequential components. The existence of a partition containing both extensions expresses a dependency in the way these can be marked. The non-existence of such a partition then imposes that the two conditions can be marked independently from one another.

2.3.3 Parial Order of Subprocesses

Orthomodular posets also arise in the analysis of concurrency in processes. In [9], the authors proposed an algebraic operation that allows to identify the subprocesses, as subsets of a process, which are concurrent. When this is the case, it is asserted that these parts of the process can run in parallel. One can then identify all such subprocesses and endow them with a structure that will carry information about their concurrency. The purpose of this section

is to provide some intuition about this structure. The full formalisation, and contribution regarding this matter, will be developed in the next chapter.

With this notion of subprocess, one can provide a structure, analogous to that of regional posets. Subprocesses, seen as subsets of a process, can be ordered by set inclusion. A form of negation can also be derived. In this case, however, rather than corresponding to set complement, it will directly encode causal independence, making the term *complement* or *orthocomplement* more suitable than *negation*. In [10], it was shown that when a process is formalised by the means of a causal net, the structure of its subprocesses forms an orthomodular partial order. As a matter of fact, in this case, meet and join operations are everywhere well-defined, so the poset is actually a lattice.

Definition 2.3.9 (Orthomodular Lattice)**.** *An* orthomodular lattice *is an orthomodular poset which is a lattice.*

The intuition behind orthomodularity is, with this interpretation, very different than in the case of regions. Indeed, since the orthocomplement operation is not interpreted as set complement, orthomodularity does not provide partitions of the process into subprocesses, in the general case. Instead, given a process, one usually interprets the lines of the underlying causal net, as the fully sequential subprocesses it holds, and it is on the set of these lines that one can see the partitions induced by orthomodularity. In this view, one considers the set of lines that cross a given subprocess, those which have non-empty intersection with it. The obtained orthomodular lattice provides information on how the considered subprocesses partition the set of lines, and the dependencies that hold among such partitions.

In this sense, the proposed algebraic operation will lead to a Boolean algebra when applied on a process consisting of independent sequential subprocesses which do not interact whatsoever. The weakening of a Boolean algebra leading to an orthomodular lattice arises from the causal dependencies between the parts of the process which are allowed to run in parallel. Indeed, when considering two elements of this orthomodular lattice as subprocesses, the fact that they cannot partition the set of lines is due to the existence of a line which crosses both of them. This expresses that there must be a causal dependence between them. A set of subprocesses that arise from the presented operation, will intersect disjoint sets of lines whenever they can be distributed. These are the situations in which they can be implemented in parallel.

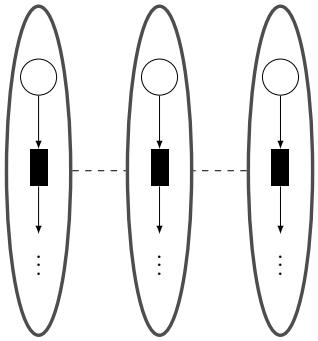

Figure 2.11: A causal net with three independent lines. It represents the process of any net system consisting of three independent cyclic state machines. Its lattice of subprocesses will form a Boolean algebra with three minimal elements. The dashed lines represent independence of subprocesses, they indicate how the process can be distributed.

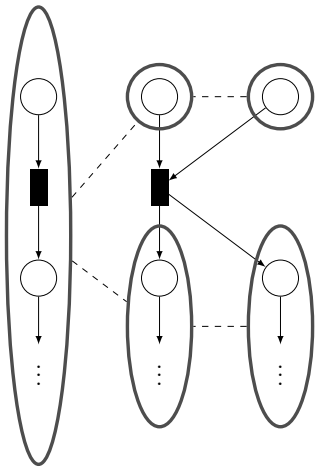

Figure 2.12: When the lines of a process synchronise, the lattice of subprocesses ceases to be Boolean. The way the process can be distributed is more complex.

Chapter 3

Process Distributability

As seen in Chapter 2, processes executed by elementary net systems are formalised by means of causal nets. In such models, one abstracts from the orders in which concurrent actions occur along a sequence, since they are interpreted as causally independent. As Petri net models, causal nets not only depict occurrences of actions but also the conditions that lead from one to another, and translate their direct causal dependency. Since all choices are resolved in such a model, the underlying net presents no conflict, and one can distinguish event occurrences from conditions structurally. Hence, the partial ordering of elements of a causal net is sufficient to perform the structural analysis that permits to identify subprocesses, as well as the way these can be distributed.

Subprocesses are singled out as subsets of elements of the causal net which inherit its partial order. When subprocesses considered as a such subsets, are contained in one another, the smaller is considered a subprocess of the larger. In the construction in which these subprocesses are extracted, an additional relation between them is obtained, orthogonality. When two subprocesses are orthogonal, they are intuitively executable in parallel. The ordering of subprocesses, together with this relation, provides them with a richer structure, that admits a logical interpretation. It is in fact an orthomodular lattice.

In particular, such a lattice presents the property that it can be faithfully represented by its set of minimal subprocesses, together with their orthogonality relations. Minimal subprocesses are of particular interest, since they constitute the pieces that can be run in parallel, in a setting of maximal distribution. These subprocesses present the added advantage that they can inherit the ordering of the process they were extracted from. This ordering therefore expresses the causal relations between minimal subprocesses. The lack of order relation among them coincides with their orthogonality relation.

The set of minimal subprocesses, with the order relation inherited from their containing process is shown to be an abstraction of it.

When structured in such a way, subprocesses carry all the information regarding how they can be distributed, but also the causal dependencies that bind them together. This new partial ordering of minimal subprocesses will be shown to be interpretable as a causal net of its own, in which subprocesses are identified with conditions expressing the proposition "the subprocess is in execution". On top of that, a means of retrieving their causal dependencies as action occurrences will be provided.

3.1 Processes as Partial Orders

Partially ordered sets are a common characteristic in different models of true concurrent processes, such as event structures [45, 62] or Mazurkiewicz traces [42]. All these models share the fact that maximal totally ordered subsets (or chains) represent sequential subprocesses, whereas maximal subsets of pairwise unordered elements (or antichains) can represent occurrences of global states. Several subset operators have been defined on these structures. In particular, *Nielsen et al.* have presented downwards closed subsets of event structures, or configurations, as forming interesting spaces when ordered by inclusion: domains [45, 62]. Intuitively, these configurations gather information on the history of the subprocess leading to a particular set of events. In this sense they can be understood as partial states, uniquely determined by the past of a given subset, thus relying solely on causal dependence relations. As a matter of fact, inclusion in a domain represents a chronological ordering of possible observations. The closure operator studied in this chapter however, evolves around concurrency relations and independence of subprocesses. Like in domains, it provides a family of subsets, and endows it with a structure. Nevertheless, unlike configurations, it presents a notion of complementation determined by concurrency. Any element is by definition concurrent to all the elements in its complement, and so it could not distinguish one of them from another without a global clock. From its local point of view its whole complement subset behaves as a single local state. In the structure obtained from this closure operator, inclusion is not to be interpreted as a chronology, but rather conveys the idea of a coarser point of view, or abstraction. Informally, when considering that sequential subprocesses can only share information by synchronising or splitting, then this operator determines which information about the process is available locally.

3.1.1 Causal nets and Processes

As introduced in Section 2.1.3, the chosen formalism to represent processes is that provided in Petri net theory. In this way, a process can be seen as one possible execution of an elementary system. The definitions in this section are taken from [17], and adapted to cover only the simpler case of elementary systems.

Throughout this section, $N = (B, E, \mathcal{F}, m_0)$ will denote an elementary net system, $N' = (B', E', \mathcal{F}')$ a causal net, and $(P, \leq)$ the partial order derived from it as $P = B' \cup E'$, and $\leq = \mathcal{F}'^{+}$

As seen in Section 2.1.3, causal nets are suitable for representing processes. Not every such net, however, does in fact represent one. The notion of process can hardly be considered without relating it to the system responsible for its execution. This relation will be a requirement in the definition.

A process run by an elementary net system should start at the initial marking, and so the first restriction imposed on the causal nets, is their capacity to represent it.

Definition 3.1.1 (Initial Cut)**.** *Let N' be a causal net, and $(P, \leq)$ the partial order derived from it. An* initial cut *of either N', or P, is the set*

$$\min(P) = \{x \in P \mid \forall y \in P : y \leq x \rightarrow x = y\}$$

It is straightforward to verify that $\min(P)$ is indeed a cut. A causal net will need to have a B-cut as initial cut in order to represent an elementary process.

Another requirement is that every element of the net must be reachable from the initial marking in a finite number of steps. Not only must this happen, but it must hold when restricted to any line of the process which contains the element, and crosses the cut. Intuitively, every causal influence of the initial cut on an arbitrary element, must be tractable in a finite number of steps.

Definition 3.1.2 (Discrete with Respect to a Cut)**.** *Let $(P, \leq)$ be a poset, and two elements $x, y \in P$. The* interval *between x, and y is defined as*

$$[x, y] := \{z \in P \mid x \leq z \text{ and } z \leq y\}$$

Note that $[x, y] = \emptyset$, unless $x \leq y$.

Given two subsets S_1 and S_2 in P, the notion of interval generalises as

$$[S_1, S_2] := \bigcup_{x \in S_1} \bigcup_{y \in S_2} [x, y]$$

Let $\mathcal{L}$ be the set of all the lines of P, and c a cut. The poset P is said to be discrete with respect to *c whenever*

$$\forall x \in P : \exists n \in \mathbb{N}^+ : \forall l \in \mathcal{L} : |[c, x] \cap l| < n \text{ and } |[x, c] \cap l| < n$$

These two structural properties make a poset suitable for representing a process. In order to represent a process, however, it must represent the process of some system, in this case elementary. The relation between the causal net and the net system is formalised by a *labelling function*, that assigns to every element of the causal net, the element of the system it is an occurrence of.

Definition 3.1.3 (Process)**.** *Let $N = (B, E, \mathcal{F}, m_0)$ be a contact-free elementary net system. Let $N' = (B', E', \mathcal{F}')$ be a causal net, and $(P, \leq)$ the poset defined as $P = B' \cup E'$ and $\leq = \mathcal{F}'^+$. Let $\lambda : B' \cup E' \to B \cup E$ be a* labelling function*. A pair (N', λ) is called a* process *of N when the following conditions are satisfied:*

1. $\min(P) \subseteq B'$
2. *P is discrete with respect to* $\min(P)$
3. $\lambda(\min(P)) = m_0$
4. $\lambda(B') \subseteq B$, *and* $\lambda(E') \subseteq E$
5. $\forall x, y \in P : \lambda(x) = \lambda(y) \to x \textbf{ li } y$
6. $\forall e \in E' : \lambda({}^\bullet e) = {}^\bullet\lambda(e)$ *and* $\lambda(e^\bullet) = \lambda(e)^\bullet$

The third condition requires that the initial cut of the net represents the initial marking of the system. The fourth statement requires that events of the causal net represent occurrences of events of the system, and analogously for conditions. The fifth condition requires that different occurrences of the same element of the system are not independent from each other. Indeed, in an elementary system, each marking allows for one single firing of an event. Therefore, two concurrent occurrences of the same event are forbidden. The last requirement states that the labelling function must preserve the flow relation. All direct causal influences of the system must be reported in the process, and the process must not represent additional dependences.

Example 3.1.1. *The causal N' of Figure 3.2 is labelled consistently with the elementary net system N of Figure 3.1. It represents a process run by N. The initial cut of N' is labelled with the initial marking of N. Lines in N'*

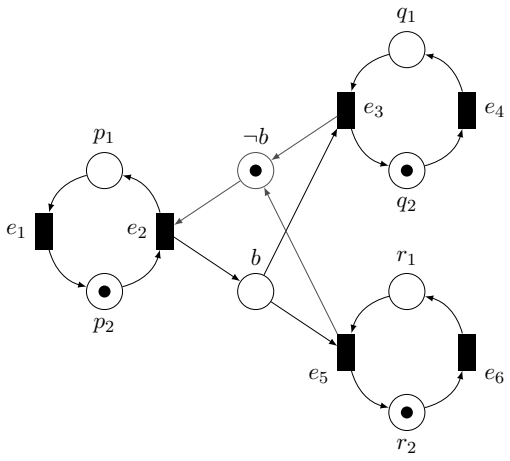

Figure 3.1: An elementary net system N modelling a cycle of one producer and two consumers. Condition b models the situation when a given resource generated by the producer has been delivered. Either of the two consumers $N(\{q_1, q_2\})$, or $N(\{r_1, r_2\})$ can consume the resource. While no resource is delivered, the consumers remain idle. The producer can not output new resources while the previous one has not been consumed. This constraint is represented by condition $\neg b$.

coincide with possible flows of tokens, they represent the flow of information in the system.

The first occurrences of p_1, b, q_1, and r_2 form a cut, they represent that the corresponding marking on the net is reachable from the initial state. However, the sequence (e_2, e_1, e_4) is a firing sequence of the system, and so e_1 could unmark p_1 before e_4 marks q_1. It is not guaranteed that such a marking is ever visited.

The E-cut labelled with e_1, e_3, and e_6 represents the concurrent occurrence of the corresponding events.

Note that the first occurrences of p_1,q_1, b, and r_1 form a B-cut. At the corresponding marking, e_5 is as enabled as e_3. This is not represented in the process, because the firing of e_3 disables e_5. The two events are in conflict, and the process represents only one outcome of the corresponding choice. This shows that the two consumers compete for the resource. In this particular execution only one of them ever consumes it. The other remains idle indefinitely.

Finally, note that without the condition $\neg b$, the system would present a contact after the occurrence of e_1 if neither e_3 nor e_5 have occurred. In this

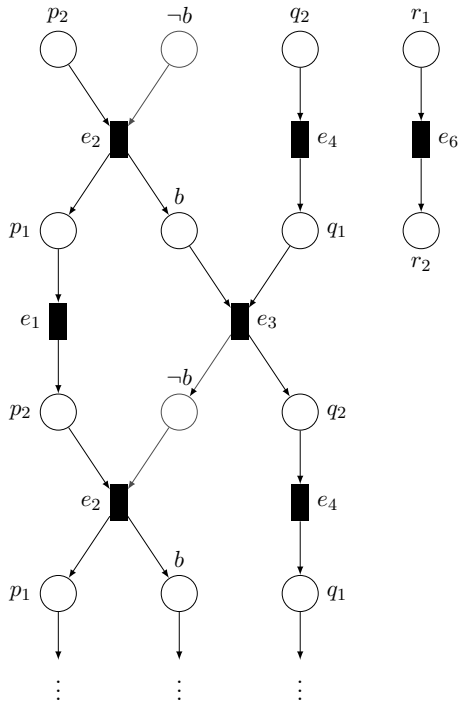

Figure 3.2: A possible process N' run by the system of N of Figure 3.1

situation, the system could not fire e_2, but this dependency would not be represented in the process.

It is to be noted that, in an elementary system, contacts are situations in which a condition has a causal influence on one of its pre-events. This causal dependence can not be represented by the causal net because it is oriented in opposite direction than the flow relation.

These situations can easily be prevented by adding conditions to the net, which do not alter its behaviour. This matter will be covered in the next first section of the next chapter.

Remark 3.1.1. *This chapter gives an interpretation of subsets of the net as subprocesses. In Petri net theory, it is usually required that a subprocess is in itself a process. In this work, the third axiom of Definition 3.1.3 is not required. A subprocess will be a subnet of the causal net, such that, with the labelling function of its containing process, all but the third axiom hold.*

Distributability of processes will be analysed, in the rest of this chapter, on the base of the partial order derived from a causal net. The next section will present the relevant properties of such posets.

3.1.2 Properties of Process Posets

Processes, are defined on the base of causal nets. In order to study how processes can be distributed, this type of net will be the main model at stake. However, results are presented in terms of the partial order derived from it. The nature of the mathematical tools that will allow for analysis of distributability, makes it is more practical to work with an order relation. Nevertheless, the fact that the partial order derives from a causal net has some implications that will be essential in the proofs of the contributions of this work.

Definition 3.1.4 (Combinatorial Poset)**.** *Let $(P, \leq)$ be a partially ordered set. It is said that $(P, \leq)$ is* combinatorial *whenever there exists a covering relation $\prec$ such that the order is its transitive and reflexive closure $\leq = (\prec)^\star$*

The term combinatorial stands for the fact that a covering relation describes a directed graph. By construction, every poset derived from an occurrence net is combinatorial.

When the transitive closure of a net is a partial order, then it will certainly be combinatorial. The relevant restriction, in this sense, is that the net forms a partial order.

Causal nets are on top of that, conflict-free. This condition implies that some configurations are forbidden.

Definition 3.1.5 (Forks and Joins)**.** *Let $(P, \leq)$ be a combinatorial poset, with covering relation $\prec$. Three elements $x, y, z \in P$ form a* fork at *x whenever $x \prec y$, $x \prec z$, and y **co** z. Three $x, y, z \in P$ form a* join at *x whenever $y \prec x$, $z \prec x$, and y **co** z.*

Clearly, in a causal net, forks o joins can only happen at events. Indeed a fork (or join) at a condition would violated the fact that it can have only one immediate successor (predecessor).

As a consequence, partial orders that arise from causal nets will satisfy the following local property.

Definition 3.1.6 (N-dense Poset)**.** *A combinatorial poset $(P, \leq_P)$ is* N-dense *iff $\forall x, y, u, v \in P$ such that $x \leq_P y, x \leq_P u, v \leq_P y, x$ **co**$_P$ v, u **co**$_P$ v,*

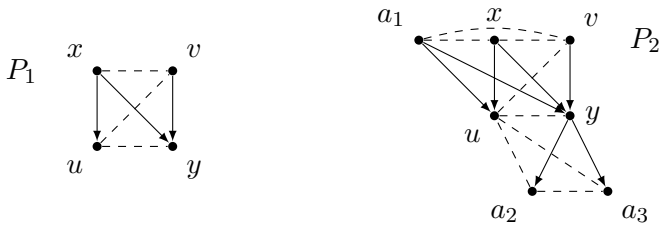

Figure 3.3: Two *combinatorial posets* P_1 and P_2 which are not N-dense. Arrows represent $\prec_{P_i}$, and dashed lines represent $\mathbf{co}_{P_i}$, $i \in \{1, 2\}$. On the left, $P_1 = \{x, y, u, v\}$ with $\leq_{P_1} = \{(x, y), (x, u), (v, y)\}$. It is apparent that (x, v), (u, y), $(u, v) \in \mathbf{co}_{P_1}$, as represented by the dashed lines.

and u $\mathbf{co}_P$ y*, there is an element* $a \in P$ *satisfying* a $\mathbf{co}_P$ u, a $\mathbf{co}_P$ v *and* $x \leq_P a \leq_P y$*.*

Example 3.1.2. *Figure 3.3 shows the following poset:* $P_1 = \{x, y, u, v\}$ *with* $\leq_{P_1} = \{(x, y), (x, u), (v, y)\}$*. It is apparent that* x $\mathbf{co}_{P_1}$ v*,* u $\mathbf{co}_{P_1}$ y *and* u $\mathbf{co}_{P_1}$ v*. The poset* P_1 *can not be interpreted as a causal net. It presents a fork at* x*, and a join at* y*. According to conflict-freeness, both* x *and* y *should be interpreted as events, but then* $x \prec_{P_1} y$ *violates the Petri net axiom that the flow relation can only relate events to conditions, or conditions to events. N-density requires the existence of an element* a *such that* $x \prec_{P_1} a \prec_{P_1} y$*. When this element exists it can be interpreted as a condition.*

The poset P_1 of Figure 3.3 is common in the literature and often referred to as the *N Poset*. It is, in particular, the paradigmatic poset that is not series-parallel. A poset is called *N-free* when it does not contain P_1 as a subposet.

The poset associated with a causal net is trivially combinatorial, and N-dense. These two properties will be fundamental in the development of the results that follow. They will be sufficient to perform the analysis on distributability of the represented process. As a matter of fact, N-density is not necessary. It will be shown that the results hold, for instance, in the poset P_2 of Figure 3.3, although this poset is not N-dense. For the scope of this work, it is satisfactory to be sure that these results can be applied to any poset which represents a process. Note that N-dense posets may contain the N poset (P_1) as a subposet. Hence, N-density is a weaker notion than N-freeness, and the class of N-dense posets is wider than that of series-parallel posets.

It will be necessary, to provide interpretation to the presented results, to assume that the poset representing a process displays a stronger form of density, K-density. In general, and quite trivially, a line and a cut can intersect in at most one element. K-density requires that they do in exactly one element, meaning that each line must intersect every cut.

Definition 3.1.7 (K-density)**.** *A poset is K-dense*[1] *whenever for every line l, and every cut c, it holds that $l \cap c \neq \emptyset$.*

K-density implies N-density. To see this, suppose that a poset $(P, \leq)$ contains the N-poset P_1 of figure 3.3 with $x \prec_P y$. Then no line containing x, and y, can intersect a cut containing u and v. It can be shown that if a poset is N-dense, then it can only fail to be K-dense if it contains a cut of infinite cardinality (see [17]). This, in turn, can only be interpreted, in the elementary case, as process run by a system with either infinite conditions, or infinite events. K-density is therefore considered a reasonable requirement.

3.1.3 Closure Operator based on Concurrency

This chapter is based on the results of [10]. In that work, the authors present a closure operator on the subsets of a poset. This notion of closed set is based on concurrency. Concurrency is defined on the elements of the poset, but it can be extrapolated to its closed subsets. This relation on the closed subsets of the poset expresses that they can be separated from one another. When considered as subprocesses, they are allowed to run in parallel.

In order to do so, the first step is to identify which elements of the partial order are concurrent to the same elements. To this aim, the concurrency relation may be extended to the power set $2^P = \{S \subseteq P\}$ of P as follows.

Definition 3.1.8 (Polarity Induced by **co**)**.** *For any subset $S \subseteq P$, define the* polarity *induced by $\mathbf{co}_P$ as*

$$(\cdot)' : \mathcal{P}(P) \longrightarrow \mathcal{P}(P)$$
$$S \longmapsto S' := \{y \in P \mid \forall x \in S : x \; \mathbf{co}_P \; y\}$$

This operator will henceforth be referred to as a polarity*, and S' as the* polar *of S.*

It is a well known result that applying such a polarity two times yields a closure operator on $\mathcal{P}(P)$. In fact, $((\cdot)', (\cdot)')$ is a Galois connection [18, Ch.V§7].

[1] K holds for 'Kombinatorisch', K-dense was originally intended as combinatorially dense

Definition 3.1.9 (Closure Induced by **co**)**.** *Let $S \subseteq P$, the* closure *induced by $\mathbf{co}_P$ is*

$$(\cdot)'' : \mathcal{P}(P) \longrightarrow \mathcal{P}(P)$$
$$S \longmapsto S'' := (S')'$$

This operator is indeed:

1. extensive $\forall S \in \mathcal{P}(P) : S \subseteq S''$
2. monotone $\forall S_1, S_2 \in \mathcal{P}(P) : (S_1 \subseteq S_2 \Rightarrow S_1'' \subseteq S_2'')$, *and*
3. idempotent $\forall S \in \mathcal{P}(P) : (S'')'' = S''$

From this point on, S'' will be called the closure *of S, and the space $L(P) = \{S'' \mid S \in \mathcal{P}(P)\}$ of closed subsets of P will be studied.*

The empty set $\emptyset$ and the full poset P are trivially closed, and polar to each other. As for any structure defined on a power set, there is a natural ordering of its elements induced by inclusion. In this way, a set precedes another if it is contained in it. When ordered in such a way, $L(P)$ forms again a poset, and as such it is common practice to represent it as a Hasse diagram.

Example 3.1.3. *In Figure 3.4 one can see the closed sets of P_1 from Figure 3.3. Since u $\mathbf{co}_{P_1}$ v and u $\mathbf{co}_{P_1}$ y, then $\{u\}' = \{v, y\}$, and no other element is concurrent to both v and y, so $\{u\}'' = \{u\}$. Consider $\{x\}' = \{v\}$, u $\mathbf{co}_{P_1}$ v implies that $u \in \{x\}''$. In fact $\{x\}'' = \{x, u\}$. $L(P_1)$ is represented as a Hasse diagram.*

It is a well known result [18, Ch.V§7] that, when considering the closure operator associated with a symmetric relation, the resulting collection of closed sets forms a complete lattice. Furthermore, since the **co** relation is also irreflexive, $L(P)$ is actually an *orthocomplemented lattice* or shortly, *ortholattice*. As such, $L(P)$ is endowed with a set of operations related to the order relation induced by inclusion.

It was furthermore shown in [10], that whenever the poset is N-dense, and combinatorial then its associated lattice of closed sets satisfies the orthomodular law. As a matter of fact, the converse also holds. Given a combinatorial poset, it is N-dense whenever the described operator provides an orthomodular lattice [12]. Hence, whenever a poset P has a subposet isomorphic to P_1 from figure 3.3 (p.58), then $L(P)$ will only be orthomodular if there is an element $a \in P$ such that $x \leq_P a$, $a \leq_P y$, and u $\mathbf{co}_P$ a $\mathbf{co}_P$ v. Such a property is called *N-density* [17, Ch.2 §3].

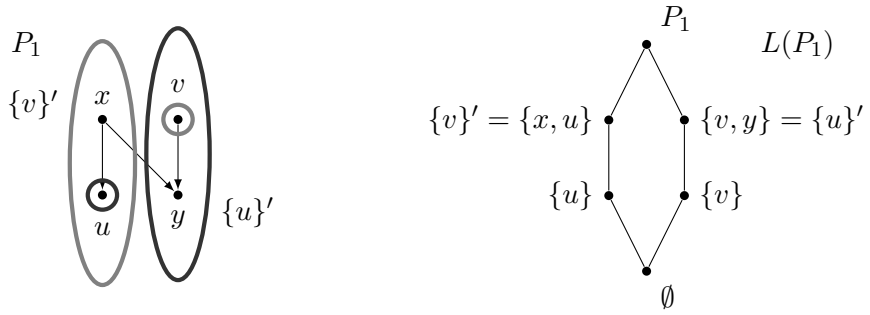

Figure 3.4: Closure Operator induced by concurrency relation on P_1 generates $L(P_1)$. On the left: closed sets as subsets of P_1. On the right: Hasse diagram of $L(P_1)$.

In the next section, it will be show that N-density implies a weaker condition, which will be sufficient for the contributions of this chapter to hold.

3.2 Orthomodular Lattice of Concurrency Closure

Even though this closure operator was originally defined excluding conflict situations [10], the authors then extended their results to include these [6]. In the same line of ideas, the results presented in this work do not consider conflict, as they are intended to be considered further on. Also, this operator is described in the frame of acyclic Petri nets, but the original authors generalise the result to partial orders. One of their core contributions is the identification of a local property of these partial orders, N-density, with a property of the space of subsets obtained from the closure operator, orthomodularity

Analogously, the results of this paper are framed in terms of Petri nets, but presented for more general partially ordered sets, so that the results could be applied to other models of concurrent processes, such as the above mentioned. In fact, instead of N-density and orthomodularity, here weaker notions are considered, so that results are presented in a slightly more general form.

In this work, this closure operator is used for reducing a partial order. The cases in which this reduction preserves the structure of closed sets are characterised, and minimality of the obtained partial order is proven. Intuitively, the obtained partial order is an abstraction of the process, carrying only the information which is available locally. In this sense, it is the skeleton of the interactions between sequential subprocesses. Indeed, all the elements

of the reduced partial order are involved either in a synchronisation, or in a branching, or in both.

3.2.1 N-density, Orthomodularity, and Atomicity

The results of this section can be found in [1].

The study of lattices of closed sets presented in this work will focus on particular elements of these structures. Atoms are the smallest non trivial closed sets.

Definition 3.2.1 (Atom of a Lattice)**.** *An* atom *is a closed set which contains no proper non-empty closed subset. Given $L(P)$, one may consider its set of atoms:*

$$\mathcal{A}(P,\leq) = \mathcal{A}_P := \{A \subseteq P \mid A \neq \emptyset \wedge \ \ \forall S \subseteq A : (S \neq \emptyset \Rightarrow S'' = A)\}$$

In terms of lattices, an atom is an element such that each other element less or equal than it is either the bottom element, or the atom itself:

$$\forall a \in L(P) : (a \in \mathcal{A}_P \Leftrightarrow a \neq \emptyset \wedge (\forall s \in L(P) : (s \leq_{L(P)} a \Rightarrow (s = \emptyset \vee s = a))))$$

A lattice is called atomic *if every element is greater or equal to some atom:*

$$L(P) \text{ is atomic} \Leftrightarrow \forall s \in L(P) : (\exists a \in \mathcal{A}_P : a \leq_{L(P)} s)$$

In an atomic lattice, one can consider the set of atoms under any given element:

$$\forall s \in L(P) : \mathcal{A}_s := \{a \in \mathcal{A}_P \mid a \leq_{L(P)} s\}$$

An atomic lattice is said to be atomistic *if every element can be expressed as the join of the atoms under it:*

$$L(P) \text{ is atomistic } \Leftrightarrow \forall s \in L(P) : s = \bigvee_{a \in \mathcal{A}_s}^{L(P)} a$$

Non-atomic lattices are those in which at least one closed set contains an infinite sequence of closed sets such that each is properly contained in the previous one. An atomistic lattice is one in which each closed set can be uniquely characterised by the atoms contained in it. In an orthomodular lattice, if a set properly contains another, then the former must intersect the orthocomplement of the latter. Each orthomodular lattice is atomistic, and every atomistic lattice must be atomic [39, Ch.3 §10] The following examples should help clarify these notions.

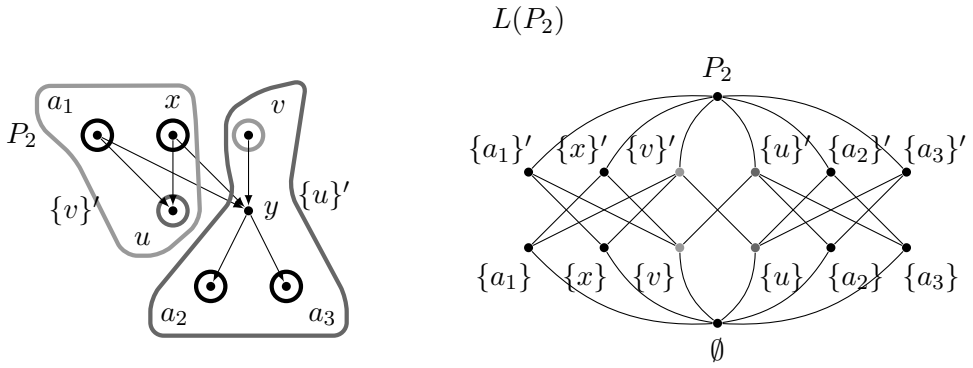

Figure 3.5: On the left: Some closed sets of P_2. On the right: Hasse diagram of $L(P_2)$

Example 3.2.1. *Consider the poset P_1 as previously defined. Figure 3.4 shows the atoms of $L(P_1)$: $\mathcal{A}_{P_1} = \{\{u\}, \{v\}\}$. Obviously, $L(P_1)$ is atomic. However, $\{x, u\}$ can not be distinguished from $\{u\}$ in terms of atoms, so it is not atomistic. Indeed $\mathcal{A}_{\{x,u\}} = \mathcal{A}_{\{u\}} = \{\{u\}\}$, so $\bigvee_{a\in\mathcal{A}_{\{u\}}}^{L(P_1)} a = \{u\} = \bigvee_{a\in\mathcal{A}_{\{x,u\}}}^{L(P_1)} a \neq \{x, u\}$. Since it is not atomistic, neither is it orthomodular, and in fact $\{u\} \subseteq \{x, u\}$ but $\{u\}' \cap \{x, u\} = \{v, y\} \cap \{x, u\} = \emptyset$. On Figure 3.5 the atoms of P_2 are depicted, together with a couple of their complements. Clearly, $L(P_2)$ is atomic. On the Hasse diagram of $L(P_2)$, all elements are above different sets of atoms, hence it is atomistic. However, it is not orthomodular. Indeed $\{u\}$ is a closed set contained in $\{v\}'$, but $\{v\}' \cap \{u\}' = \emptyset$, so $\{u\} \vee^{L(P_2)} (\{u\}' \wedge^{L(P_2)} \{v\}') = \{u\} \vee^{L(P_2)} \emptyset = \{u\} \neq \{v\}'$*

Being orthomodular is a stronger notion than being atomistic. In both cases, one can retrieve any element as the join of the atoms under it, but under orthomodularity, it is sufficient to consider pairwise orthogonal elements. This follows as a direct consequence of Axiom 4. of Definition 2.3.6, and is therefore only stated as a remark.

Remark 3.2.1. *Let L be an orthomodular lattice. For every element $s \in L$, consider $\downarrow\{s\}$, and let $c \subseteq \downarrow\{s\}$ be a set of pairwise orthogonal elements which is maximal in $\downarrow\{s\}$. This reads as $\forall x, y \in c : (x \neq y) \rightarrow x \perp_L y$, and $\forall z \notin c : (z \not\leq s)$ or $(\exists x \in c : z \not\perp_L x)$. Then*

$$s = \bigvee_{x\in c} x$$

In particular, c can be composed of atoms of L.

So an atomistic lattice requires all the atoms under an element to retrieve it as their join, but in an orthomodular lattice it is sufficient to consider cliques of orthogonality.

3.2.2 Inheritance of the Process Ordering

The results of this section can be found in [1].

This section is dedicated to showing that the order of a poset can be extended to the collection of its atomic closed sets. To this aim, some properties of atoms, seen as subsets of the poset are studied.

As a first remark, note that all atoms are pairwise disjoint.

Proposition 3.2.1. $\forall A_1, A_2 \in \mathcal{A}_P : (A_1 \cap A_2 \neq \emptyset \Rightarrow A_1 = A_2)$

Proof. If $A_1 \cap A_2 \neq \emptyset$ then $(A_1 \cap A_2) \subseteq A_1$ implies that $\emptyset \neq (A_1 \cap A_2)'' \subseteq A_1'' = A_1$. Since A_1 is an atom, it contains no proper closed subset. Hence $(A_1 \cap A_2)'' = A_1$. Analogously, $(A_1 \cap A_2)'' = A_2$, so $A_1 = A_2$ □

Another interesting property of atoms is that all the elements inside one of them relate identically to the elements outside of it, either by the order relation, or the concurrency one. Such a property of subsets is rather common in the literature, and referred to diversely according to the subject (See *D-autonomous* sets defined on pre-orders in [43] for additional references). This property is proven in Propositions 3.2.2 through 3.2.5. This will subsequently allow for defining a consistent order on the set of atoms.

In the case of concurrency, the result is quite straightforward:

Proposition 3.2.2. *Let* $A \in \mathcal{A}_P, x \in A, y \in P \setminus A : y \ \mathbf{co}_P \ x$. *Then* $\forall z \in A : y \ \mathbf{co}_P \ z$.

Proof. Suppose $\exists z \in A : y \ \mathbf{li}_P \ z$. Then $y \in \{x\}' \setminus \{z\}' \Rightarrow \{x\}'' \subseteq \{y\}' \wedge \{z\}'' \not\subseteq \{y\}'$. But $x, z \in A \Rightarrow \{x\}'' = \{z\}'' = A$, which is a contradiction. □

The following result will serve to prove the counterpart of Proposition 3.2.2 for the ordering relation.

Proposition 3.2.3. *Closed sets are convex. Formally:*

$$\forall S \subseteq P, \ \forall x, y \in S'' : \ (x \leq z \leq y) \Rightarrow (z \in S'')$$

Proof. Note that $\{x, y\}'' \subseteq S'' \ \forall x, y \in S''$. Let $x \leq z \leq y$, and suppose $\exists v \in \{x, y\}'$ such that $z \ \mathbf{li}_P \ v$. Then, by transitivity, $z \leq v \Rightarrow x \leq v$ and $v \leq z \Rightarrow v \leq y$. Hence, it must be $\forall v \in \{x, y\}' : \ z \ \mathbf{co}_P \ v$. So $z \in \{x, y\}'' \subseteq S''$. □

With these results, it can be shown that if an element of an atom is ordered before an element outside of it, then all the elements of this atom must be ordered accordingly.

Proposition 3.2.4. *Let $A \in \mathcal{A}_P, x \in A, y \in P \setminus A$. If $x \leq_P y$, then for each $z \in A : z \leq_P y$.*

Proof. Let $z \in A$, and suppose $z \ \mathbf{co}_P \ y$, then by Proposition 3.2.2 $x \ \mathbf{co}_P \ y$ which is impossible. So either $z \leq_P y$ or $y \leq_P z$. Now suppose, $y \leq_P z$. Then, A being a closed set, by Proposition 3.2.3 it must be convex, so $y \in A$ contradicting the hypothesis. Hence $\forall z \in A : \ z \leq_P y$. □

An analogous proof leads to the following result.

Proposition 3.2.5. *Let $A \in \mathcal{A}_P, x \in A, y \in P \setminus A$. If $y \leq_P x$, then for each $z \in A : y \leq_P z$.*

For the sake of clarity, these results are summarised as follows. Let $A \in \mathcal{A}_P, \ y \in P \setminus A$:

1. $(\exists x \in A : x \ \mathbf{co}_P \ y) \Rightarrow (\forall z \in A : z \ \mathbf{co}_P \ y)$
2. $(\exists x \in A : x \leq_P y) \Rightarrow (\forall z \in A : z \leq_P y)$
3. $(\exists x \in A : y \leq_P x) \Rightarrow (\forall z \in A : y \leq_P z)$

Under these conditions, $\leq_{\mathcal{A}_P}$ may be defined as follows: Let $A_1, A_2 \in \mathcal{A}_P$. Then $A_1 \leq_{\mathcal{A}_P} A_2 :\Leftrightarrow: \exists x \in A_1, \exists y \in A_2 : x \leq_P y$. Then clearly $A_1 \leq_{\mathcal{A}_P} A_2 \Leftrightarrow \forall x \in A_1, \forall y \in A_2 : x \leq_P y$. Furthermore, $A_1 \ \mathbf{co}_{\mathcal{A}_P} \ A_2 \Leftrightarrow A_1 \perp_P A_2$.

$\leq_{\mathcal{A}_P}$ is rather trivially an order on $\mathcal{A}_P$. It inherits reflexivity and transitivity from $\leq_P$, and if it were not antisymmetric, neither would $\leq_P$. As a matter of fact, any choice function f, defined as follows, is an order embedding:

$$\begin{aligned} f : \mathcal{A}_P &\longrightarrow P \\ A &\longmapsto f(A) := x \in A \end{aligned} \tag{3.1}$$

And so, $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ can be embedded into $(P, \leq_P)$. Indeed, provided f exists, its injectiveness comes as a consequence of atoms being pairwise disjoint (see

Proposition 3.2.1). This justifies the idea that $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ is a reduced version of $(P, \leq_P)$. Intuitively, the reduction can be seen as a collapsing of the atoms. As the next section will show, atoms are sufficient to recover the lattice of closed sets. On the other hand, Propositions 3.2.2 to 3.2.5 imply not only that atoms are totally ordered subsets, but also that no branching occurs at any of their elements. Indeed, no element of an atom has more than one predecessor and one successor. In particular, if a line of a poset intersects an atom, then the whole atom must be contained in the line.

Proposition 3.2.6. *Let $A \in \mathcal{A}_P$, and l be a line of P such that $l \cap A \neq \emptyset$, then $A \subseteq l$.*

Proof. Let $x \in A \cap l$, and suppose $\exists z \in A \setminus l$. Maximality of l implies that there must be a $y \in l : z \ \mathbf{co}_P \ y$. If $y \notin A$, then $x, z \in A$ with $x \ \mathbf{li}_P \ y$ and $z \ \mathbf{co}_P \ y$ contradicts Proposition 3.2.2. It must be that $y \in A$. Since A is an atom, in particular it must be the closure of all its elements, $\{x\}'' = \{z\}''$ and so $\{x\}' = \{z\}'$. But this is a contradiction, since $z \ \mathbf{co}_P \ y \rightarrow y \in \{z\}'$, and $x \ \mathbf{li}_P \ y \rightarrow y \notin \{x\}'$. □

This should clarify that the inner structures of atoms provide no information on the interactions between the different sequential subprocesses represented in the poset. Instead, all this information is condensed in their outer structure: the reduced poset $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$. For instance, a totally ordered set, representing a single sequential process will consist of one single atom, and its reduced version will then be a single isolated element. A set of n non interacting sequential processes would consist of the corresponding n atoms, leading to a reduced version of n pairwise concurrent elements. Naturally, the structure of the reduced poset would grow more complex as sequential processes interact, stepping away from these trivial examples.

3.2.3 Preservation of Concurrency

The results of this section can be found in [1].

This section will prove that the concurrency structure of $(P, \leq_P)$ is preserved in $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ by means of orthogonality, and under which conditions this statement fully holds.

The results presented in this section rely heavily on the fact that an atomistic lattice is uniquely determined by its set of atoms, and their orthogonality relation. This implies that $L(\mathcal{A}_P)$ and $L(P)$ are isomorphic. This will be proved by showing that $L(\mathcal{A}_P)$ is always atomistic, and after inspecting the orthogonality relation in both lattices, building the actual isomorphism.

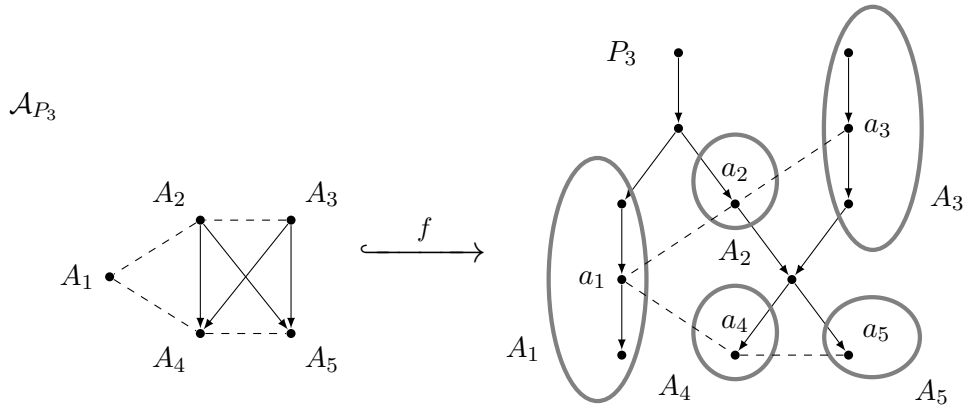

Figure 3.6: On the left, $\mathcal{A}_{P_3} = \mathcal{A}(P_3, \leq_{P_3})$, obtained from P_3 on the right. Consider $f : \mathcal{A}_{P_3} \longrightarrow P_3$ such that $f(A_i) = a_i \forall i \in 1, .., 5$. The elements in its image have been labelled showing the actual embedding of $\mathcal{A}_{P_3}$ into P_3. Clearly, f both preserves and reflects concurrency. Note that since $L(P_3)$ is atomistic, it is isomorphic to $L(\mathcal{A}_{P_3})$.

The idea behind the following result arises from the observation that $\forall A_1, A_2 \in \mathcal{A}_P : \{A_1\} \perp_{\mathcal{A}_P} \{A_2\} \Leftrightarrow A_1 \ \mathbf{co}_{\mathcal{A}} \ A_2 \Leftrightarrow A_1 \perp_P A_2$, which can in fact be extended to arbitrary subsets of $\mathcal{A}_P$.

Proposition 3.2.7. *Let* $B_1, B_2 \subseteq \mathcal{A}_P$. *Then* $B_1 \perp_{\mathcal{A}_P} B_2 \Leftrightarrow (\bigcup_{A \in B_1} A) \perp_P (\bigcup_{A \in B_2} A)$

Proof. $B_1 \perp_{\mathcal{A}_P} B_2 \Leftrightarrow B_1 \times B_2 \subseteq \mathbf{co}_{\mathcal{A}_P} \Leftrightarrow \forall A_1 \in B_1, \forall A_2 \in B_2 : \ A_1 \ \mathbf{co}_{\mathcal{A}_P} A_2 \Leftrightarrow \forall A_1 \in B_1, \forall A_2 \in B_2 : \ A_1 \perp_P A_2 \Leftrightarrow (\bigcup_{A \in B_1} A) \perp_P (\bigcup_{A \in B_2} A)$ □

$L(\mathcal{A}_P)$ can now be defined in an analogous manner to $L(P)$, to the point of showing that $L(\mathcal{A}_P)$ can be embedded into $L(P)$. The conditions under which this embedding is actually an isomorphism will then be studied.

To this aim, the following two propositions show that $L(\mathcal{A}_P)$ is an atomistic lattice.

Proposition 3.2.8. $\forall A \in \mathcal{A}_P : \ \{A\} \in L(\mathcal{A}_P)$

Proof. Suppose $\exists A' \in \mathcal{A}_P : \ A' \in \{A\}''$. Then $\{A\}' \subseteq \{A'\}'$, and clearly $A, A' \subseteq P : \ A' \subseteq (A')'$. So $A' = A''' = ((A')')' \subseteq (A')' = A'' = A$, and since atoms contain no proper closed subsets, it must be that either $A' = \emptyset$ or $A' = A$. □

This trivially implies that $L(\mathcal{A}_P)$ is atomistic.

Proposition 3.2.9. *$L(\mathcal{A}_P)$ is* atomistic*, formally:*

$$\forall B \in L(\mathcal{A}_P) : B = \bigvee_{A \in B} \{A\} = (\bigcup_{A \in B} \{A\})''$$

Or equivalently: $\forall B_1, B_2 \in L(\mathcal{A}_P)$:
$(\{A \in \mathcal{A}(\mathcal{A}_P, \leq_{\mathcal{A}_P}) \mid \{A\} \subseteq B_1\} = \{A \in \mathcal{A}(\mathcal{A}_P, \leq_{\mathcal{A}_P}) \mid \{A\} \subseteq B_2\}) \Rightarrow B_1 = B_2$

Proof. From Proposition 3.2.8, every element of $\mathcal{A}_P$ constitutes a closed singleton, which can clearly be nothing but an atom. Obviously, no other subset can be an atom, and as a matter of fact, two different closed sets will always differ in at least one atom. □

At this point, it is shown that $L(\mathcal{A}_P)$ can be embedded into $L(P)$. This is achieved by defining a map between them, and showing that it is an order homomorphism in Proposition 3.2.10, and that it is injective in Proposition 3.2.11.

Definition 3.2.2. *The morphism ϕ will turn out to be an injective order homomorphism*

$$\begin{aligned} \phi : L(\mathcal{A}_P) &\longrightarrow L(P) \\ B &\longmapsto \phi(B) := \bigvee_{A \in B} A = (\bigcup_{A \in B} A)'' \end{aligned}$$

Next proposition proves that ϕ preserves and reflects the order of the corresponding lattices.

Proposition 3.2.10. $\forall B_1, B_2 \in L(\mathcal{A}_P) : B_1 \leq_{L(\mathcal{A}_P)} B_2 \Leftrightarrow \phi(B_1) \leq_{L(P)} \phi(B_2)$

Proof. Let $\forall i = 1, 2 : S_i = \bigcup_{A \in B_i} \{A\} \subseteq P$. Then clearly $B_1 \leq_{L(\mathcal{A}_P)} B_2 \Leftrightarrow B_1 \subseteq B_2 \Leftrightarrow S_1 \subseteq S_2 \Leftrightarrow \phi(B_1) = S_1'' \subseteq S_2'' = \phi(B_2) \Leftrightarrow \phi(B_1) \leq_{L(P)} \phi(B_2)$. □

This implies that ϕ is an order homomorphism, which reflects order. So if it were injective it would be an order embedding. This is confirmed by proving the injectiveness of ϕ.

Proposition 3.2.11. *Let $B_1, B_2 \in L(\mathcal{A}_P)$ such that $\phi(B_1) = \phi(B_2)$, then $B_1 = B_2$.*

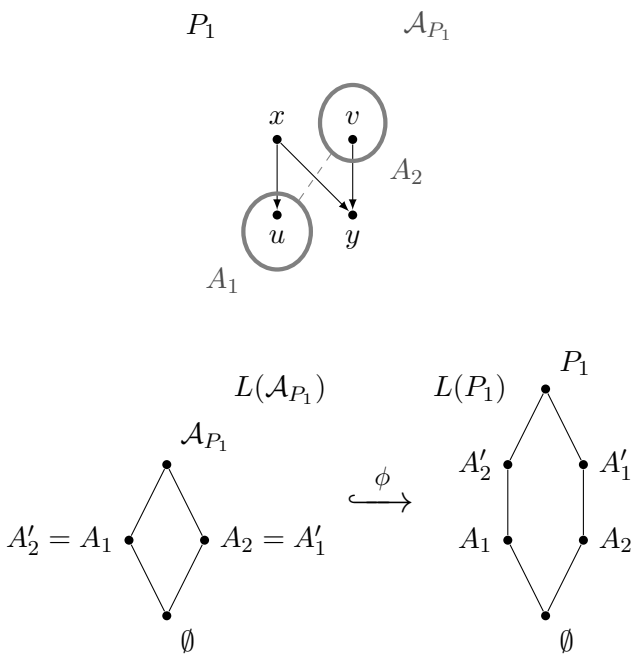

Figure 3.7: On the left, the poset P_1, with its two only atoms drawn: $A_1 = \{u\}$ and $A_2 = \{v\}$. Clearly A_1 $\mathbf{co}_{\mathcal{A}_{P_1}}$ A_2 so $(\mathcal{A}_{P_1}, \leq_{\mathcal{A}_{P_1}}) = (\{A_1, A_2\}, \emptyset)$. On there, the corresponding lattice $L(\mathcal{A}_{P_1})$, which is embeddable into $L(P_1)$, on the right. Note that $A_2 \leq_{L(P_1)} A'_1 \Rightarrow A_2 \perp_{L(P_1)} A_1$, so the embedding ϕ preserves orthogonality.

Proof. Let $B_1 \neq B_2$, and suppose, without loss of generality, $\exists A \in B_1 \setminus B_2$ so that $B_1 \not\subseteq B_2$. Then $B_1 \not\leq_{L(\mathcal{A}_P)} B_2$, hence by Proposition 3.2.10, $\phi(B_1) \not\leq_{L(P)} \phi(B_2)$, so by reflexivity of $\leq_{L(P)}$, $\phi(B_1) \neq \phi(B_2)$. □

Thus, ϕ^{-1} is a well defined function on the codomain of ϕ. As a matter of fact, provided a closed set of the codomain, it simply returns the set formed by the atoms under it : $\phi^{-1}(S) = \{A \in \mathcal{A}_P \mid A \leq_{L(P)} S\}$

It can now be positively stated that $L(\mathcal{A}_P)$ can be embedded into $L(P)$:

$$\phi : L(\mathcal{A}_P) \hookrightarrow L(P)$$

The goal of this section, however, is to go further and have $L(\mathcal{A}_P)$ and $L(P)$ isomorphic, for which ϕ is required to be surjective. In general this is not the case, as depicted in Figure 3.7. The fact preventing ϕ from being surjective, seems to be that A'_2 (respectively A'_1) cannot be differentiated from A_1 (A_2)

solely in terms of atoms. This observation leads naturally to the following result:

Proposition 3.2.12. *ϕ is surjective iff $L(P)$ is atomistic.*

Proof. Clearly, if ϕ is surjective, every element $S \in L(P)$ is the image of some $B \in L(\mathcal{A}_P) :\ \phi(B) = S$. Hence $S = \bigvee_{A \in B} A = \bigvee_{A \in \mathcal{A}_P : A \leq_{L(P)} S} A$, and so $L(P)$ must be atomistic. Conversely, if $L(P)$ is atomistic, then $\forall S \in L(P) : S = \bigvee_{A \in \mathcal{A}_P : A \leq_{L(P)} S} A$, and so $B = \{A \in \mathcal{A}_P \mid A \leq_{L(P)} S\} \in L(\mathcal{A}_P)$ is such that $\phi(B) = S$. □

Naturally, one would like to characterise the posets $(P, \leq)$ for which $L(P)$ is atomistic, in this sense it seems clear that a necessary and sufficient condition will be:

$$\forall S \subseteq P : (\exists A_1 \subseteq P : A_1'' \subsetneq S'') \Rightarrow (\exists A_2 \subseteq P : A_2'' \subsetneq S'' \text{ and } A_1'' \neq A_2'') \quad (3.2)$$

Note that this condition is strictly weaker than N-density, which in turn is weaker than N-freeness. For instance, poset P_2 of Figures 3.3 and 3.5 satisfies it, although it is not N-dense, (and therefore neither N-free). In particular, this condition is satisfied by any partial order derived from a causal net. When $L(P)$ is atomistic, ϕ is an order isomorphism. Furthermore, when this is the case, Proposition 3.2.14 will prove that ϕ is even an orthocomplemented lattice isomorphism, preserving not only order, but orthogonality as well. Propositions 3.2.15 and 3.2.16 will then show that ϕ preserves lattice operations.

In order to do this, the following technical result is required.

Proposition 3.2.13. *If $L(P)$ is atomistic, then*

$$\forall S \in L(P) :\ S' = (\bigcup_{A \in \mathcal{A}_P : A \perp_P S} A)''$$

Proof. Clearly $\forall A \in \mathcal{A}_P : A \perp_P S \Rightarrow A \subseteq S'$, then $\bigcup_{A \in \mathcal{A}_P : A \perp_P S} A \subseteq S'$, and since S' is a closed set, by monotonicity of the closure operator, it holds that $(\bigcup_{A \in \mathcal{A}_P : A \perp_P S} A)'' \subseteq S'$.

Now suppose $(\bigcup_{A \in \mathcal{A}_P : A \perp_P S} A)'' \subsetneq S'$, since both sets are closed, and $L(P)$ is atomistic, there must exist an atom $A_0 \in S' \setminus (\bigcup_{A \in \mathcal{A}_P : A \perp_P S} A)''$, but $A_0 \in S' \Rightarrow A_0 \perp_P S$, so $A_0 \in \bigcup_{A \in \mathcal{A}_P : A \perp_P S} A \subseteq (\bigcup_{A \in \mathcal{A}_P : A \perp_P S} A)''$ which is impossible. □

The next proposition shows that ϕ preserves orthocomplementation.

Proposition 3.2.14. *If $L(P)$ is atomistic then $\forall B \in L(\mathcal{A}_P) :\ \phi(B') = \phi(B)'$*

Proof. Let $S = \bigcup_{A \in B} \{A\} \subseteq P$, and note that $A \in B' \Leftrightarrow A \perp_P S$. Then $\phi(B') = (\bigcup_{A \in B'} A)'' = (\bigcup_{A \perp_P S} A)'' = (S')'' = (S'')' = ((\bigcup_{A \in B} A)'')' = \phi(B)'$. □

Therefore, ϕ is an ortholattice isomorphism, hence $L(P) \simeq L(\mathcal{A}_P)$. This is confirmed by the two following results.

Proposition 3.2.15. $\forall B_1, B_2 \in L(\mathcal{A}_P) : \phi(B_1 \vee^{L(\mathcal{A}_P)} B_2) = \phi(B_1) \vee^{L(P)} \phi(B_2)$

Proof. At this point it is clear that, since $L(\mathcal{A}_P)$ is atomistic, $\forall B_1, B_2 \in L(\mathcal{A}_P) : \phi(B_1 \vee^{L(\mathcal{A}_P)} B_2) = \phi(\bigvee^{L(\mathcal{A}_P)}_{A \in B_1 \cup B_2} \{A\}) = \bigvee^{L(P)}_{A \in B_1 \cup B_2} A = \bigvee^{L(P)}_{A \in B_1} A \vee^{L(P)} \bigvee^{L(P)}_{A \in B_2} A = \phi(B_1) \vee^{L(P)} \phi(B_2)$. □

The following result is however not as trivial, and requires $L(P)$ to be atomistic as well.

Proposition 3.2.16. $\forall B_1, B_2 \in L(\mathcal{A}_P) : \phi(B_1 \wedge^{L(\mathcal{A}_P)} B_2) = \phi(B_1) \wedge^{L(P)} \phi(B_2)$

Proof. First, note that in an atomistic lattice $L : x \wedge^L y = \bigvee^L_{a \in \mathcal{A}_{x \wedge^L y}} a = \bigvee^L_{a \in \mathcal{A}_x \cap \mathcal{A}_y} a$. So clearly, since both $L(\mathcal{A}_P)$ and $L(P)$ are atomistic, $\forall B_1, B_2 \in L(\mathcal{A}_P) : \phi(B_1 \wedge^{L(\mathcal{A}_P)} B_2) = \phi(\bigvee^{L(\mathcal{A}_P)}_{A \in B_1 \cap B_2} \{A\}) = \bigvee^{L(P)}_{A \in B_1 \cap B_2} A = (\bigvee^{L(P)}_{A \in B_1} A) \wedge^{L(P)} (\bigvee^{L(P)}_{A \in B_2} A) = \phi(B_1) \wedge^{L_P} \phi(B_2)$. □

3.3 Minimal Process representation

In this section, the characteristics of constructed poset of atoms will be analysed. A notion of process abstraction will be formalised, and it will be shown that the partial order of atoms is the coarsest abstraction which preserves all the concurrency of the process it is built on. This motivates the fact that this representation depicts the information on how the process can be distributed.

3.3.1 Coarsest Process Abstraction

Suppose that the given poset is the specification of a concurrent process that has to be distributed among a given set of active resources, or components. Atomic closed sets represent 'local' subprocesses, they must be sequential. If an element of an atom is assigned a given component, all the other elements in the same atom will be required to belong to the same component. Suppose

several atoms are assigned the same component, in this interpretation the subprocess corresponding to the closure of their union must be entirely assigned to the same component. Suppose the process is maximally distributed, in the sense that two orthogonal atoms are always assigned different components. Then the reduced poset provides a specification of the causal dependencies among the subprocesses represented by the atoms, which can be interpreted as the flow of information among the components of the system.

This motivates the idea that the reduced poset is an abstraction of the process. In the literature there are many different notions of abstraction. Here a formalisation is proposed, supported by the fact that the lines of the two representations of the process are in bijection. In this sense, an abstraction of a process will not only be an induced subposet, but it will be required to represent, and distinguish all its sequential subprocesses. Note that the following definition is not standard, it reflects the view of the author of this work.

Definition 3.3.1 (Abstraction of a Poset)**.** *Let $(P, \leq), (P', \leq')$ be two posets, and consider $\mathcal{L}$, and $\mathcal{L}'$, their respective sets of lines. $(P', \leq')$ will be called an* abstraction *of $(P, \leq)$ whenever $(P', \leq')$ is an induced subposet of $(P, \leq)$, $\mathcal{L}' = \{l \cap P' \mid l \in \mathcal{L}\}$, and is isomorphic to $\mathcal{L}$.*

With this definition, it can be shown, not only that the reduced poset is an abstraction of the specified poset, but also the coarsest one. In order to do so, however, the following results will consider only K-dense posets. First, it is shown that every line of a K-dense poset intersects some atom.

Proposition 3.3.1. *Let $(P, \leq)$ be a combinatorial K-dense poset, and l be a line. Let C be an arbitrary cut in $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$, namely a set of pairwise orthogonal atoms, which is maximal. Then $\exists a \in C : a \cap l \neq \emptyset$.*

Proof. Assume there exists a choice function that associates to each $a \in C$, an $x_a \in a$. Then for each pair $a, b \in C$, it holds that $x_a \ \mathbf{co}_P \ x_b$. Hence, the set $c = \{x_a \mid a \in C\}$ is a co-set. Suppose that c is not maximal, then $\exists y \in P : (\forall x_a \in c : y \ \mathbf{co}_P \ x_a)$, and so $c \subseteq \{y\}'$. In fact, out of Proposition 3.2.2, it must hold that $(\bigcup_{a \in C} a) \subseteq \{y\}'$, but then $\{y\}'' \subseteq (\bigcup_{a \in C} a)'$. Since $\{y\}'' \in L(P)$, there must be some atom a_y contained in it. This implies that $a_y \subseteq C'$ which, in turn, imposes that $\forall a \in c : a_y \perp_{L(P)} a$, or equivalently $\forall a \in c : a_y \ \mathbf{co}_{\mathcal{A}_P} \ a$. This last statement contradicts maximality of C. Hence c must be maximal, and so it is a cut. Out of K-density, $l \cap c \neq \emptyset$, and so finally $\exists a \in C : l \cap a \neq \emptyset$. □

It will be useful to consider the following technical result.

Proposition 3.3.2. *Let $(P, \leq)$ be a combinatorial and K-dense poset, let $x \in P$, and l be a line of P such that $x \in l$. Then there is an atom $a \in \mathcal{A}_P$, such that $a \subseteq l \cap \{x\}''$.*

Proof. Since $L(P)$ is orthomodular, then there must be a set A_x of pairwise orthogonal atoms in $\mathcal{A}_{\{x\}''}$ such that $\{x\}'' = \bigvee_{a \in A_x} a$. A_x is a co-set in $\mathcal{A}_P$, so it can be extended to a cut C. Out of Proposition 3.3.1 there must be an $a \in C$ such that $a \cap l \neq \emptyset$. In fact, Proposition 3.2.6 shows that $a \subseteq l$. Now suppose that $a \notin A_x$, then since C is a cut, $\forall b \in A_x : a \perp_{\mathcal{A}_P} b$, so $\forall b \in A_x : b \leq_{L(P)} a'$, and so must also their supremum $\{x\}'' = \bigvee_{b \in A_x} b \leq_{L(P)} a'$. Hence, $a \subseteq \{x\}'$, meaning that $\exists y \in a : x \ \mathbf{co}_P \ y$, but $a \subseteq l$ implies $x, y \in l$, a contradiction. Therefore it must be $a \in A_x$. □

It will also be required to show that every two different lines are distinguished by a pair of atoms.

Proposition 3.3.3. *Let $(P, \leq)$ be a combinatorial K-dense poset, and l_1, l_2 be two distinct lines. Then $\exists a_1, a_2 \in \mathcal{A}_P : a_1 \cap l_1 \neq \emptyset$ and $a_1 \cap l_2 = \emptyset$ and $a_2 \cap l_1 = \emptyset$ and $a_2 \cap l_2 \neq \emptyset$, and $a_1 \perp_{L(P)} a_2$*

Proof. $l_1 \neq l_2$, so suppose, without loss of generality, that $\exists x \in l_1 \setminus l_2$. Clearly, $\exists y \in l_2 : y \ \mathbf{co}_P \ x$, and obviously, it must satisfy that $y \notin l_1$. Now $\{y\}'' \perp_{L(P)} \{x\}''$. Let C_y, and C_x be two maximal sets of pairwise orthogonal atoms, contained respectively in $\{y\}''$, and $\{x\}''$. Then $\forall a_x \in C_x : \forall a_y \in C_y : a_x \perp_{L(P)} a_y$. Hence $C_x \cup C_y$ are a co-set in $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$, and so there must be a cut C containing it. From Proposition 3.3.1 there must be $a_1, a_2 \in C$, and $x_1 \in a_1$, $x_2 \in a_2$ such that $x_1 \in l_1 \cap a_1$, and $x_2 \in l_2 \cap a_2$. Note that $x_1 \in l_1$ implies that $x_1 \ \mathbf{li}_P \ x$. Since $(P, \leq)$ is K-dense, it is also N-dense, and so $L(P)$ is orthomodular. C_x is, among those contained in $\{x\}''$, a maximal set of pairwise orthogonal elements of $L(P)$. As a consequence, $\{x\}'' = (\bigcup_{a \in C_x} a)''$, as stated in Remark 3.2.1. Suppose $a_1 \in C \setminus C_x$, then it must hold that $\forall a \in C_x : a_1 \in C_x'$, but then $a_1 \in \bigcap_{a \in C_x} a' = (\bigcup_{a \in C_x} a)' = \{x\}'$. So $\forall z \in a_1 : x \ \mathbf{co}_P \ z$, and in particular $x_1 \ \mathbf{co}_P \ x$, a c contradiction. Hence it must be $a_1 \in C_x$. A similar argument leads to $a_2 \in C_y$. Hence, $a_1 \subseteq \{x\}''$, and $a_2 \subseteq \{y\}''$. $\{x\}'' \perp_{L(P)} \{y\}''$ implies that $\{x\}'' \cap \{y\}'' = \emptyset$, and so $a_1 \neq a_2$. Furthermore, $a_2 \subseteq \{y\}''$ implies that $\{y\}' \subseteq a_2'$, and $\{x\}'' \perp_{L(P)} \{y\}''$ implies that $\{x\}'' \subseteq \{y\}'$. Finally, $a_1 \subseteq \{x\}'' \subseteq \{y\}' \subseteq a_2'$, so $a_1 \perp_{L(P)} a_2$. □

These results show that the lattice of closed sets holds information about the interactions between the sequential subprocesses. In some sense this information is complementary to the ordering, which determines the inner structure of these subprocesses.

In particular, it can be shown that a K-dense poset, and its reduced poset have the same set of lines.

Proposition 3.3.4. *Let $(P, \leq)$ be a combinatorial and K-dense poset, and consider $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$. Let $f : \mathcal{A}_P \hookrightarrow P$ the embedding as in Equation (3.1), and call $P' = f(\mathcal{A}_P)$ and consider $\mathcal{L}$, and $\mathcal{L}'$ the respective sets of lines of P, and $\mathcal{A}_P$. Then*

$$\begin{aligned} \pi : \mathcal{L} &\to \mathcal{L}' \\ l &\mapsto f^{-1}(l \cap P') \end{aligned}$$

is well defined and bijective.

Proof. First, note that as since f is an embedding, then $f^{-1}\restriction_{P'}$ is an isomorphism. From Proposition 3.3.1, every line l must intersect some atom, and out of Proposition 3.2.6 that atom must be contained in l. Then clearly $a \subseteq l$ implies that $f(a) \in l \cap P'$, so that every line of P intersects P'.

Note that, as an order embedding, f reflects the order. Then $\forall l \in \mathcal{L}$, $l \cap P'$ is a li-set in P, and so $f^{-1}(l \cap P')$ is a li-set in $\mathcal{A}_P$. Suppose it is not maximal, then $\exists a \in (\mathcal{A}_P \setminus f^{-1}(l \cap P')) : \forall x \in f^{-1}(l \cap P') : a \ \mathbf{li}_P \ x$. Since, f also preserves the order, it must be that $f(a) \ \mathbf{li}_P \ f(x)$. But since l is maximal, and $f(a) \notin l \cap P$, there must be an element $y \in l : y \ \mathbf{co}_P \ f(a)$, in particular, as a subset of P, $\{y\}'' \subseteq a'$. Now out of Proposition 3.3.2 there is an atom b such that $b \subseteq \{y\}'' \cap l$. Then $b \in f^{-1}(l \cap P')$, and $b \leq_{L(P)} a'$, which contradicts that $\forall x \in f^{-1}(l \cap P') : a \ \mathbf{li}_P \ x$. Hence $f^{-1}(l \cap P')$ is a line of $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$, and π maps lines to lines.

To see that π is surjective, note that f preserves the order, so for every line $l_{\mathcal{A}_P}$ in $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$, $f(l_{\mathcal{A}_P})$ is a li-set of P. Any line l_P that contains $f(l_{\mathcal{A}_P})$, will provide $\pi(l_P) = l_{\mathcal{A}_P}$, and so π is surjective.

Finally to prove it is injective, consider two distinct lines $l_1, l_2 \in \mathcal{L}$. Then Proposition 3.3.3, provides two atoms $a_1, a_2 \in \mathcal{A}_P : a_1 \cap l_1 \neq \emptyset$ and $a_1 \cap l_2 = \emptyset$ and $a_2 \cap l_1 = \emptyset$ and $a_2 \cap l_2 \neq \emptyset$, and $a_1 \perp_{L(P)} a_2$. Hence, as elements of $\mathcal{A}_P$, $a_1 \ \mathbf{co}_{\mathcal{A}_P} \ a_2$. Then Proposition 3.2.6, $a_1 \subseteq l_1$, and $a_2 \subseteq l_2$, and so $f(a_1) \in l_1 \cap P'$, and $f(a_2) \in l_2 \cap P'$. Then $a_1 = f^{-1}(f(a_1)) \subseteq f^{-1}(l_1 \cap P') = \pi(l_1)$, and $a_2 = f^{-1}(f(a_2)) \subseteq f^{-1}(l_2 \cap P') = \pi(l_2)$. From $a_1 \ \mathbf{co}_{\mathcal{A}_P} \ a_2$ it follows that $a_1 \notin \pi(l_2)$, and $a_2 \notin \pi(l_1)$, which concludes the proof. □

This last result shows that, according to the provided definition, the reduced poset is in fact an abstraction.

At this point, it can be stated that $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ is the smallest poset embeddable into $(P, \leq_P)$ such that $L(P) \simeq L(\mathcal{A}_P)$, provided $L(P)$ is atomistic.

This notion of minimality is formalised as follows: Let $(P', \leq_{P'})$ be an arbitrary poset. If $(P', \leq_{P'})$ can be embedded into $(P, \leq_P)$, and $L(P) \simeq L(P')$, then $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ can be embedded into $(P', \leq_{P'})$. The following result can be found in [1].

Theorem 3.3.1. *Let $(P, \leq_P)$ and $(P', \leq_{P'})$ be two posets such that $L(P)$ and $L(P')$ are atomistic and isomorphic, and there exists an order embedding $g : (P', \leq_{P'}) \hookrightarrow (P, \leq_P)$. Let $\mathcal{A}_P = \mathcal{A}(P, \leq_P)$, then $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ can be embedded into $(P', \leq_{P'})$.*

Proof. Since both $L(P)$ and $L(P')$ are atomistic, from Proposition 3.2.12 it holds that $L(\mathcal{A}_P) \simeq L(P) \simeq L(P') \simeq L(\mathcal{A}_{P'}) = L(\mathcal{A}(P', \leq'))$. Now out of Propositions 3.2.8 and 3.2.9, and since g is an order embedding, there must be an isomorphism $\phi : \mathcal{A}(P, \leq) \to \mathcal{A}(P', \leq')$. Consider a choice function $f : \mathcal{A}(P', \leq') \to (P', \leq')$ as defined in (3.1). Then clearly $\phi \circ f : \mathcal{A}(P, \leq) \to (P', \leq')$ is an order embedding. □

As a consequence of the last two results, it can be said that whenever a poset is embeddable in another, and they have isomorphic closed set lattices, then the first one is an abstraction of the other. Indeed, the last theorem shows that they must have, up to isomorphism, the same reduced poset, and both their sets of lines are in bijection with the set of lines of the reduced one. Then their respective sets of lines must be in bijection as well.

3.3.2 Dedekind-McNeil Completion

The results of this section can be found in [1].

A poset can be reduced to a minimal form that preserves the lattice obtained from the closure induced by $\mathbf{co}_P$. This minimality has some implications in terms of local structure, which shall be exploited to offer an interpretation of the presented reduction.

Combinatorialness, and N-density are not sufficient for a partial order to be a causal net. Under these conditions one could still be unable to find a suitable bipartition of the elements.

However, the atoms of a poset, as defined in this work, structurally behave like conditions in the following sense. Given a combinatorial and N-dense poset $(P, \leq_P)$, the poset $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ resulting from reduction can be completed with events, so as to obtain a causal net.

First note that, whenever $\leq_P$ is combinatorial, so must be $\leq_{\mathcal{A}_P}$. On the other hand, reduction also preserves N-density.

Proposition 3.3.5. *Let $(P, \leq_P)$ be combinatorial, and N-dense. Then the poset $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ is N-dense.*

Proof. The proof is based on a result of [10] stating that if $(P, \leq_P)$ is N-dense then $L(P)$ is orthomodular, and therefore atomistic [39]. Then by Propositions 3.2.12 and 3.2.14, $L(\mathcal{A}_P) \simeq L(P)$ so that $L(\mathcal{A}_P)$ must also be orthomodular. This in turn implies, as shown in [12], that if $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ is combinatorial, then it must be N-dense. □

It will henceforth be assumed that both $(P, \leq_p)$, and $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ are combinatorial and N-dense.

In order to obtain the elements that will be interpretable as events, a Dedekind-MacNeille completion is performed on
$(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ (see for example [27, Ch7§36]). Some notation is required:

Definition 3.3.2 (Dedekind Mac-Neille Completion)**.** *Let $(P, \leq_P)$ be a poset, consider, for each $S \subseteq P$:*
The up-set of S: $\uparrow_\cap S := \{x \in P \mid \forall s \in S : s \leq_P x\}$
The down-set of S: $\downarrow_\cap S := \{x \in P \mid \forall s \in S : x \leq_P s\}$
Then $(\downarrow_\cap, \uparrow_\cap)$ forms a Galois connection.
The Dedekind-MacNeille completion *of $(P, \leq_P)$ is $\mathbf{DM}(P) := \{S \subseteq P \mid \downarrow_\cap(\uparrow_\cap S) = S\}$ with the order induced by inclusion.*

The following statements are known results, the reader is referred to [27, Ch7§36-44] for the full proofs. First note that $(\uparrow_\cap, \downarrow_\cap)$ is a Galois connection, hence $\downarrow_\cap(\uparrow_\cap \cdot)$ is a closure operator. $\mathbf{DM}(P)$ is a complete lattice, thus justifying the name. It contains the intersection of any of its elements. The empty set and P are trivially in $\mathbf{DM}(P)$, and it is common practice not to include them in $\mathbf{DM}(P)$, as it will be the case in this work. On the other hand, $\downarrow_\cap(\uparrow_\cap \cdot)$ constitutes a closure operator, so that $\forall S \subseteq P : S \subseteq \downarrow_\cap(\uparrow_\cap S)$. On top of that, it holds that $\forall x \in P : \downarrow_\cap(\uparrow_\cap(\downarrow_\cap\{x\})) = \downarrow_\cap\{x\}$ hence $\forall x \in P : \exists! \downarrow_\cap\{x\} \in \mathbf{DM}(P)$. This way, $(P, \leq)$ can be embedded into $(\mathbf{DM}(P), \subseteq)$ so that order is both preserved and reflected. Naturally, $\mathbf{co}_{DM(P)} := \{(s_1, s_2) \in \mathbf{DM}(P) \mid s_1 \not\subseteq s_2 \text{ and } s_2 \not\subseteq s_1\}$.

In what follows, $\leq_{DM(P)}$ will denote the order induced by inclusion on $\mathbf{DM}(P)$. Furthermore, for any combinatorial poset $(P, \leq_P)$, $\prec_P$ will denote the immediate successor relation, as in Definition 3.1.4 (see p.57). In general, $x \prec_P y$ iff $x \leq_P y$ and $x \neq y$ and $\forall z \in P : x \leq_P z \leq_P y \rightarrow (z = x \text{ or } z = y)$. When P is combinatorial, then $\leq_P$ is the reflexive and transitive closure of $\prec_P$.

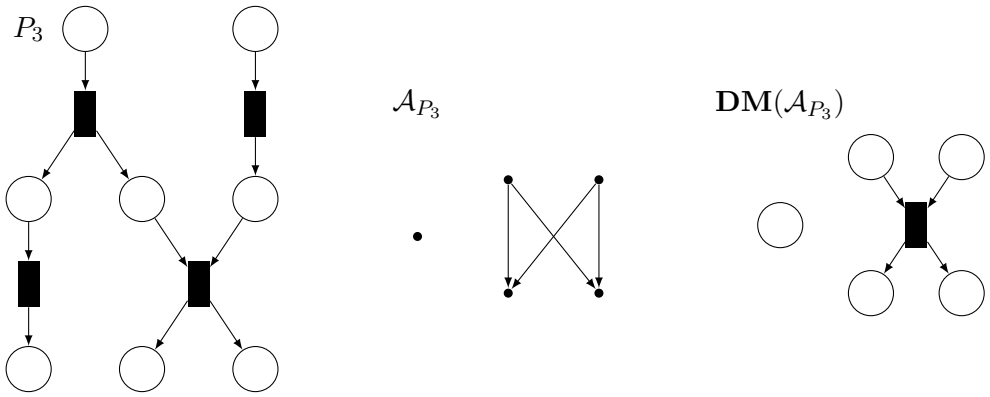

Figure 3.8: From left to right: poset P_3 of previous examples seen as a causal net, its atomic reduction, and its completion.

It is worth noting that, in the non-trivial case, $(\mathcal{A}_P, \leq_{\mathcal{A}_P})$ has no maximal, (nor minimal) element. Indeed, if there is an $x \in \mathcal{A}_P$ such that $\forall a \in \mathcal{A}_P$: $x \leq_{\mathcal{A}_P} a$ ($a \leq_{\mathcal{A}_P} x$) then clearly $x' = \emptyset$, so $x = \mathcal{A}_P$, and $(\mathcal{A}_P, \leq_{\mathcal{A}_P}) = (\{\mathcal{A}_P\}, \emptyset)$.

In the following, it will be shown that $\mathbf{DM}(\mathcal{A}_P)$ is a causal net. To this aim, some notation will ease the reading. Let:

1. $B_N = \{\downarrow_\cap \{a\} \mid a \in \mathcal{A}_P\}$ be the set of principal ideals, those elements of $\mathbf{DM}(\mathcal{A}_P)$ that can be identified with the ones of $\mathcal{A}_P$;

2. $E_N = \mathbf{DM}(\mathcal{A}_P) \setminus \{\downarrow_\cap \{a\} \mid a \in \mathcal{A}_P\} = \mathbf{DM}(\mathcal{A}_P) \setminus B_N$ be the elements introduced by the completion; and

3. $N = (B_N, E_N, \prec_{DM(\mathcal{A}_P)})$

The following two propositions prove, on one hand that the order on $\mathbf{DM}(\mathcal{A}_P)$ is nowhere dense (i.e. $\leq_{DM(\mathcal{A}_P)} = (\prec_{DM(\mathcal{A}_P)})^\star$); and on the other hand, that it is bipartite, hence $(\mathbf{DM}(\mathcal{A}_P), \prec_{DM(\mathcal{A}_P)}) \simeq N = (B_N, E_N, \prec_{DM(\mathcal{A}_P)})$ is an occurrence net.

Proposition 3.3.6. *Let* $a_1, a_2 \in \mathcal{A}_P : a_1 \prec_{\mathcal{A}_P} a_2$. *Then* $\exists! s \in \mathbf{DM}(\mathcal{A}_P) : \downarrow_\cap \{a_1\} \subsetneq s \subsetneq \downarrow_\cap \{a_2\}$.

Proof. By Proposition 3.2.8, it holds that $\exists a_3, a_4 \in \mathcal{A}_P$ such that $a_3 \ \mathbf{co}_{\mathcal{A}_P} \ a_1$, $a_3 \not\mathbf{co}_{\mathcal{A}_P} \ a_2$, $a_4 \ \mathbf{co}_{\mathcal{A}_P} \ a_2$, and $a_4 \not\mathbf{co}_{\mathcal{A}_P} \ a_1$. Since $a_1 \leq_{\mathcal{A}_P} a_2$, it must be $a_3 \leq_{\mathcal{A}_P} a_2$, and $a_1 \leq_{\mathcal{A}_P} a_4$. $a_3 \ \mathbf{co}_{\mathcal{A}_P} \ a_4$ contradicts N-density of $\mathcal{A}_P$, and

$a_4 \leq_{\mathcal{A}_P} a_3$ would imply $a_4 \leq_{\mathcal{A}_P} a_2$, so it must be that $a_3 \leq_{\mathcal{A}_P} a_4$. Then $\{a_2, a_4\} \subseteq \uparrow_\cap \{a_1, a_3\}$, and since a_4 $\mathbf{co}_{\mathcal{A}_P}$ a_2, $\{a_2, a_4\} \cap \downarrow_\cap (\uparrow_\cap \{a_1, a_3\}) = \emptyset$. Clearly neither $\downarrow_\cap \{a_1\} \subseteq \downarrow_\cap \{a_3\}$ nor $\downarrow_\cap \{a_3\} \subseteq \downarrow_\cap \{a_1\}$, so $s = \downarrow_\cap (\uparrow_\cap \{a_1, a_3\})$ satisfies $\downarrow_\cap \{a_3\}, \downarrow_\cap \{a_1\} \subsetneq s \subsetneq \downarrow_\cap \{a_2\}, \downarrow_\cap \{a_4\}$.

Now let $s_1, s_2 \in \mathbf{DM}(\mathcal{A}_P) : \downarrow_\cap \{a_1\} \subsetneq s_1 \subseteq s_2 \subsetneq \downarrow_\cap \{a_2\}$, and suppose $\exists a_5 \in s_2 \setminus s_1 \subseteq \downarrow_\cap \{a_2\}$. Clearly, $a_5 \notin \downarrow_\cap \{a_1\}$, and $a_1 \leq_{\mathcal{A}_P} a_5$ would contradict $a_1 \prec_{\mathcal{A}_P} a_2$. Consider any $a_6 \in \uparrow_\cap s_1 \setminus \uparrow_\cap \{a_5\} \subseteq \uparrow_\cap \{a_1\}$. $a_6 \leq_{\mathcal{A}_P} a_5$ contradicts a_1 $\mathbf{co}_{\mathcal{A}_P}$ a_5. Then again, $a_1 \prec_{\mathcal{A}_P} a_2$ implies that $a_6 \not\leq_{\mathcal{A}_P} a_2$, and $a_2 \leq_{\mathcal{A}_P} a_6$ would mean that a_5 $\not\mathbf{co}_{\mathcal{A}_P}$ a_6. But then a_1, a_2, a_5, a_6 form a configuration which contradicts N-density of $\mathcal{A}_P$. Then $\uparrow_\cap s_1 \setminus \uparrow_\cap \{a_5\} = \emptyset$. So $\uparrow_\cap s_1 \subseteq \uparrow_\cap \{a_5\}$ and then clearly $a_5 \in \downarrow_\cap \{a_5\} \subseteq \downarrow_\cap s_1 = s_1$ which is absurd. Hence, $s_1 = s_2$. □

Proposition 3.3.7. *Let $S \in E_N$, and $a_i, a_f \in \mathcal{A}_P$ be such that $\downarrow_\cap \{a_i\} \subseteq S \subseteq \downarrow_\cap \{a_f\}$. Then $\exists a, a' \in \mathcal{A}_P$ such that $a \prec_{\mathcal{A}_P} a'$, and $\downarrow_\cap \{a_i\} \subseteq \downarrow_\cap \{a\} \subseteq S \subseteq \downarrow_\cap \{a'\} \subseteq \downarrow_\cap \{a_f\}$*

Proof. Since $\mathcal{A}_P$ is combinatorial, and $a_i \leq_{\mathcal{A}_P} a_f$, there must exist a finite sequence $\{a_j\}_{j=1}^n$ such that $a_i = a_1, a_f = a_n$, and $\forall 1 \leq k < n : a_k \prec_{\mathcal{A}_P} a_{k+1}$. Note that $a_f \in S$ would mean that $S = \downarrow_\cap \{a_f\} \in \{\downarrow_\cap \{a\} \mid a \in \mathcal{A}_P\} = B_N$ which is absurd. On the other hand, $a_{k_0} \in S$ implies that $\forall k \leq k_0 : a_k \in S$. So $\{a_j\}_{j=1}^n \cap S$ has a maximal element a_{k_m}. Now suppose $\exists a_s \in \uparrow_\cap S \setminus \uparrow_\cap \{a_{m+1}\} \subseteq \uparrow_\cap \{a_m\}$. Then it must be a_s $\mathbf{co}_{\mathcal{A}_P}$ a_{m+1}. Now $\forall a'_s \in \downarrow_\cap \{a_{m+1}\}$, either $a'_s \leq_{\mathcal{A}_P} a_m$ or a'_s $\mathbf{co}_{\mathcal{A}_P}$ a_m. In the latter case, $a_s \not\leq_{\mathcal{A}_P} a'_s$, and a'_s $\mathbf{co}_{\mathcal{A}_P}$ a_m would contradict N-density of $\mathcal{A}_P$. As a consequence, $\forall a'_s \in \downarrow_\cap \{a_{m+1}\} : a'_s \leq_{\mathcal{A}_P} a_s$. Therefore $\downarrow_\cap \uparrow_\cap S = S \subseteq \downarrow_\cap \{a_{m+1}\}$, and so $\downarrow_\cap \{a_m\} \subseteq S \subseteq \downarrow_\cap \{a_{m+1}\}$. □

These results imply that $\mathbf{DM}(\mathcal{A}_P)$ is combinatorial, and that $\forall a_1, a_2 \in \mathcal{A}_P : a_1 \not\prec_{DM(\mathcal{A}_P)} a_2$. Furthermore, suppose $s_1, s_2 \in E_N : s_1 \prec_{DM(\mathcal{A}_P)} s_2$. Then $\exists a_1 \in s_2 \setminus s_1 : s_1 \subsetneq \downarrow_\cap \{a_1\} \subsetneq s_2$ which is absurd. And so $N_{\mathcal{A}_P} = (B_N, E_N, \prec_{DM(\mathcal{A}_P)})$ is a Petri net, which is certainly acyclic. Furthermore, for $s \in E_N$ either $s = \emptyset$, $s = \mathcal{A}_P$, or $\exists a_1, a_2 \in \mathcal{A}_P$ such that $\downarrow_\cap \{a_1\} \prec_{DM(\mathcal{A}_P)} s \prec_{DM(\mathcal{A}_P)} \downarrow_\cap \{a_2\}$.

It is finally proven that N is a causal net. This is achieved by showing that it is conflict-free, in other words, that all forks and joins happen at elements of E_N. Proposition 3.3.8 proves that no forks can happen at B_N, whereas Proposition 3.3.9 shows the respective result for joins.

Proposition 3.3.8. *Let $a \in \mathcal{A}_P$, and $s_1, s_2 \in E_N$ such that $\downarrow_\cap \{a\} \prec_{DM(\mathcal{A}_P)} s_1$, and $\downarrow_\cap \{a\} \prec_{DM(\mathcal{A}_P)} s_2$. Then $s_1 = s_2$*

Proof. Clearly $s_2 \subseteq s_1$ would contradict $a \prec_{DM(\mathcal{A}_P)} s_1$. In particular $s_1 \neq \mathcal{A}_P$ and $s_2 \neq \emptyset$ and by analogy $s_2 \neq \mathcal{A}_P$ and $s_1 \neq \emptyset$. So there must be $a_1, a_2 \in \mathcal{A}_P : s_1 \prec_{DM(\mathcal{A}_P)} \downarrow_\cap \{a_1\}$ and $s_2 \prec_{DM(\mathcal{A}_P)} \downarrow_\cap \{a_2\}$. Then $a \prec_{\mathcal{A}_P} a_1$ and $a \prec_{\mathcal{A}_P} a_2$. Obviously $a_1 \ \mathbf{co}_{\mathcal{A}_P} \ a_2$.

Suppose $s_1 \neq s_2$ and, without loss of generality, let $a_3 \in s_1 \setminus s_2 \subseteq \downarrow_\cap \{a_1\}$. Then $a \leq_{\mathcal{A}_P} a_3$ implies that $a \not\prec_{\mathcal{A}_P} a_1$, and $\downarrow_\cap \{a\} \subseteq s_2$ implies that $a_3 \not\leq_{\mathcal{A}_P} a$. So $a \ \mathbf{co}_{\leq_{\mathcal{A}_P}} \ a_3$. Now $a_2 \not\leq_{\mathcal{A}_P} a_3$ since $a_1 \ \mathbf{co}_{\mathcal{A}_P} \ a_2$. Now suppose $a_3 \leq_{\mathcal{A}_P} a_2$. Then $a_2 \in \uparrow_\cap (\{a_3\} \cup s_2)$, and so $\downarrow_\cap \{a\} \subsetneq s_2 \subsetneq \downarrow_\cap (\uparrow_\cap (\{a_3\} \cup s_2)) \subsetneq \downarrow_\cap \{a_2\}$, thus contradicting Proposition 3.3.6. Therefore, the only possibility is that $a_3 \ \mathbf{co}_{\mathcal{A}_P} \ a_2$, and so a, a_1, a_2, a_3 form a configuration which contradicts N-density of $\mathcal{A}_P$. So it must be $s_1 = s_2$ □

Proposition 3.3.9. *Let $a \in \mathcal{A}_P$, and $s_1, s_2 \in E_N$ such that $s_1 \prec_{DM(\mathcal{A}_P)} \downarrow_\cap \{a\}$, and $s_2 \prec_{DM(\mathcal{A}_P)} \downarrow_\cap \{a\}$. Then $s_1 = s_2$*

Proof. Just like in the previous proof, $\emptyset \neq s_1 \neq \mathcal{A}_P$ and $\emptyset \neq s_1 \neq \mathcal{A}_P$, so there must be $a_1, a_2 \in \mathcal{A}_P : \downarrow_\cap \{a_1\} \prec_{DM(\mathcal{A}_P)} s_1$ and $\downarrow_\cap \{a_2\} \prec_{DM(\mathcal{A}_P)} s_2$, such that $a_1 \prec_{\mathcal{A}_P} a$ and $a_2 \prec_{\mathcal{A}_P} a$. Note that $a_1 \ \mathbf{co}_{\mathcal{A}_P} \ a_2$. Let $a_3 \in \uparrow_\cap \{a_1\}$. Then either $a \leq_{\mathcal{A}_P} a_3$ or $a \ \mathbf{co}_{\mathcal{A}_P} \ a_3$. Such an element must exist, since $a_1, a \in \mathcal{A}_P$, and by Proposition 3.2.8, there must be an $a_3 \in \mathcal{A}_P$ such that $a_3 \ \mathbf{co}_{\mathcal{A}_P} \ a$, and $a_1 \ \mathbf{li}_{\mathcal{A}_P} \ a_3$. Then clearly $a_1 \leq_{\mathcal{A}_P} a_3$. So $a_3 \ \mathbf{co}_{\mathcal{A}_P} \ a_2$ would contradict N-density of $\mathcal{A}_P$. Since $a_3 \ \mathbf{co}_{\mathcal{A}_P} \ a$, it holds that $a_2 \leq_{\mathcal{A}_P} a_3$. This means $\uparrow_\cap \{a\} \subseteq \uparrow_\cap \{a_1\} \subseteq \uparrow_\cap \{a_2\}$. Analogously, $\uparrow_\cap \{a\} \subseteq \uparrow_\cap \{a_2\} \subseteq \uparrow_\cap \{a_1\}$. Therefore, $\downarrow_\cap \{a_1\} \cup \downarrow_\cap \{a_2\} \subseteq \downarrow_\cap \uparrow_\cap \{a_1, a_2\} \subseteq \downarrow_\cap \{a\}$. Finally, by applying Proposition 3.3.6 it must be that $s_1 = \downarrow_\cap \uparrow_\cap \{a_1, a_2\} = s_2$. □

So $N_{\mathcal{A}_P} = (B_N, E_N, \prec_{DM(\mathcal{A}_P)})$ is a causal net.
These last results furthermore imply that $L(\mathbf{DM}(\mathcal{A}_P)) \simeq L(\mathcal{A}_P)$.

Proposition 3.3.10. $L(\mathbf{DM}(\mathcal{A}_P)) \simeq L(\mathcal{A}_P)$

Proof. Let $s \in E_N$,and $a \in B_N$ such that $a \prec_{DM(\mathcal{A}_P)} s$. Suppose $\exists s' \ \mathbf{co}_{DM(\mathcal{A}_P)} s$, then clearly $s' \not\leq_{DM(\mathcal{A}_P)} a$, and $a \leq_{DM(\mathcal{A}_P)} s'$ would contradict Proposition 3.3.8. So $s' \ \mathbf{co}_{DM(\mathcal{A}_P)} \ a$, and then it must hold that $\forall s \in E_N : \exists a \in B_N : s' \subseteq a'$, and so $\{a\}'' \subseteq \{s\}''$. Therefore, by Proposition 3.2.4, and since $\forall a_1, a_2 \in \mathcal{A}_P : \downarrow_\cap \{a_1\} \leq_{DM(\mathcal{A}_P)} \downarrow_\cap \{a_2\} \Leftrightarrow a_1 \leq_{\mathcal{A}_P} a_2$, $\mathcal{A}_{\mathbf{DM}(\mathcal{A}_P)} \simeq \mathcal{A}_P$. Note that by Proposition 3.2.2, the $\perp$ relation must be preserved. But this means that, whenever $\mathcal{A}_P$ is N-dense and therefore $L(\mathcal{A}_P)$ is atomistic, then by Proposition 3.2.12, $L(\mathbf{DM}(\mathcal{A}_P)) \simeq L(\mathcal{A}_P)$. □

3.3.3 Maximal Distribution, and Communication Protocol

When seen as subsets of P, the atoms of its lattice $L(P)$ are total orders. This means that the elements of an atom are totally ordered, and must occur in that particular order. As a consequence, when distributing the process, the elements of an atom require a consistent measurement of time, which can be achieved by allocating them to one single component of the system. Since they are totally ordered, the elements of an atom have a single minimal element, and a singe maximal element. These elements in turn, are the only ones in covering relation with the rest of the process. Hence, when interpreting an atom as a subprocess, it can only synchronise with the rest of the process through its first, and last element. In the reduced poset, these totally ordered sequences are collapsed to one single element, which gathers the interactions with the rest of the system of their first, and last elements. Thus, the reduced poset represents all the interactions between the fully sequential parts of the system, and this is the reason why the two lattices are isomorphic. Furthermore, since the two lattices are isomorphic, the poset composed of atomic closed sets has the same set of lines as the poset it abstracts.

The fact that the lattice of closed sets gathers the information regarding the interactions among subprocesses was already justified in [10], by showing that the closure operator here presented is equivalent to another closure operator, based on these interactions. This closure operator is defined on causal nets. It will now be briefly presented.

Definition 3.3.3 (Causally Closed Set)**.** *Let $N' = (B', E', \mathcal{F}')$ be an interval-finite causal net of finite degree. A subset $C \subset B' \cup E'$ is* causally closed*, when it satisfies the following axioms.*

1. $\forall e \in E' : {}^\bullet e \subseteq C \to e \in C$
2. $\forall e \in E' : e^\bullet \subseteq C \to e \in C$
3. $\forall e \in E' : e \in C \to {}^\bullet e \cup e^\bullet \subseteq C$
4. $\forall x, y \in C : x \ \mathbf{li}_{N'} \ y \to [x, y] \subseteq C$

It was proven in [10] that if the net is K-dense, then this closure operator coincides with the one based on concurrency studied in this work.

Intuitively, a causally closed set can be obtained by interpreting the causal net as a condition/event system. Suppose a set is composed of pairwise concurrent conditions. Its causal closure can be obtained as follows. Consider the partial marking composed of these conditions. Note that this marking may

not be a cut. Then the collection of reachable markings with the backward and forward firing rule provides all the conditions in the closure. The fired events are those included in the closure.

This construction admits the following interpretation. When observing a partial state of the process, as a subset of conditions. Its causal closure is the maximal subprocess that can be inferred from the observed information. In order to acquire knowledge about the process beyond this subprocess requires additional observations. In particular, in order to enlarge the obtained subprocess, one must add a condition to the initial partial state. This condition is concurrent to all the already known subprocess.

This supports the idea that the observation of a particular subprocess provides no information on its polar. Elements of the polar always provide additional information. This is expressed by the orthomodular law. In this sense $x \leq y \rightarrow y = x \vee (y \wedge x')$ is to be read as 'If a subprocess is larger than another, then all the observations required to retrieve the larger lie in the polar of the smaller'.

Additionally, it was shown in [10] that under the same hypothesis, every line crosses either a closed set, or its polar, but never both. This again, supports the view, that information never flows between a subprocess and its polar.

Atoms correspond to the subprocesses which can be reconstructed with a minimal observation. And in fact, they can be interpreted, in the reduced poset, as the observation in itself. In the reduced poset, observing a single element provides no information about the rest of the process. This is to be interpreted as the fact, that all the interactions in the process are depicted in the reduced partial order.

Chapter 4

System Distributability

In analysing how processes can be distributed, the causal relations between the occurrences of actions are exploited. However, the partial ordering of these plays a crucial role, and the technique presented in the last chapter do not transfer trivially to systems. Indeed, as representations of the computing devices that run processes, the causal dependencies between the actions are represented in a more complex fashion. The fact that system models depict the actions themselves rather than their occurrences, weakens the partial order relation binding them. The transitive closure of the direct dependencies would lead to a relation of reachability rather than a partial order. However, the closure operator defined on the last chapter relies on the lack of causal dependences on the process model. This is interpreted as a lack of information transfer between subprocesses, so that these can not observe the execution of remote actions, or for instance their effect on the corresponding local states.

An analogous strategy for studying how to distribute a system consists in relying on the notion of observable property, thus taking advantage of the principle of locality. Indeed, elementary models allow for identification of these properties with the local states of a system.

The elementary framework provides a richly developed theory for the study of the relation of these local states, and their role in composing global states and behaviour of the systems. In Chapter 2, the standard derivation of the behaviour of a net system as a labelled transition system was presented. This transition system is called the case graph of the net. By analysing this case graph, one can deduce which subsets of states are the extension of some observable property, and hence, of some local state. Such local states might not be explicitly represented in the net system. However, they would be redundant, adding them to the system wouldn't affect its behaviour. In this way,

Petri Net Synthesis provides means to perform a completion of the net system, in which all implicit places are represented. The original net system, and its completion have then the same behaviour. The sequential components of the system can be isolated with structural analysis on the completion. This is commonly called a state machine decomposition. An elementary system need not be state machine decomposable, however considering its completion ensures that all the required local sates are explicit in the model, in order to achieve this decomposition. Furthermore, the complete collection of local states can be endowed with a rich structure. Such a structure admits a logical interpretation when local states are considered as the observable properties of the system. Indeed, when the extensions of such properties are ordered by set inclusion, by interpreting set complement as logical negation, the resulting partial order is orthomodular. This algebraic structure gathers the information on how the system can be distributed, filtering out behavioural aspects such as the relation between actions belonging to the same component. This orthomodular partial order can be understood as a specification of the system architecture available. Indeed, the specification of the properties that need to be observable imposes restrictions on how actions can be distributed. For an observer to determine if a given property holds, the set of global states that compose its extension must be expressible in terms of the local states of one single sequential component.

The orthomodular partial order of the observable properties of a system is rich, faithfully representing the relations between local and global states. It is also regular, it encodes the fact that collections of local states belong to the same component. Under these conditions, an orthomodular structure carries all the information on how local states can be distributed spatially, and how global state compose from them. Not only can it express the set of available global states of a given distribution of components, but it also represents the interactions available for communication between localities. A set of available actions, or events, can be derived from this specification, together with the information regarding which sequential components each of them is involved in. Channels of communication are distinguished from strictly local states, and local actions can be distinguished from communication attempts. The state of the art provides a formalisation of all available behaviour, given an orthomodular poset specification. An elementary transition system can be constructed which can run any possible process on the specified architecture, it is thus called saturated. Indeed, it represents all possible global states and actions one could feature, in order to make sure that the desired properties can be observed. The saturated transition system contains, as subsystems, all the elementary transition systems for which the specified properties are

indeed observable.

The saturated transition system is elementary, as such it is known to have an orthomodular poset as structure for its observable properties. This poset, however, need not coincide with the specified one, it is only known to contain it as substructure. It is thus ensured that every specified property is indeed observable in it, but it may present more synchronisations than allowed by the specification. The fact that all specified properties are indeed observable is an interesting feature, but one would further like to make sure that local states can be distributed as specified by this structure. It has been conjectured that this is the case whenever the orthomodular partial order arises as the set of regions of an elementary transition system. This matter will be addressed in Chapter 5. In the present chapter, however, a saturated transition system is understood to be related to two different orthomular partial orders. One corresponds to the specification of the provided architecture, whereas the other corresponds to the ordered set of its regions. The first one embeds into the other, meaning that sequential components encoded in the first one will be present in the second. These, however, could have been merged, or synchronised. On the other hand, all concurrency depicted by the second, is certain to arise from the first. In this way, the two orthomodular posets behave like boundaries for the behaviour of the saturated transition system. All synchronisations specified by the first one will be present in the transition system, but its concurrent behaviour will be specified by the second.

Hence, in this chapter the way orthomodular partial orders express concurrency is studied. Events are the key feature in this sense, capturing the focus of the last section. It is shown that some events are sufficient to express the belonging of local states to different sequential components. Thus, one can obtain alternative transition systems from a given orthomodular poset specification, such that the properties observable on them coincide with those observable on the saturated version. Intuitively, the behaviour of these systems are bounded by the same orthomodular posets as the saturated transition system. The considered events capture all the information regarding concurrency of the system, and are so sufficient to distribute it efficiently. These events inherit features from the orthomodular poset they are built upon, and can hence be endowed with a structure that allows to study their interactions, and their involvement in the distributability of the systems they represent the actions of.

4.1 Observable Properties of Elementary Systems

The aim of this section is to formalise the notion of observable property of a system. When modelled on labelled transition systems, properties are identified with their extensions, the states of the system at which they hold. A property is considered observable when its truth value can be determined from a single location. This notion is formalised using the theory of Petri net synthesis.

4.1.1 Net Synthesis

In the seminal series of papers by Ehrenfeucht and Rozenberg regarding 2-structures [30, 31], the authors characterised the class of labelled transition systems which are the case graph of some elementary net system. They used a slightly different, although equivalent, formalism than the one presented in this work.

2-structures are directed graphs ([4]) equipped with an equivalence relation defined on the set of arcs. In relation to labelled transition systems, their vertices are to be interpreted as states of the system, and the arcs as the transitions. The equivalence relation then encodes the labelling of the transitions. Two arcs are considered equivalent when they carry the same label. This correspondence transfers their characterisation results to the frame of elementary systems.

The problem of characterising elementary transition system goes together with the problem of, given a transition system, constructing the net system that will have a case graph isomorphic to it. The solution to this problem was already introduced in Section 2.2.2. In the present section, the construction is presented in more detail, with the scope of motivating the interpretation of regions as the extensions of the observable properties of the system.

Definition 4.1.1 (Saturated Net System)**.** *Let $A = (Q, E, T, q_0)$ be an initialised labelled transition system, as in Definition 2.2.7. Let $\mathcal{R}(A)$ be its set of regions. Define $\mathcal{F} := \{(r, e) \in \mathcal{R}(A) \times E \mid e \in r^\bullet\} \cup \{(e, r) \in E \times \mathcal{R}(A) \mid e \in {}^\bullet r\}$, and $m_0 := \{r \in \mathcal{R}(A) \mid q_0 \in r\}$.*

The saturated net system *associated with A is the Petri net system $N = (\mathcal{R}(A), E, \mathcal{F}, m_0)$*

The following theorem gathers the results of elementary net synthesis.

Theorem 4.1.1. *If A is an elementary transition system, then N, as in Definition 4.1.1, is an elementary net system, and $\boldsymbol{CG}(N) \simeq A$.*

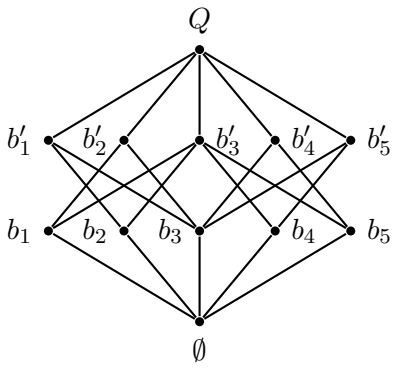

Figure 4.1: Hasse diagram representation of the regional poset corresponding to the transition system of Figures 2.7 (p.36) and 2.10 (p.47).

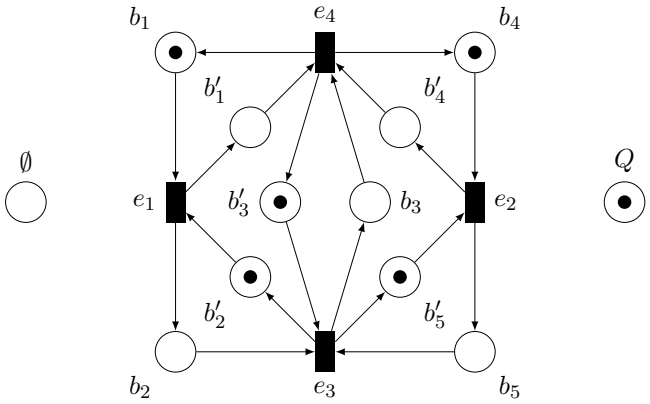

Figure 4.2: Saturated net system obtained by synthesis from the transition system of Figure 2.7 (p.36). The conditions of the net correspond to the regions of Figure 4.1.

Example 4.1.1. *Let $A = (Q, E, T, q_1)$ be the elementary transition of Figure 2.7 system such that $Q = \{q_1, \ldots, q_5\}$, and $E = \{e_1, \ldots, e_4\}$, with*

$$\begin{aligned} T = \{&(q_1, e_1, q_3), (q_2, e_1, q_4),\\ &(q_1, e_2, q_2), (q_3, e_2, q_4),\\ &(q_4, e_3, q_5), (q_5, e_4, q_1)\} \end{aligned}$$

The set of regions is composed of $\emptyset$, $b_1 = \{q_1, q_2\}$, $b_2 = \{q_3, q_4\}$, $b_3 = \{q_5\}$, $b_4 = \{q_2, q_4\}$, $b_5 = \{q_1, q_3\}$, and their set complements. When these sets are ordered by inclusion, one obtains the logic depicted in Figure 4.1. The incidence with respect to the events of E is as follows.

$$\begin{aligned} {}^\bullet e_1 = \{b_1, b_2'\}, \quad e_1^\bullet = \{b_2, b_1'\}\\ {}^\bullet e_2 = \{b_4, b_5'\}, \quad e_2^\bullet = \{b_5, b_4'\}\\ {}^\bullet e_3 = \{b_2, b_5, b_3'\}, \quad e_3^\bullet = \{b_3, b_2', b_5'\}\\ {}^\bullet e_4 = \{b_3, b_1', b_4'\}, \quad e_4^\bullet = \{b_1, b_4, b3'\} \end{aligned}$$

In this way, one can synthesis the saturated net system of Figure 4.2. The initial marking is given by all the regions which contain q_1.

Note that, in the setting of Definition 4.1.1, a marking m_q contains a condition r, whenever the corresponding state q is contained in its extension.

By dropping the initial state, one can consider the condition/event case.

Theorem 4.1.2. *Let $A = (Q, E, T)$ be a labelled transition system, Let $\mathcal{R}(A)$ be its set of regions. Define $\mathcal{F} := \{(r, e) \in \mathcal{R}(A) \times E \mid e \in r^\bullet\} \cup \{(e, r) \in E \times \mathcal{R}(A) \mid e \in {}^\bullet r\}$. To every $q \in Q$, associate the set of regions that contain it $m_q := \{r \in \mathcal{R}(A) \mid q \in r\}$, and let $M := \{m_q \subseteq \mathcal{R}(A) \mid q \in Q\}$. If A is a connected condition/event transition system, then $N = (\mathcal{R}(A), E, \mathcal{F}, M)$ is a condition/event net system, and and $\boldsymbol{CG}(N) \simeq A$.*

Given a condition/event, or elementary transition system, its saturated net system presents all conditions consistent with the specified behaviour. In the saturated net system, the set of conditions is identified with the orthomdular poset of regions, providing interesting interpretations to their interactions.

Proposition 4.1.1. *Let $A = (Q, E, T)$ be a condition/event transition system, and $N = (\mathcal{R}(A), E, \mathcal{F}, M)$ its saturated net system, as in Definition 4.1.1. Consider a collection R of pairwise disjoint regions. Then their union $\bigvee R$ is a region, and*

$${}^\bullet(\bigvee R) \subseteq (\bigcup_{r \in R} {}^\bullet r) \text{ and } (\bigvee R)^\bullet \subseteq (\bigcup_{r \in R} r^\bullet)$$

$\bigvee R$ *is called an* abstraction *of* R.

Proof. From the definition of the flow relation in Definition 4.1.1, an element of ${}^\bullet(\bigvee R)$ must be an event e such that $\forall (q_1, e, q_2) \in T : q_1 \notin (\bigvee R)$ and $q_2 \in (\bigvee R)$. Then, since $(\bigvee R) = \bigcup_{r \in R} r$, there must be a region $r \in R$ such that $q_2 \in r$. Clearly $q_1 \notin r$, so it must be $e \in {}^\bullet r$. The result is analogous for $(\bigvee R)^\bullet$. □

The abstraction of a collection of conditions is marked iff one of them is. In the saturated net system, any collection of mutually exclusive conditions has an abstraction.

This notion of abstraction can be defined in net systems in general, however an arbitrary net system does not need to present an abstraction for each mutually exclusive collection. The reason for the results of Theorems 4.1.1,and 4.1.2 can be explained in terms of separation axioms. Given an arbitrary initialised transition system, the separation axioms induce a set of problems that can be solved by sets of regions.

Definition 4.1.2 (Separation Problems, Separating Regions)**.**
Let $A = (Q, E, T, q_0)$ *be an initialised transition system. The elementary separation axioms of Definition 2.2.6 (see p. 37) define a set of problems.*

The state-state separation problems *are based on the first axiom. One problem is defined for every pair* $(q_1, q_2) \in Q \times Q$ *such that* $q_1 \neq q_2$*. A region* $r \in \mathcal{R}(A)$ *is said to solve the problem whenever* ($q_1 \in r$*, and* $q_2 \notin r$*), or* ($q_2 \in r$*, and* $q_1 \notin r$*).*

A state-event separation problem *is defined for each pair* $(q, e) \in Q \times E$ *such that* q *does not enable* e*. A region* $r \in \mathcal{R}(A)$ *is said to solve the problem when* $r \in {}^\bullet e$ *and* $q \notin r$*.*

If the regions in $\mathcal{R}(A)$ *solve all these separation problems, then the elementary separation axioms are verified. Hence, if every state is reachable from* q_0*, the transition system is elementary.*

When dropping the initial state, $A = (Q, E, T)$*, one can consider the state-state, and state-event separation problems, together with the set of problems corresponding to the third axiom in Definition 2.2.6.*

An event-state separation problem *is given for each pair* $(e, q) \in E \times Q$ *such that* q *does not backward-enable* e*. Then a region* $r \in \mathcal{R}(A)$ *is said to solve the problem when* $r \in e^\bullet$*, and* $q \notin r$*.*

If the set of regions $\mathcal{R}(A)$ *solves all these separation problems, then the transition system is condition/event.*

Furthermore, a collection of regions $R \in \mathcal{R}(A)$ is called separating*, or said to* separate *the transition system, when it gathers enough regions to solve all the separation problems.*

A collection of regions which solve all separation problems is sufficient to reproduce the behaviour of the system.

The following result can be found in [29]. It is here stated for condition/event systems, the result for the elementary case is analogous

Theorem 4.1.3. *Let $A = (Q, E, T)$ be a labelled transition system. Let $\mathcal{R}(A)$ be its set of regions, and let $B \subseteq \mathcal{R}(A)$ be a separating collection of regions.*

Then B exists iff A is a condition/event transition system.

To every $q \in Q$, associate the set of regions that contain it $m_q := \{r \in \mathcal{R}(A) \mid q \in r\}$, and let $M := \{m_q \subseteq \mathcal{R}(A) \mid q \in Q\}$.
Define $\mathcal{F} := \{(r, e) \in B \times E \mid e \in r^\bullet\} \cup \{(e, r) \in E \times B \mid e \in {}^\bullet r\}$. If A is a connected condition/event transition system, then $N = (B, E, \mathcal{F}, M)$ is a condition/event net system, and $\boldsymbol{CG}(N) \simeq A$.

With this result, Theorems 4.1.1 and 4.1.2 can be restated as "The collection of all regions of an elementary (condition-event) transition system separates it."

The idea developed in this section is that the saturated net system depicts all the local states which are consistent with the specified behaviour. Indeed, the set of regions contains the extension of all conditions which admit a consistent flow relation with the events. According to Petri's extensionality principle, there can be only one condition with the flow relation described by each region. Hence, any other condition would certainly affect the behaviour of the system.

Note that, when a collection of regions B separates a system A, then any collection B' such that $B \subseteq B'$ is also separating. Given a separating collection of regions B, consider the net system built from them, as in Theorem 4.1.3. Then adding the condition associated to any region which is not present in the collection will not alter the behaviour of the system. Such a condition is said to be *redundant with respect to B*.

The saturated net system presents one solution to the synthesis problem. This is sufficient to characterise the class of elementary transition, or condition/event systems. However, given a transition system A, other net systems than the saturated one might have a case graph isomorphic to A.

Nevertheless, it is worth noting that another canonical representative of this class can be constructed. The idea behind this alternative construction

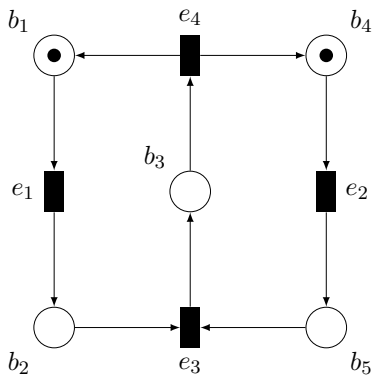

Figure 4.3: Net system obtained as in Example 4.1.1 when considering only the minimal regions b_1, b_2, b_3, b_4, b_5. The case graph of this system is represented in Figures 2.7 (p.36), and 2.10 (p.47).

is that a particular collection of regions is sufficient to satisfy the separation axioms.

Definition 4.1.3 (Minimal Region)**.** *Let A be a labelled transition system, and let $\mathcal{R}(A)$ be its set of regions. A region $r \in \mathcal{R}(A)$ is called* minimal *whenever it does not contain any non-trivial region as a proper subset,*

$$\forall r' \in \mathcal{R}(A) : (r' \subseteq r) \rightarrow (r = r')$$

The set of minimal regions of A will be denoted $\mathcal{R}_{\text{MIN}}(A)$.

The following result was proven in [5]. It is here presented for the condition/event case, but it transfers naturally to the elementary one.

Theorem 4.1.4 (Minimal Net System)**.** *Let $A = (Q, E, T)$ be a condition/event transition system, Let $\mathcal{R}(A)$ be its set of regions, and $\mathcal{R}_{\text{MIN}}(A)$ as in Definition 4.1.3, then $\mathcal{R}_{\text{MIN}}(A)$ separates A.*

Note that the term 'minimal net system' is motivated only by the fact that it is built from the minimal regions of the transition system. It has, in general, not been shown to be minimal with respect to a comparison relation on net systems. However, it supports the view that the saturated net system, and the minimal net system are two canonical solutions to the synthesis problem. The following result is a corollary of Theorem 4.1.4

Corollary 4.1.1. *Let $A = (Q, E, T)$ be a labelled transition system, Let $\mathcal{R}(A)$ be its set of regions, and $\mathcal{R}_{\text{MIN}}(A)$ as in Definition 4.1.3. Let B be any set of regions which contains the minimal ones. $\mathcal{R}_{\text{MIN}}(A) \subseteq B \subseteq \mathcal{R}(A)$. Define $\mathcal{F} := \{(r, e) \in B \times E \mid e \in r^\bullet\} \cup \{(e, r) \in E \times B \mid e \in {}^\bullet r\}$. If A is a connected condition/event transition system, then $N = (B, E, \mathcal{F}, Q)$ is a condition/event net system, and and $\boldsymbol{CG}(N) \simeq A$.*

4.1.2 Properties as Monitors

This section presents an application of the results of net synthesis, to support the proposed interpretation of observable properties of the system, in terms of region theory.

The setting will in this case assume that a Petri net system (either elementary or condition/event) is provided, and analyse the modifications that can be applied to it without altering its behaviour. In particular, conditions of a net system are interpreted as Boolean variables carrying the value of a set of propositions of the system, which depend on the state the system is at.

The following result is a corollary of those of the last section. Again, the result is provided for condition/event systems, but can be transferred naturally to the elementary case.

Corollary 4.1.2. *Let $N = (B, E, \mathcal{F}, M)$ be a condition/event net system, and $\boldsymbol{CG}(N)$ be its case graph. Consider $\mathcal{R}_B = \{\mathbf{ext}\, b \subseteq M \mid b \in B\} \subseteq \mathcal{R}(\boldsymbol{CG}(N))$, the collection of extensions of the conditions of N. Let B' be a set of regions which contains it, $\mathcal{R}_B \subseteq B' \subseteq \mathcal{R}(\boldsymbol{CG}(N))$. Define $\mathcal{F}' := \{(r, e) \in B' \times E \mid e \in r^\bullet\} \cup \{(e, r) \in E \times B' \mid e \in {}^\bullet r\}$, then $N' = (B', E, \mathcal{F}', M)$ is a condition/event system, and $\boldsymbol{CG}(N') \simeq \boldsymbol{CG}(N)$.*

The introductory example 2.3 (see p. 22) is an application of this result.

Consider a property regarding states of the system. Such a property could be expressed by propositions like "the system reached a deadlock", or "the system is able to provide service x". It is common practice to identify this kind of properties with the set of states at which they hold, as it is done, for example, in Kripke structures (see for example [24]). The set of states at which a property holds is called its *extension*. Note that through this identification, two properties are considered equivalent when they hold at exactly the same states. In the following, properties will be identified with their extensions. Among the models considered in this work, labelled transition systems depict the global states of the system, and so it is in these models that properties will

be seen as subsets of states. In principle, any subset of states of the system represents some property.

The question of whether a property is observable on the system will be tackled thanks to the notion of monitor. Intuitively, a monitor for a given property is the implementation of a Boolean variable which is true exactly at the states in which the property holds. Monitors will be formalised, in this work, as conditions of a net system. This notion of monitor is not a standard one. The choice of the term represents the point of view of the author.

Definition 4.1.4 (Monitor of a Property)**.** *Let $N = (B, E, \mathcal{F}, M)$ be a condition/event net system, and let $S \subseteq M$ be a property of the system. Then $b_S \in B$ is a* monitor *for S whenever* $\mathbf{ext}(b_S) = S$.

Of course, a net system may not have the condition that monitors a given property. However, Theorem 4.1.2 provides that, when the property coincides with a region, then the corresponding condition can be added to the net system without altering its behaviour. In such a case, it is said that the property is observable. Indeed, a property is considered observable when its truth value can be tested without interfering with the system behaviour.

Theorem 4.1.2 states that every region is an observable property of the system. In fact, all observable properties are regions. Whenever a property is not a region, the corresponding monitor must interfere with the behaviour of the system. Indeed, when a subset of states is not a region, it must violate the uniform crossing property. Then there must be an event which does not admit a consistent orientation with it. This brings the problem of defining the flow relation accordingly. As a matter of fact, since the extension of a condition of a net system is always a region of its case graph, a property can only be monitored when it is a region. Given a net system, and a property which is not a region of its case graph, then adding a condition to the net system such that its extension is the property, implies that the case graph of this augmented net system will present the property as a region. This in turn, implies that the two case graphs are not isomorphic, and so the monitor for the property interferes indeed with the behaviour of the system.

Example 4.1.2. *Consider the synthesis of a net system from the transition system in Figure 2.7, taking minimal regions as conditions, as in Figure 4.3. Consider the property with extension $\{q_1, q_2, q_4\}$. This subset is not a region of the system, since one occurrence of e_1 exits it, and another does not. Figure 4.4 shows an attempt to monitor the property on the system. By adding this condition, the behaviour of the system is altered. In fact, it forces the previously concurrent events e_1, and e_2 to occur in sequence.*

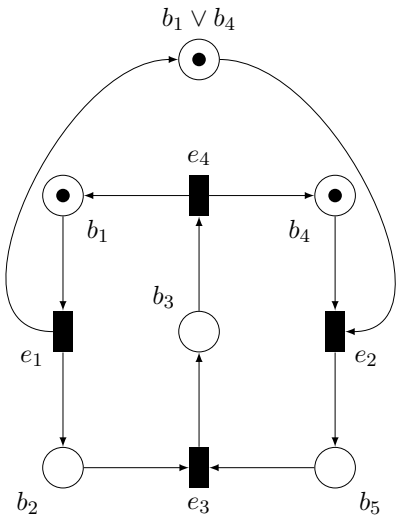

Figure 4.4: Net system obtained from the net system of Figure 4.3, by adding a condition which is not a region.

Hence, regions of the case graph coincide with the observable properties of the system. In the following section, it will be seen that the orthomodular poset of regions allows to reason about these properties in logical terms.

4.1.3 Orthomodular Poset of Regions

This section gathers properties of regions. After being presented, they will be summarised by stating a theorem from [7]. The reader is referred to that work for the proofs. Throughout this section, $A = (Q, E, T, M)$ will be either an elementary or condition/event transition system.

Regions of a condition/event (or equivalently elementary) transition system can be ordered by set inclusion, so that for any two regions $r_1, r_2 \in \mathcal{R}(A)$: $r_1 \leq r_2 \leftrightarrow r_1 \subseteq r_2$. When seen as properties, r_2 holds in every state in which r_1 holds, so this order can be seen as an implication. $\leq$ is quite trivially an order relation, and so $(\mathcal{R}(A), \leq)$ is a partial order.

This section will cover all the axioms of Definition 2.3.6, and show that $(\mathcal{R}(A), \leq)$ is orthomodular.

The negation of a proposition p, is that which holds precisely when p does not. The corresponding properties must then be set complements.

It can be easily checked that $r \in \mathcal{R}(A)$ iff $(Q \setminus r) \in \mathcal{R}(A)$. This provides

$(\mathcal{R}(A), \leq)$ with a unary operation

$$\begin{aligned}(\cdot)' : \mathcal{R}(A) &\to \mathcal{R}(A)\\ r &\mapsto Q \setminus r\end{aligned}$$

Note that both $\emptyset$ and Q are regions, and clearly $\forall r \in \mathcal{R}(A) : \emptyset \subseteq r$ and $r \subseteq Q$. So $(\mathcal{R}(A), \leq)$ is bounded, call $0 = \emptyset$, and $1 = Q$.

Furthermore, it holds that $r \cap (Q \setminus r) = \emptyset$, so 0 is the only element which satisfies both $0 \leq a$ and $0 \leq a'$. Then $a \wedge a' = 0$. Analogously, $r \cup (Q \setminus r) = Q$ implies that $a \vee a' = 1$, and so $(\mathcal{R}(A), \leq)$ is complemented.

Furthermore, $(r')' = Q \setminus (Q \setminus r) = r$, and $r_1 \leq r_2'$ implies that $r_1 \cap r_2 = \emptyset$, and so that $r_2 \leq r_1'$. Hence, $(\mathcal{R}(A), \leq)$ is orthocomplemented.

The observation that, whenever two regions r_1 and r_2 are disjoint, then their union is again a region, is less trivial to show. Intuitively, if none violates the uniform crossing property, then neither does their union. However, the fact that they are disjoint is crucial. In general unions of regions, and intersections of regions may not be regions. The following is a well-known result [5],

Proposition 4.1.2. *Let $A = (Q, E, T)$ be a condition/event transition system. Let $\mathcal{R}(A)$ be its set of regions. Consider $r_1, r_2 \in \mathcal{R}(A)$. Then $(r_1 \cup r_2) \in \mathcal{R}(A)$ iff $(r_1 \cap r_2) \in \mathcal{R}(A)$*

The argument for proving Proposition 4.1.2, is that if the union of two regions violates the uniform crossing property, then this violation must happen to their intersection, and conversely. When regions are seen as properties of the system, Proposition 4.1.2 can be read as 'the disjunction of two observable properties is observable if, and only if, their conjunction is as well'.

When r_1 and r_2 are orthogonal, they are disjoint, and the empty set being a region, $r_1 \vee r_2 \in \mathcal{R}(A)$ is well defined.

Note that this statement can be extended inductively to finite families of pairwise orthogonal elements. Furthermore, when considering a finite transition system, the collection of its subsets will also be finite. Hence, every countable family of regions will be finite. It can be stated that the union of any countable family of pairwis disjoint regions is a region. When seen as properties, the fact that they are disjoint means that none can hold when any of the others does. They imply each other's negation.

When properties mutually exclude each other, their pairwise intersection, as regions, is empty, and corresponds to 0, the constantly false property. 0, as a constant, is always observable. Analogously, 1, the constantly true property is always observable with extension Q.

Finally, it can be shown that if a region r_1 is contained in another region r_2, then the relative complement $r_2 \setminus r_1$ is a region. Note that r_1 and r_2' are disjoint, so $r_1 \vee r_2'$ is well defined. Since $(\mathcal{R}(A), \leq)$ is orthocomplemented, so must be its complement, and out of De Morgan's laws $(r_1 \vee r_2')' = r_2 \wedge r_1'$ is well defined. But $r_2 \setminus r_1 = r_2 \cap (Q \setminus r_1) = r_2 \wedge r_1'$ is the relative complement of r_1 in r_2. Clearly, r_1 and $r_2 \setminus r_1$ are disjoint so their union is a region, in fact $r_2 = r_1 \cup (r_2 \setminus r_1)$. Finally, this last statement is equivalent to the orthomodular law.

$$r_1 \leq r_2 \rightarrow r_2 = r_1 \vee (r_2 \wedge r_1')$$

The following theorem summarises these properties. The proof can be found in [7]

Theorem 4.1.5. *Let A be a condition/event transition system, and $\mathcal{R}(A)$ its set of regions, and define $\forall r \in \mathcal{R}(A) : r' := Q \setminus r$. Then $\langle \mathcal{R}(A), \subseteq, (\cdot)', \emptyset, Q \rangle$ is an orthomodular poset.*

In what follows, when no confusion is possible, $\langle \mathcal{R}(A), \subseteq, (\cdot)', \emptyset, Q \rangle$ will be denoted simply by $\mathcal{R}(A)$.

The converse of Theorem 4.1.5, that every logic is isomorphic to the poset of regions of some transition system does not hold. Chapter 5 will display a collection of counter-examples. The full characterisation of the class of logics which are isomorphic to some poset of regions is an open problem, for which it is pertinent to identify properties which do not hold in orthomodular posets in general, but do when they arise as collections of regions.

Definition 4.1.5 (Regional Logic)**.** *A logic L is called* regional *when there exists either a condition/event, or elementary transition system A such that L is isomorphic to $\mathcal{R}(A)$.*

The role of regional partitions will be fundamental in the rest of this work. These constitute the sequential components of the system.

Consider a condition/event transition system and its corresponding saturated net system. The extensions of conditions can be seen on the transition system, and are regions. A collection of regions is disjoint whenever no marking of the system selects more than one. In this case, the property corresponding to their disjunction is monitored by a condition of the saturated net system, and so is its negation. In a collection of mutually exclusive conditions, none can be marked without unmarking the other, and none can be unmarked without marking another. Consider the union of all sets of their pre- and post-events. The events in this union can never be concurrently enabled. Indeed, the firing of one of these would certainly alter the state of one

of the conditions, disabling any event that would have been enabled. Hence, these events, and conditions, must belong to the same sequential components.

Note that, out of orthomodularity, the complement of the disjoint union of regions is a region, and so any collection of disjoint regions can be extended to a partition. When seen as a set of conditions of the saturated net system, a partition consisting of regions satisfies the requirements of Proposition 2.1.1. Every reachable marking contains exactly one condition in the partition. As a consequence, the subsystem they generate is a state machine.

This provides an interesting application of Corollary 4.1.2. Every condition/event (elementary) net system can be extended with conditions to a state machine decomposable net system with the same behaviour.

4.2 Saturated Transition System

This section revises some properties of regional logics which do not hold in general orthomodular posets, as proved in [7]. When these hold, a synthesis procedure presented in [8] can generate a labelled transition system such that the elements of the orthomodular posets can be identified with regions of the system. This procedure will be presented at the end of the section.

4.2.1 Regularity and Sequential Components

This section presents a property satisfied by regional logics, which does not hold in general. It allows to interpret the maximal Boolean sublogics of a given logic as the sequential components of the corresponding system.

Let $\mathcal{C} \subseteq \mathcal{R}(A)$ be a collection of pairwise disjoint regions, and suppose there is a region $r_1 \notin \mathcal{C}$ properly contained in a region $r_2 \in \mathcal{C}$. Out of orthomodularity, $r_3 = r_2 \setminus r_1 \in \mathcal{R}(A)$, and so $\mathcal{C}' = (\mathcal{C} \setminus \{r_2\}) \cup \{r_1, r_3\}$ is a disjoint collection of regions.

This line of thought inspires the following definition.

Definition 4.2.1 (Refinement)**.** *Let X be an arbitrary set, let $\mathcal{D} \subseteq 2^X$. Consider $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{D}$ two collections of pairwise disjoint subsets. Then $\mathcal{C}_1$ is a* refinement *of $\mathcal{C}_2$, whenever $\forall c_2 \in \mathcal{C}_2 : \exists C \subseteq \mathcal{C}_1 : c_2 = \bigcup_{c \in C} c$. In this case $\mathcal{C}_2$ is said to be* coarser *than $\mathcal{C}_1$, and $\mathcal{C}_1$ is said to be* finer *than $\mathcal{C}_2$.*

Two such collections $\mathcal{C}_1, \mathcal{C}_2 \subseteq \mathcal{D}$ are said to be consistent with respect to *$\mathcal{D}$ when they admit a common refinement $\mathcal{C}_3 \subseteq \mathcal{D}$.*

Refinement is quite trivially an order relation. Consistency is symmetric and reflexive, but in general fails to be transitive.

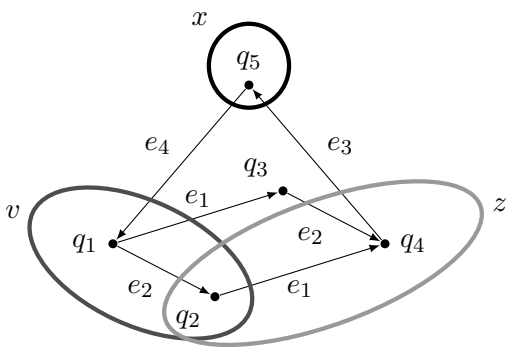

Figure 4.5: Some regions of the transitions system of Figure 2.7 (p.36). b_1 and b_3 are disjoint as subsets of states, so they are orthogonal elements of the logic in Figure 4.1(p.87). On the other hand, b_1 and b_4 have a non-empty intersection as subsets of states, and since $b_1 \cap b_4 = \{q_2\}$ is not a region, b_1 and b_4 are incompatible elements.

Note that if $\mathcal{D} = \mathcal{R}(A)$, each disjoint collection of regions consists of conditions of the saturated net system which belong to the same sequential components. When a disjoint collection $\mathcal{C}_1$ refines another $\mathcal{C}_2$, their sequential components must be the same. To see this, note that the markings of the conditions in $\mathcal{C}_2$ are abstractions of the ones in $\mathcal{C}_1$. Thus, out of Proposition 4.1.1, all events neighbouring the conditions of $\mathcal{C}_2$ must be in the neighbourhood of $\mathcal{C}_1$. This implies that whenever two disjoint collections are consistent (with respect to $\mathcal{R}(A)$) they must also belong to the same sequential components.

A special role, in a logic L, is played by *compatible* elements.

Definition 4.2.2 (Compatibility)**.** *Let L be a logic. Two elements $x, y \in L$ are* compatible*, denoted $x \$ y$ if, and only if, there exist three mutually orthogonal elements $\hat{x}$, $\hat{y}$ and z in L such that $x = \hat{x} \vee z$ and $y = \hat{y} \vee z$.*

When two elements $x, y \in L$ are not compatible, they are said to be incompatible*, denoted $x \not\$ y$.*

Compatibility is reflexive, and symmetric, but fails in general to be transitive. As its complement, incompatibility is irreflexive, and symmetric, but in general not transitive.

Orthogonal elements are always compatible. If $x \perp y$ just take $\hat{x} = x$, $\hat{y} = y$, and $z = 0$. Ordered elements must be compatible as well. If $x \leq y$, out of orthomodularity, $y \wedge x' \in L$, so put $\hat{x} = x$, $z = 0$, and $\hat{y} = y \wedge x'$. Cearly, $\hat{y} = y \wedge x' \leq x' = \hat{x}'$, and so $\hat{y} \perp \hat{x}$. Note that if two regions $r_1, r_2 \in \mathcal{R}(A)$ are

compatible, then there are three disjoint regions $\hat{r}_1, \hat{r}_2$, and $r_3 \in \mathcal{R}(A)$ such that $r_1 = \hat{r}_1 \cup r_3$ and $r_2 = \hat{r}_2 \cup r_3$. Then clearly $r_3 = r_1 \cap r_2$. Conversely, for any two regions satisfying $r_1 \cap r_2 \in \mathcal{R}(A)$, the relative complements $\hat{r}_1 = r_1 \setminus (r_1 \cap r_2) \in \mathcal{R}(A)$, and $\hat{r}_2 = r_2 \setminus (r_1 \cap r_2) \in \mathcal{R}(A)$ show that they are compatible. Hence, two regions are compatible iff their intersection is a region.

Furthermore, for any two compatible regions $r_1, r_2 \in \mathcal{R}(A)$, the disjoint collections $\{r_1, r_2 \setminus r_1\}$ and $\{r_2, r_1 \setminus r_2\}$ are both regional, and admit a common regional refinement $\{r_1 \setminus r_2, r_1 \cap r_2, r_2 \setminus r_1\}$, and so they are consistent. This means, in particular, that there must be at least one sequential component of the saturated net system which contains both r_1 and r_2 $\{r_1 \setminus r_2, r_1 \cap r_2, r_2 \setminus r_1\}$ is called an orthogonal covering of $\{r_1, r_2\}$.

Definition 4.2.3 (Orthogonal Covering, Compatible Set)**.** *Let L be a logic, and let S be a finite subset of L. An* orthogonal covering *of S is a collection F of pairwise orthogonal elements such that $\forall x \in S : \exists G \subseteq F : x = \bigvee G$.*

If S admits an orthogonal covering, it is called a compatible set*.*

Naturally one would expect every set of pairwise compatible elements to be a compatible set. This is however not the case in general. In order for this to happen, an additional axiom is to be considered.

Definition 4.2.4 (Regular Logic)**.** *([51], definition 1.3.26) A logic L is called* regular *if, for any set $\{x, y, z\} \subseteq L$ of pairwise compatible elements, then $x \$ (y \vee z)$.*

In regional terms, regularity is interpreted as follows. Consider any three regions $r_1, r_2, r_3 \in \mathcal{R}(A)$. If their pairwise intersections are in $\mathcal{R}(A)$, then $r_1 \cap (r_2 \cup r_3) \in \mathcal{R}(A)$.

The proof of the following proposition can be found in [51].

Proposition 4.2.1. *A logic L is regular iff every set of pairwise compatible elements is compatible.*

In the terms of regions, compatible sets are a relevant notion regarding the motivation of this work. Indeed, when interpreted as conditions of the saturated net system, every region in a compatible set must belong to the same sequential components as the one that contains its orthogonal covering. A trivial consequence of this, is that every compatible set is contained in some sequential component.

As a matter of fact, in a regional logic, every set of pairwise compatible elements is a compatible set. The following result was shown in [7]

Theorem 4.2.1. *Every regional logic is regular.*

As a consequence, every set of pairwise compatible regions is contained in some sequential component.

At this point, it is worth noting that a collection of pairwise orthogonal elements generates a Boolean algebra.

Proposition 4.2.2. *([51]) Let L be a logic, and let F be a collection of pairwise orthogonal elements. Then $\mathcal{B}(F) := \{\bigvee S \mid S \subseteq F\}$ is a Boolean algebra. Furthermore, if $\bigvee F = 1$, then $\mathcal{B}(F)$ is a sublogic of L.*

The proof shows that joins and meets are well defined among elements of $\mathcal{B}(F)$, and that they distribute over each other. The reader is referred to [51] for the details. Naturally, every collection F of pairwise orthogonal elements can be extended with $f' = (\bigvee F)'$, such that elements of $F \cup \{f'\}$ are pairwise orthogonal, and $f' \vee (\bigvee F) = 1$. Hence, every collection F generates a Boolean sublogic of L.

The following result summarises this section.

Proposition 4.2.3. *([51]) A logic L is* regular *if and only if every pairwise compatible subset of L admits an enlargement to a* Boolean sublogic *of L.*

In terms of regions, when a set is contained in a Boolean sublogic, its elements belong to a same sequential components of the saturated net system. As a matter of fact, sequential components of the system are identified with the maximal Boolean sublogics of its regional poset.

4.2.2 Richness and Concrete Representation

Not every logic is regional. As a matter of fact, a logic in general is not representable as a collection of subsets. When it is, it is said to admit a concrete representation.

Definition 4.2.5 (Concrete Logic)**.** *The couple (Ω, Δ) composed by a set Ω and a collection Δ of subsets of Ω is a* concrete logic *if, and only if, the following conditions are satisfied:*

1. $\emptyset \in \Delta$;
2. $A \in \Delta \Rightarrow \Omega \setminus A \in \Delta$;
3. *if $\{A_i \mid i \in \mathbb{N}\} \subseteq \Delta$ is a countable family of mutually disjoint subsets of Ω, then $\bigcup_{i \in \mathbb{N}} A_i \in \Delta$.*

For every concrete logic (Ω, Δ), $L = \langle \Delta, \subseteq, (\cdot)', \emptyset, \Omega \rangle$ is an orthomodular poset. Ω is called the *carrier set* of L. Indeed, all the axioms in the Definition 2.3.6, of orthomodular posets are shown to hold, for example in [51]. In a concrete logic, $\emptyset$, and Ω play the role of least and greatest elements respectively. Set complement behaves as an orthocomplement, and the orthomodular law holds. When a logic L is isomorphic to a concrete logic (Ω, Δ), it is said to admit a concrete representation. Clearly, every regional logic must be concrete.

Stone's representation theorem for Boolean algebras [60] states that each Boolean algebra is representable as a collection of subsets, with union, intersection, and set complement respectively standing for join, meet, and orthocomplement. As a matter of fact in the finite case, every Boolean algebra is isomorphic to the power set of the set of its atoms.

The analogous result for orthomodular partial order does not hold. Not every logic is a concrete logic. Stanley Gudder provided a characterisation of the logics which are isomorphic to a concrete logic, and the results can be found in, for example [51]. This characterisation relies on the central notion of state. Intuitively, states of the logic will compose the *carrier set* of its concrete representation.

Definition 4.2.6 (Two-Valued State)**.** *A* two-valued state*, or simply* state *on a logic L is a mapping $s : L \to \{0, 1\}$ such that:*

1. $s(1) = 1$;

2. $\forall x, y \in L : x \perp y \to s(x \vee y) = s(x) + s(y)$

An immediate consequence of the definition is that a state s preserves order. Furthermore for any $x \in L$, and $s \in \mathcal{S}(L)$, since $x \vee x' = 1$, it must follows that $s(x) = 1$ iff $s(x') = 0$. States will often be identified with their supports.

In [51] a distinction is made between *states*, that is mappings defined on L whose co-domain is the interval $[0, 1]$, and *two-valued states* as in definition 4.2.6 above. Every two-valued state is a state. The present work reports exclusively *two-valued states*, and for simplicity, the term *state* will refer to a *two-valued state* as in Definition 4.2.6.

Given a logic L, $\mathcal{S}(L)$ will denote the set of all (two-valued) states on L. Given an element $x \in L$, the set of states that contain it will be denoted $\mathcal{S}_x := \{s \in \mathcal{S}(L) \mid s(x) = 1\}$, and called its extension.

When a state is different from another, it always selects an element which is not selected by the other.

Proposition 4.2.4. *Let L be a logic, and $\mathcal{S}(L)$ be its set of states. Consider $s_1, s_2 \in \mathcal{S}(L)$, then*

$$s_1 \neq s_2 \text{ iff } \exists r_1 \in s_1 \setminus s_2 \text{ and } \exists r_2 \in s_2 \setminus r_1$$

Proof. Suppose $s_1 \neq s_2$, then certainly either $\exists r_1 \in s_1 \setminus s_2$, or $r_2 \in s_2 \setminus r_1$. Assume, without loss of generality, that $\exists r \in s_1 \setminus s_2$. Then $r \in s_1 \to r' \notin s_1$, and $r \notin s_2 \to r' \in s_2$. So $r' \in s_2 \setminus s_1$.

The converse is trivial. □

The following properties of states can easily be derived from the definition.

Proposition 4.2.5 (Properties of two-valued states)**.** *Let L be a logic, and consider $s \in \mathcal{S}(L)$. The following properties hold:*

1. $\forall x, y \in L : x \leq y \to (s(x) = 1 \to s(y) = 1)$
2. $\forall x, y \in L : s(x \vee y) = 0 \to (s(x) = 0 \wedge s(y) = 0)$
3. $\forall x, y \in L$: *if* $x \,\$\, y$ *then* $s(x \vee y) = 1 \to (s(x) = 1 \wedge s(y) = 0)$ *or* $(s(x) = 0 \wedge s(y) = 1)$.

As a consequence of these, the support $\boldsymbol{supp}(s)$ *is* upwards closed.

$$\forall x \in \boldsymbol{supp}(s) : \uparrow\{x\} \subseteq \boldsymbol{supp}(s)$$

The support of a state is also weakly downwards directed

$$\forall x, y \in \boldsymbol{supp}(s) : (x \,\$\, y \text{ and } x \wedge y \neq 0) \to x \wedge y \in \boldsymbol{supp}(s)$$

In the following, the term state will be used either for states or their supports. The notion of state becomes clear when considering regions.

Proposition 4.2.6. *Let $A = (Q, E, T)$ be a transition system, and $\mathcal{R}(A)$ its regional logic. Then for any state $q \in Q$, the collection $\mathcal{R}_q = \{r \in \mathcal{R}(A) \mid q \in r\}$ is the support of a state.*

Proof. Trivially, $q \in Q$, and so $Q \in \mathcal{R}_q$. Given two regions $r_1, r_2 \in \mathcal{R}(A)$, $r_1 \perp r_2 \to r_1 \cap r_2 = \emptyset$. Then $q \in r_1 \to q \notin r_2$, and $q \in r_2 \to q \notin r_1$. Clearly, either $q \in r_1$ or $q \in r_2$ lead to $q \in r_1 \cup r_2$. Finally, note that if $q \notin r_1$ and $q \notin r_2$, then $q \notin r_1 \cup r_2$, which completes the proof. □

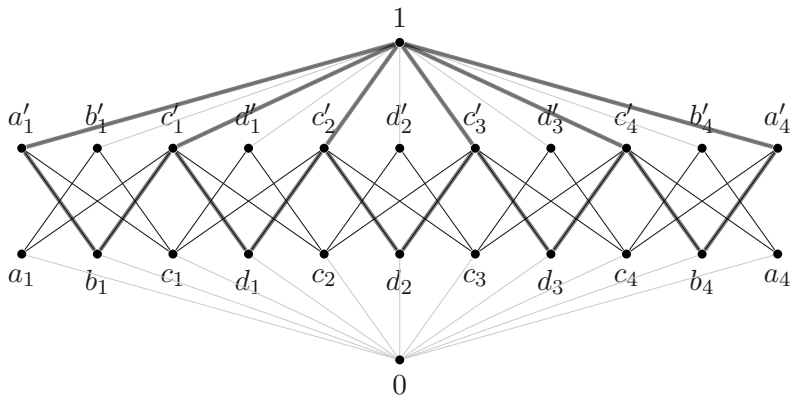

Figure 4.6: A logic with five Boolean sublogics. Each pattern of the form $\{0, a_1, b_1, c_1, a'_1, b'_1, c'_1, 1\}$ forms one of them. The support of a state is highlighted.

To every state of the system, there corresponds a state of the logic. States of the logic are to be considered as the admissible markings of the corresponding saturated net system. However, not every state of the logic is expressed as a state of the system. Such a state might not be reachable with the provided set of events E. In the condition/event case, consider the Petri net underlying the saturated net system. A marking is considered admissible on the net, when the resulting system is state machine decomposable. Indeed, different markings may lead to different behaviours, and an admissible marking might not be in the class defined by the condition/event transition system. However, the reachability class of any admissible marking provides a case graph which will be separated by the regions of the logic.

In general, given a condition/event transition system $A = (Q, E, T)$, and its regional logic $\mathcal{R}(A)$, it holds that $\forall r \in \mathcal{R}(A) : r = \mathcal{S}_r \cap Q$.

In general, logics can have no state at all. Logics having "enough" states in such a way that the order relation can be re-constructed by the evaluation of the states are called *rich*.

Definition 4.2.7 (Rich Logic)**.** *Let L be a logic and $x, y \in L$. L is* rich *iff:*

$$\mathcal{S}_x \subseteq \mathcal{S}_y \rightarrow x \leq y$$

Note that the Property 1. in Proposition 4.2.5 can be restated as $\forall x, y \in L : x \leq y \rightarrow \mathcal{S}_x \subseteq \mathcal{S}_y$, and holds in any logic. A logic is rich when the converse also holds.

Intuitively, a rich logic is one faithfully represented by its set of states. And in fact, Gudder's theorem shows that rich logics admit a concrete representation.

Theorem 4.2.2. *A logic L is isomorphic (as a logic) to a concrete logic if, and only if, it is rich.*

The proof of this theorem can be found, for example, in [51]. It uses the fact that, if a logic L is rich, there is a duality between L and the set $\mathcal{S}(L)$: each element in L is fully characterised by the set of states to which it belongs

Since regional logics are concrete, Theorem 4.2.2 implies that they must be rich. To get some intuition about this fact, suppose that a condition/event transition system $A = (Q, E, T)$ satisfies that to every state of its regional logic there corresponds a reachable state of the system, namely $\forall s \in \mathcal{S}(\mathcal{R}(A)) : \exists q \in Q : \mathcal{R}_q = s$. Then for every region $r \in \mathcal{R}(A)$, $\mathcal{S}_r = \{\mathcal{R}_q \in \mathcal{S}(\mathcal{R}(A)) \mid q \in r\}$. Suppose that $r_1 \leq r_2$, then $r_1 \subseteq r_2$, and $\forall q \in r_1 : q \in r_2$. Hence $\{\mathcal{R}_q \in \mathcal{S}(\mathcal{R}(A)) \mid q \in r_1\} \subseteq \{\mathcal{R}_q \in \mathcal{S}(\mathcal{R}(A)) \mid q \in r_2\}$, and so $\mathcal{S}_{r_1} \subseteq \mathcal{S}_{r_2}$.

4.2.3 Events as Local Transitions

In this section, representability of a logic as the regions of a transition system is tackled. When the logic is rich, sets of states can be defined on the logic so that elements of the logic can be retrieved as its subsets. However, as a power set, the collection of all subsets of states would systematically provide a Boolean algebra. The extension of elements of the logic can certainly be found in such a collection. One would wish to impose that only the extensions of elements of the logic can be found as regions of this space of states. In order to do so, one should endow this space of states with labelled transitions, so that the uniform crossing property forbids the existence of unwanted regions, the ones which are not the extension of an element of the logic.

A straightforward approach is to consider all pairs of different states as potential transitions, and determine which of them carry the same label. The resulting transition system could be seen a complete directed graph with a labelling function defined on the arcs. As such it will be called saturated. It presents all the events consistent with the sequential components described by the maximal Boolean algebras of the logic. It is now explained how one can derive the labelling of transitions. The saturated transition system was described in [8]. In that work, the authors followed a different nomenclature, consistent with that of [37]. In particular, in [7], *regular* logics are there called *coherent*, *rich* logics are called *prime*, and *states* are called *prime filters*. In this work, the nomenclature is taken from [51].

Consider a condition/event net system $N = (B, E, \mathcal{F}, Q)$, and its case graph $\mathbf{CG}(N) = A = (Q, E, T)$. In Section 4.2.2, it was seen that the set $\mathcal{R}_q$ of regions containing a state $q \in Q$ is a two-valued state of $\mathcal{R}(A)$.

Petri's principle of extensionality [48] allows one to identify events with their observable effect. Indeed, the net underlying N is simple. Hence, for any pair of events $e_1, e_2 \in E$, if ${}^\bullet e_1 = {}^\bullet e_2$, and $e_1^\bullet = e_2^\bullet$ it must be $e_1 = e_2$. This allows for the identification of each event $e \in E$ with its observable effect $\langle {}^\bullet e, e^\bullet \rangle$.

It was shown in Proposition 2.2.2, that $\forall b \in B : {}^\bullet b = {}^\bullet \mathbf{ext}(b)$ and $b^\bullet = \mathbf{ext}(b)^\bullet$. Now, consider $N_S = (\mathcal{R}(A), E, \mathcal{F}', Q)$, the saturated net system associated with A. Definition 4.1.1 expresses $\mathcal{F}'$ as $\{(r, e) \in \mathcal{R}(A) \times E \mid e \in r^\bullet\} \cup \{(e, r) \in E \times \mathcal{R}(A) \mid e \in {}^\bullet r\}$ Then trivially, Proposition 2.2.2 gives a one-one correspondence between the observable effect of an event, on the saturated net system, and its incidence with respect to each of the regions. Any occurrence of an event determines its observable effect. If $(q_1, e, q_2) \in T$ is a transition, then ${}^\bullet e = \mathcal{R}_{q_1} \setminus \mathcal{R}_{q_2}$, and $e^\bullet = \mathcal{R}_{q_2} \setminus \mathcal{R}_{q_1}$. Two transitions carrying the same label, must have the same observable effect. It follows from the definition of region that these differences are independent of the individual occurrence of e in A. The proof of the following statement was already given in [30, 31].

Proposition 4.2.7. *Let $A = (Q, E, T)$ be a condition/event transition system. If $(q_1, e, q_2), (q_1', e, q_2') \in T$, then $\mathcal{R}_{q_1} \setminus \mathcal{R}_{q_2} = \mathcal{R}_{q_1'} \setminus \mathcal{R}_{q_2'} = {}^\bullet e$, and $\mathcal{R}_{q_2} \setminus \mathcal{R}_{q_1} = \mathcal{R}_{q_2'} \setminus \mathcal{R}_{q_1'} = e^\bullet$.*

These determine the flow relation of the corresponding saturated net system. This is consistent with the principle of locality, the effect of an event is only observable locally.

This characterisation of events according to their incidence on the set of regions (or equivalently their flow relation on the saturated net system) can be extended to the whole set of regions, and used to determine the label of the saturated transition system.

Definition 4.2.8 (Saturated Transition System)**.** *Let L be a rich and regular logic. Let $\mathcal{S}(L)$ be its set of states. An* event *of L is the observable effect of the transition from a state $s_1 \in \mathcal{S}(L)$ to another $s_2 \in L$.*

$$[s_1, s_2] = \langle s_1 \setminus s_2, s_2 \setminus s_1 \rangle$$

Note that whenever $s_1 = s_2$, the corresponding event $e = \langle \emptyset, \emptyset \rangle$ has no observable effect. According to Petri's principle of extensionality, this is not an

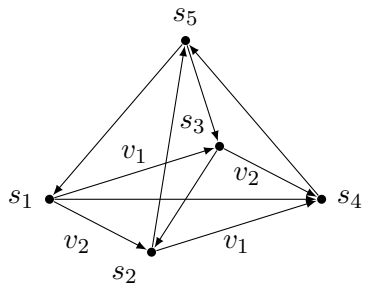

Figure 4.7: Saturated Transition System synthesised from the logic in Figure 4.1 (p.87). For clarity, not all transitions have been depicted: for each arc in the figure, there is another one in the opposite direction. Also, only repeated labels have been indicated.

event. The set of events *of L is thus defined as*

$$E(L) = \{[s_1, s_2] \mid s_1, s_2 \in \mathcal{S}(L), s_1 \neq s_2\}$$

The set of transitions *of a logic is now naturally defined as*

$$T(L) = \{(s_1, [s_1, s_2], s_2) \mid s_1, s_2 \in \mathcal{S}(L), s_1 \neq s_2\}$$

The saturated transition system *associated with L is then defined as*

$$A(L) = (\mathcal{S}(L), E(L), T(L))$$

Example 4.2.1. *The transition system partially depicted on Figure 4.7, is the saturated transition system built from the logic on Figure 4.1 (p.87).* $s_1 = \uparrow\{b_1, b_5\}$, $s_2 = \uparrow\{b_1, b_4\}$, $s_3 = \uparrow\{b_2, b_5\}$, $s_4 = \uparrow\{b_2, b_4\}$, *and* $s_5 = \uparrow\{b_3\}$. *Note that the symmetric differences involving* s_5 *are all different, since it is the only state containing* b_3. *Analogously,* $\langle s_1 \setminus s_4, s_4 \setminus s_1\rangle$ *has only one occurrence, as well as* $\langle s_4 \setminus s_1, s_1 \setminus s_4\rangle$, $\langle s_2 \setminus s_3, s_3 \setminus s_2\rangle$, *and* $\langle s_3 \setminus s_2, s_2 \setminus s_3\rangle$. *However,* $\langle s_1 \setminus s_2, s_2 \setminus s_1\rangle = \langle s_3 \setminus s_4, s_4 \setminus s_3 rlangle = v_2$, $\langle s_2 \setminus s_1, s_1 \setminus s_2\rangle = \langle s_4 \setminus s_3, s_3 \setminus s_4\rangle$, $\langle s_1 \setminus s_3, s_3 \setminus s_1\rangle = \langle s_2 \setminus s_4, s_4 \setminus s_2\rangle = v_1$, *and* $\langle s_3 \setminus s_1, s_1 \setminus s_3\rangle = \langle s_4 \setminus s_2, s_2 \setminus s_4\rangle$.

Note that the logic that generates this transition system is the regional logic of Example 4.1.1. It is worth noting, with respect to that example, that $s_i = \mathcal{R}_{q_i}$, *for* $i = 1, \ldots, 5$. *Analogously, every element of the logic* b_i, *provides a region of this synthesised transition system as* $\mathcal{S}_{b_i}$. *In fact, the regional poset of this system is isomorphic to the one on Figure 4.1.*

Note that two transitions carry the same label when they have the same observable effect. As a consequence, the following result holds. The proof can be found in [7].

Proposition 4.2.8. *If* $\exists(s_1, e, s_2) \in T$ *then* $s_2 = (s_1 \setminus {}^\bullet e) \cup e^\bullet$.

A transition is defined for every ordered pair of states, and so the underlying graph is complete, and strongly connected. Then any state is trivially reachable from any other.

Proposition 4.2.9. *Let* L *be a rich logic.*
Then for any $s_0 \in \mathcal{S}(L)$, $\mathrm{A}(L) = (\mathcal{S}(L), E(L), T(L), s_0)$ *is an initialised transition system.*

The proof of this statement is trivial. The following properties follow from the construction.

Proposition 4.2.10. $\forall s_1, s_2 \in \mathcal{S}(L)$:

1. $s_1 \neq s_2 \rightarrow (\exists r \in s_1 \setminus s_2 : s_1 \in \mathcal{S}_r$ *and* $s_2 \notin \mathcal{S}_r$
2. $\forall e \in E(L) : (s_1, e, s_2) \notin T(L) \rightarrow \exists r \in {}^\bullet e : r \notin s_1$
3. $\forall e \in E(L) : (s_1, e, s_2) \notin T(L) \rightarrow \exists r \in e^\bullet : r \notin s_2$

Proof. Property 1 is a direct consequence of Proposition 4.2.4. Properties 2 and 3 follow from Proposition 4.2.8 □

As a consequence, L separates $A(L)$, providing the following theorem [7].

Theorem 4.2.3. *Let* L *be a rich and regular logic, then* $A(L)$ *is a condition/event transition system*

Note that, out of Proposition 4.2.9, this also implies that for any $s_0 \in \mathcal{S}(L)$, $(\mathcal{S}(L), E(L), T(L), s_0)$ is an elementary transition system.

The remaining of this chapter will analyse this construction, and endow $E(L)$ with a partial group structure.

4.3 Model Reduction

This section will present the contributions of this chapter. In this section, the structure of the transition system synthesised from a logic is analysed. Finite logic can be represented in terms of their atoms, and it is shown that the saturated transition system is also described in this representation. This is used to analyse the structure of its events, in order to identify concurrency in terms of atoms of the logic. It is shown that a particular subset of events is sufficient to describe the concurrent behaviour of the saturated transition system.

4.3.1 Minimal Regions and Block Diagrams

Stone's representation theorem states that the elements of a Boolean algebra can be identified with particular subsets of its set of ultra filters. A filter is an upwards closed, downwards directed subset of the Boolean algebra. A filter is called ultrafilter when it is maximal.

In the case of orthomodular posets, Gudder's representation expresses the logic as a collection of subsets of its set of states. Unsurprisingly, when restricting a state to a maximal Boolean subalgebra of the logic, one obtains an ultrafilter. Note that a finite Boolean algebra is a particular case of orthomodular poset, and in such a situation, both representations coincide.

Atomic Boolean algebras are in particular atomistic, and their ultra filters are the up-closures of their atoms. Hence, in the finite Boolean case, the set of atoms can be taken as the carrier for representing the algebra.

In the case of logics, however, states represent a selection of atoms for each maximal Boolean subalgebra, and the correspondence between states and atoms does not hold.

The problem of representing an atomic orthomodular poset using its set of atoms as a carrier, instead of its set of states, was tackled early in the theory [33, 52]. D. Foulis and C. Randall greatly contributed to the development of this theory by proposing the notion of *manual*, as an axiomatic formalisation of the spaces suitable to represent the atoms of a quantum logic [53, 54, 55]. A manual is a set endowed with a collection of its subsets which identify their belonging to the same maximal Boolean algebra. In his Ph.D. dissertation [26], J. C. Dacey characterised the class of manuals which generate an orthomodular poset, by requiring an additional axiom. However, this axiomatisation only allowed for generating regular logics, called coherent in this field of research. Alternative axiomatisations for these spaces followed, among which it is worth noting *orthogonality spaces* [38] where an orthogonality relation is

taken instead of the collection of subsets. Finally, a commonly used axiomatisation, is that of *test spaces*. With this axiomatisation non-regular logics can be generated. Still regularity of the generated logic can be asserted by requiring an additional axiom. The reading of [61] is recommended for a good overview of the subject.

The present section explores this approach in order to represent the transition system synthesised from a logic, in terms of the atoms of the latter. To this aim, the link between the two alternative representation theories is explored.

The expressivity of atoms regarding the structure of an orthomodular lattice was already explored, and exploited in Chapter 3. In the case of regional structures, their relevance is justified by Theorem 4.1.4. Indeed, the atoms of a regional logic are the minimal regions. They are sufficient to solve the synthesis problem, and so one would expect that considering them should be sufficient to accurately represent, and distinguish the sequential components of the system.

An element of a quantum logic L is called an *atom* whenever it is minimal for the partial order when one excludes 0. Let $\mathcal{A}(L)$ be the set of such elements. When restricted to atoms, the compatibility relation \$ coincides with orthogonality relation $\perp$.

Proposition 4.3.1. *Let L be an orthomodular poset, and let $x, y \in \mathcal{A}(L)$. Then*

$$x \,\$\, y \leftrightarrow x \perp y$$

Proof. From Definition 4.2.2, $x \,\$\, y \leftrightarrow \exists \hat{x}, \hat{y}, z \in L : x = \hat{x} \vee z$ and $y = \hat{y} \vee z$, with $\hat{x}, \hat{y}$ and z pairwise orthogonal. If $x \perp y$ just take $\hat{x} = x$, $\hat{y} = y$, and $z = 0$. For the converse, note that since $x, y \in \mathcal{A}(L)$, the only possibilities are $\hat{x}, z \in \{x, 0\}$, and $\hat{y}, z \in \{y, 0\}$. Then z must be in $\{x, 0\} \cap \{y, 0\} = \{0\}$, and so $\hat{x} = x$, and $\hat{y} = y$, hence $x \perp y$. □

As a consequence, the atoms of a maximal Boolean sublogic form a maximal clique of $\perp$.

A compact graphical representation of finite logics can be obtained with a technique due to Richard Greechie.

Remark 4.3.1. *The set of elements L of a logic, or any of its subsets S, together with a symmetric relation such as \$, $\not\$$, or $\perp$ can be considered as an undirected graph. All notions of graph theory as in [4] can then be applied. For instance, a* clique *of a symmetric relation is any subset of L such that*

each pair of its elements is in R. Let S be clique of R, then it is maximal *if* $\forall x \notin S : (\exists y \in S : x \not R\, y)$.

From point 3 in Definition 2.3.6, it is clear that any element of the logic can be retrieved as the join of a subset of a clique of $\perp$, and actually, the pair $(\mathcal{A}(L), \perp\restriction_{\mathcal{A}(L)\times\mathcal{A}(L)})$ is sufficient to recover the whole structure of the logic.

The *block diagram* of a logic exploits this fact, and depicts only the atoms of the logic. Instead of representing all orthogonality dependencies, maximal cliques of $\perp$ are represented by straight lines.

Remark 4.3.2. *Technically, these straight lines represent the hyperedges, associated with the maximal cliques of* $\perp$*, of the hypergraph whose vertices are* $\mathcal{A}(L)$ *(see for example [4]).*

In this way, the *block diagram* of a Boolean algebra will be composed only by one maximal clique of $\perp$ while at least two *blocks* are needed for the representation of a non-Boolean logic L. A result will now be presented, that will allow us to characterise the states of the logic on its block diagram. Informally, this result states that a state must contain exactly one atom per maximal Boolean sublogic. The formalisation of this calls for some notation.

Definition 4.3.1. *Let* $\mathcal{C}_\perp(\mathcal{A}(L))$ *be the set of maximal cliques of the* $\perp$ *relation.*
Let $\mathcal{C}_{\text{NC}}(\mathcal{A}(L))$ *be the set of maximal cliques of* $\not\$$.
$CL(\mathcal{A}(L))$ *will denote the set of elements* $\alpha \in \mathcal{C}_{\text{NC}}(\mathcal{A}(L))$ *such that* $\forall \beta \in \mathcal{C}_\perp(\mathcal{A}(L))$, $|\alpha \cap \beta| = 1$.

The next Theorem, summarises some theorems and lemmas published in [7] In that work, they were presented for regular logic (there called coherent. The next statement generalises that result, in that regularity is not required.

Theorem 4.3.1. *Let* $\langle L, \leq, (.)', 0, 1\rangle$ *be a logic,* $\mathcal{A}(L)$ *its set of atoms,* $s \in \mathcal{S}_L$ *and* $CL(\mathcal{A}(L))$ *as in definition 4.3.1, then* $(s \cap \mathcal{A}(L)) \in CL(\mathcal{A}(L))$*. Moreover, if* $\alpha \in CL(\mathcal{A}(L))$ *then* $\uparrow\alpha$ *is a state in L.*

Proof. Note that $\$ \cap (\mathcal{A}(L) \times \mathcal{A}(L)) = \perp$, and let $\beta = s \cap \mathcal{A}(L)$, then $x, y \in \beta \Rightarrow x \not\$\, y$. Indeed, suppose $x \,\$\, y$, then $x, y \in \mathcal{A}(L) \Rightarrow x \perp y$, and so by definition of $\perp$: $x \leq y'$. Then by the orthomodular law $y' = x \vee (y' \wedge x')$, and so $y' \in s$, hence $y \notin s$, which contradicts the hypothesis. Thus, β is a clique of $\not\$$. It will now be show that $\forall \gamma \in \mathcal{C}_\perp(\mathcal{A}(L)) : |\gamma \cap \beta| = 1$, so let

$\gamma = \{z_1, z_2, \ldots, z_n\} \in \mathcal{C}_\perp(\mathcal{A}(L))$. Then $z_\gamma = \bigvee_{i\leq n} z_i$ is well defined. If $z_\gamma \neq 1$, then $\forall i \leq n : (z_i \leq z_\gamma \rightarrow (\exists a \in \mathcal{A}(L) : a \leq z'_\gamma \wedge a \perp z_i))$, contradicting maximality of γ. Hence $z_\gamma = 1$, and it must hold, from the definition of state, that $\exists! i \leq n : s \cap \gamma = \{z_i\}$. Now suppose $\exists z \in \mathcal{A}(L) \setminus \beta$ such that $z \not\S x$ for every $x \in \beta$. Since it is an atom, $z \notin s$ but z must be in some maximal clique $\gamma \in \mathcal{C}_\perp(\mathcal{A}(L))$. As it has just been proven, $\exists \hat{z} \in \beta \cap \gamma$, so either $\hat{z} = z$ or $\hat{z} \perp z$, yielding a contradiction in both cases. So $\beta \in CL(\mathcal{A}(L))$.

Let now $\alpha \in CL(\mathcal{A}(L))$, and let $a \in \alpha$. Since 1 is, by definition, the greatest element of L, certainly $a \leq 1$. Now let $\{x_1, x_2, \ldots, x_n\}$ be mutually orthogonal elements of L, and let $x = \bigvee_{i\leq n} x_i$. Then certainly, $\forall i \leq n : x_i \leq x$. Consider x_i, and let $a \leq x_i$ be an atom, the orthomodular law provides $x_i = a \vee (x_i \wedge a')$. Put $x_i^1 = x_i \wedge a'$, $A^1 = \{a\}$. Now suppose A^k is a set of pairwise orthogonal atoms $a \leq x_i$. Let $x_i^k = x_i \wedge (\bigwedge_{a\in A^k} a')$, and suppose $x_i = (\bigvee_{a\in A^k} a) \vee x_i^k$. Then $x_i^k \neq 0$ implies there is an atom $a^k \leq x_i^k$. In particular $a^k \leq x_i$, and $\forall a \in A^k : a^k \leq a'$. Put $A^{k+1} = A^k \cup \{a^k\}$, then from the orthomodular law, it follows that again that $x_i^k = a^k \vee (x_i^k \wedge (a^k)') = a^k \vee x_i^{k+1}$. Furthermore, $x_i = (\bigvee_{a\in A^k} a) \vee x_i^k = (\bigvee_{a\in A^{k+1}} a) \vee x_i^{k+1}$. Clearly, $x_i^{k+1} \lneq x_i^k$, so L being finite, there must exists an $N \in \mathbb{N}$ such that $x_i^N = 0$, then A^N is a set of pairwise orthogonal atoms, such that $x_i = (\bigvee_{a\in A^N} a) \vee 0 = \bigvee_{a\in A^N} a$. And so, for each x_i, there is a set A_i of pairwise orthogonal atoms such that $x_i = \bigvee_{a\in A_i} a$. Now Let a_i be an atom such that $a_i \leq x_i$ for some x_i. It follows that $x'_i \leq a'_i$, then for any $x_j \neq x_i$, and for any atom a_j under it, it holds that $a_j \leq x_j \leq x'_i \leq a'_i$. Hence, $\bigcup_{i\leq n} A_i$ is a set of pairwise orthogonal atoms, and so there is a $\gamma \in \mathcal{C}_\perp(\mathcal{A}(L)) : \bigcup_{i\leq n} A_i \subseteq \gamma$. Put $A_0 = \gamma \setminus \bigcup_{i\leq n} A_i$, since $x = \bigvee_{a\in(\bigcup_{i\leq n} A_i)} a$, maximality of γ gives $x' = \bigvee_{a\in A_0} a$, put $x' = x_0$.

Clearly $\exists! a_0 \in \alpha \cap \gamma$, so let $i_0 \leq n : a_0 \in A_{i_0}$. Let $j \neq i_0$, since $\forall a \in A_j : a \leq a'_0$, then $x_j = (\bigvee_{a\in A_j} a) \leq a'_0$, and so $\alpha \in \mathcal{C}_{\text{NC}}(\mathcal{A}(L))$ implies $\forall b \in \alpha \setminus \{a_0\} : b \not\leq x_j$. If $i_0 = 0 : a_0 \leq x'$, otherwise $a_0 \leq x_{i_0} \leq x$.

□

Example 4.3.1. *With reference to Figure 4.1 (see p.87)the states of L are:* $\uparrow\{a, d\}, \uparrow\{a, e\}, \uparrow\{b, d\}, \uparrow\{b, e\}, \uparrow\{c\}$. *A maximal clique of* $\S$ *is not, in general, sufficient for the definition of a state. The requirement, as in definition 4.3.1, that each maximal clique of* $\S$ *meets each maximal clique of* $\perp$ *is essential. With reference to Figure 4.9, the up-closure of the maximal clique* $\{a_1, b_2, g_1\}$ *of* $\S$, $\uparrow\{a_1, b_2, g_1\}$ *is not a state since an element from the block* $\{c_1, c_2, c_3\}$ *is missing.*

Example 4.3.2. *The regional logic L of the transition system in Figure 4.8 is represented, as* block diagram *of its atoms. The minimal regions in L,*

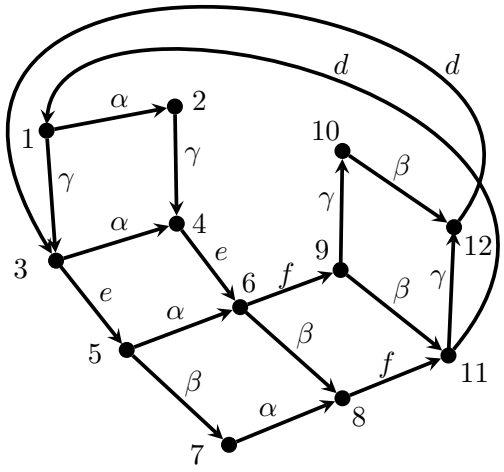

Figure 4.8: A condition/event transition system. It is the case graph of the net system of Figure 4.10, the block diagram of its logic is represented in Figure 4.9 p.113

corresponding to the atoms, are: $a_2 = \{2,4,6,8\}$, $a_1 = \{1,3,5,7\}$, $g_1 = \{1,2,9,11\}$, $g_2 = \{3,4,10,12\}$, $b_2 = \{7,8,11,12\}$, $b_1 = \{5,6,9,10\}$, $c_1 = \{9,10,11,12\}$, $c_2 = \{1,2,3,4\}$ *and* $c_3 = \{5,6,7,8\}$. *Its representation as* Net System *is in Figure 4.10 where only one representative of the class of markings is drawn. This marking is composed by the regions* $c_2 = \{1,2,3,4\}$, $a_1 = \{1,3,5,7\}$ *and* $g_1 = \{1,2,9,11\}$.

A new result is now presented, that allows for characterising events on the block diagram of L. It is a straightforward consequence of Theorem 4.3.1, and will be useful in the following section.

The events of a logic L are fully characterised by the atoms in their neighbourhoods. Recall that the neighbourhood of an event e is $\nu(e) = {}^\bullet e \cup e^\bullet$

Definition 4.3.2 (Atomic Neighbourhood)**.** *For each* $e \in E(L)$, *the* atomic neighbourhood *of* e *is:* $\nu_{\mathcal{A}}(e) = ({}^\bullet e \cap \mathcal{A}(L)) \cup (e^\bullet \cap \mathcal{A}(L))$

Lemma 4.3.1. *Let* $e_1, e_2 \in E(L)$. *If* $\nu_{\mathcal{A}}(e_1) = \nu_{\mathcal{A}}(e_2)$, *then* $e_1 = e_2$.

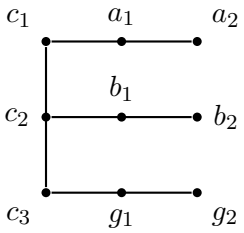

Figure 4.9: The regional logic of the transition system in Figure 4.8

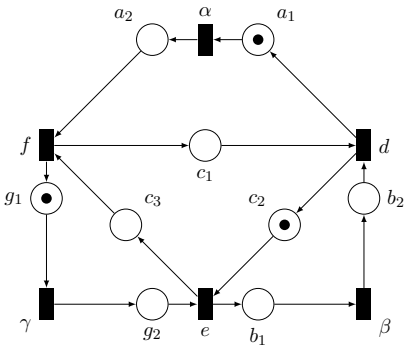

Figure 4.10: Net System representation for the Transition System in Figure 4.8, the marking corresponds to state 1

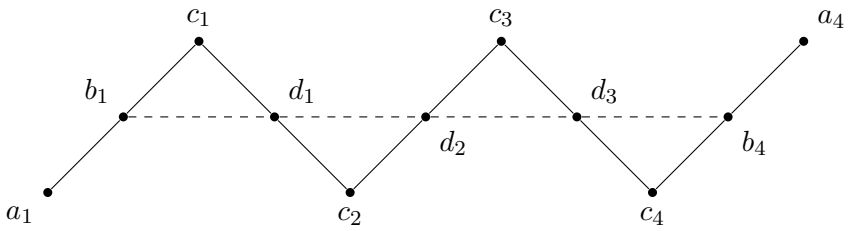

Figure 4.11: A block diagram of the logic in Figure 4.6 (p.103). Each solid segment is a block, the dashed line represent a clique $\alpha \in CL(\mathcal{A}(L))$ such that its up-closure corresponds to the state in Figure 4.6.

Proof. Let $e_1 = \langle s_1 \setminus s_2, s_2 \setminus s_1 \rangle$. Choose $x \in s_1 \setminus s_2$, and suppose that x is not an atom of L. Then there is at least one atom a of L in s_1, with $a < x$. From $x \notin s_2$, it follows that $a \notin s_2$. A symmetric argument applies to $s_2 \setminus s_1$, hence e_1 cannot be distinguished from e_2 because of a non-atomic element in their neighbourhoods. □

With these results at hand, the structure of the transition system synthesised from a logic can be analysed in terms of atoms.

4.3.2 Partial group of Events

The results of this section can be found in [15].

The set of events of the saturated transition system contains the labels of all possible transitions between states. This form of completeness, permits to endow it with a richer structure. A composition operation is defined, based on sequential composition. Since the saturated transition system is saturated of events, this operation has associated notions of inverse, and neutral element. This idea was already noted as a remark in [20] The neutral element will just be $e_\emptyset := \langle \emptyset, \emptyset \rangle$ the empty event. This event should be considered only for structural purposes. Intuitively, it would just leave any state unchanged; hence no transition carries it as a label.

First the operation of sequential composition of two events is defined. The resulting event will correspond to the consecutive occurrence of its operands. The existence of this new event will depend, however, on the existence of a state binding the operands in sequence. In this setting, events are defined over a fixed structure of states, making the following a partial operation.

Let L be a regional logic.

Definition 4.3.3 (Sequentiable Events)**.** *Two events $e_1, e_2 \in E(L)$ are said to be* sequentiable *iff*

$$\exists s \in \mathcal{S}(L) : e_1^\bullet \subseteq s \wedge {}^\bullet e_2 \subseteq s \tag{4.1}$$

If two events are sequentiable, their sequential composition is:

$$\begin{aligned} e_1 \oplus e_2 &= \langle ({}^\bullet e_1 \setminus e_2^\bullet) \cup ({}^\bullet e_2 \setminus e_1^\bullet), (e_2^\bullet \setminus {}^\bullet e_1) \cup (e_1^\bullet \setminus {}^\bullet e_2) \rangle \\ &= \langle ({}^\bullet e_1 \cup {}^\bullet e_2) \setminus (e_1^\bullet \cup e_2^\bullet), (e_1^\bullet \cup e_2^\bullet) \setminus ({}^\bullet e_1 \cup {}^\bullet e_2) \rangle \end{aligned}$$

The definition does not depend on the choice of s. Condition 4.1 requires an occurrence of e_1 to bring the system to a state which enables e_2. The following result exemplifies why this is required, and shows that $e_1 \oplus e_2$ belongs to $E(L)$.

Proposition 4.3.2. *Let $s_0, s, s_1 \in \mathcal{S}(L)$ be three distinct states, and $e_1, e_2 \in E$ be events such that $s_0[e_1\rangle s[e_2\rangle s_1$. Then $s_0[e_1 \oplus e_2\rangle s_1$*

Proof. By definition of E, $e_1 = \langle s_0 \setminus s, s \setminus s_0 \rangle$, and $e_2 = \langle s \setminus s_1, s_1 \setminus s \rangle$. Then by definition of $\oplus$, $e_1 \oplus e_2 = \langle ((s_0 \setminus s) \cup (s \setminus s_1)) \setminus ((s \setminus s_0) \cup (s_1 \setminus s)),$ $((s \setminus s_0) \cup (s_1 \setminus s)) \setminus ((s_0 \setminus s) \cup (s \setminus s_1)) \rangle$. Straightforward set operations then lead to $e_1 \oplus e_2 = \langle s_0 \setminus s_1, s_1 \setminus s_0 \rangle$.

□

Associativity of the composition comes as a consequence of this last result. The inverse of an event is now defined.

Definition 4.3.4 (Inverse Event)**.** *Let* $e \in E : e = \langle {}^\bullet e, e^\bullet \rangle$*. Then* $e^{-1} = \langle e^\bullet, {}^\bullet e \rangle$ *is the* inverse *of* e*.*

The transition system synthesised from a logic is saturated with events. For any ordered pair of states, E contains an event which labels the corresponding transition. Hence, as stated in the following proposition, any event has an inverse.

Proposition 4.3.3. *Let* $e \in E$ *then* $e^{-1} \in E$*.*

Proof. $e \in E \Rightarrow \exists s, s' \in \mathcal{S}_L : s[e\rangle s'$. By definition, it holds that $e = \langle s \setminus s', s' \setminus s \rangle$. Now, $\exists t \in T : t = (s', [s', s], s)$, and clearly, it will carry as a label $\langle s' \setminus s, s \setminus s' \rangle = e^{-1} \in E$. □

The inverse is well defined for all event, and so is the composition $e \oplus e^{-1} = e^{-1} \oplus e = e_\emptyset$.

Whenever composition of two events is well defined, inversion behaves accordingly, in the sense that $(e_1 \oplus e_2)^{-1} = e_2^{-1} \oplus e_1^{-1}$ whenever they are defined.

The sequential composition operation is, of course, not commutative in general. Concurrency of two events can be formalised as the commutativity of their sequential composition: the occurrence of any of them does not disable the other.

Definition 4.3.5 (Independent Events on a Logic)**.** *Two events* $e_1, e_2 \in E$ *are called* independent *iff*

$$\nu_{\mathcal{A}}(e_1) \cap \nu_{\mathcal{A}}(e_2) = \emptyset$$

Note that this condition holds iff $({}^\bullet e_1 \cup e_1^\bullet) \cap ({}^\bullet e_2 \cup e_2^\bullet) = \emptyset$

They are called concurrent *if they are independent, and there exists a state that enables both of them:*

$$\exists s \in \mathcal{S}(L) : {}^\bullet e_1 \cup {}^\bullet e_2 \subseteq s \wedge (e_1^\bullet \cup e_2^\bullet) \cap s = \emptyset$$

It follows from independence that, when two events are concurrent, their composition becomes simply $e_1 \oplus e_2 = \langle {}^\bullet e_1 \cup {}^\bullet e_2, e_1^\bullet \cup e_2^\bullet \rangle$.

The next proposition shows that two events are concurrent if and only if their composition is commutative and some state enables both of them. This correspondence between concurrency among two events, and commutativity of their composition accurately translates the idea that concurrent events form diamonds in the saturated transition system. This fact is used to prove the following result.

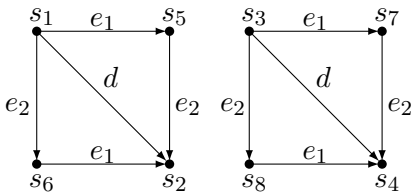

Figure 4.12: Two diamonds

Proposition 4.3.4. *Let $e_1, e_2 \in E$; then they are concurrent iff $e_1 \oplus e_2$ and $e_2 \oplus e_1$ are well defined and equal, and there exists a state which enables both.*

Proof. It is first proved that if e_1 and e_2 are concurrent, then their composition is commutative. So let $s, s_1, s_2, s' \in \mathcal{S}(L) : s[e_1\rangle s_1 \wedge s_1[e_2\rangle s' \wedge s[e_2\rangle s_2$. The firing rule yields $s_1 = (s \setminus {}^\bullet e_1) \cup e_1^\bullet$, and $s' = (s_1 \setminus {}^\bullet e_2) \cup e_2^\bullet$, and it follows from independence of e_1 and e_2 that $s' = (s \setminus ({}^\bullet e_1 \cup {}^\bullet e_2)) \cup (e_1^\bullet \cup e_2^\bullet) = (s_2 \setminus {}^\bullet e_1) \cup e_1^\bullet$. Hence $s_2[e_1\rangle s'$.

For the converse, suppose $s, s_1, s_2, s' \in \mathcal{S}(L) : s[e_1\rangle s_1 \wedge s_1[e_2\rangle s' \wedge$ $s[e_2\rangle s_2 \wedge s_2[e_1\rangle s'$. Assume, as a contradiction hypothesis that $\exists a \in \nu_{\mathcal{A}}(e_1) \cap \nu_{\mathcal{A}}(e_2) \neq \emptyset$. There are four cases. If $a \in {}^\bullet e_1 \cap {}^\bullet e_2$ then $a \notin s_1$ and so $s_1[\not{e_2}\rangle$. If $a \in {}^\bullet e_1 \cap e_2^\bullet$ then $a \in s_2$, and so $s[\not{e_2}\rangle s_2$. Analogously $a \in {}^\bullet e_2 \cap e_1^\bullet$ implies that $s[\not{e_1}\rangle s_1$. Finally, If $a \in e_1^\bullet \cap e_2^\bullet$ then $s_1[\not{e_2}\rangle$. □

The following example should clarify the last proof. The composition of two concurrent events is a diagonal of the diamond they form. In the net system interpretation, this corresponds to an event which has the same effect as firing the operands simultaneously. This view justifies the term of *step*.

Example 4.3.3. *In Figure 4.12 one can see that $s_1[e_1\rangle s_5[e_2\rangle s_2$, and $s_1[e_2\rangle s_6[e_1\rangle s_2$. The fact that from s_1 the system reaches s_2 after firing e_1 and e_2 regardless of the order of firing, implies that $e_1 \oplus e_2 = e_2 \oplus e_1 = d$.*

An event is called a *step* if it is the diagonal of some diamond. Whenever e_1, e_2 are concurrent events, $e = e_1 \oplus e_2 = e_1$ **step** e_2, and say that e is the step $\{e_1, e_2\}$. This composition operation will serve as a tool for analysing the inner structure of events, seen as pairs of subsets of atoms.

4.3.3 Minimal Events

The results of this section can be found in [15].

There is an essential difference between the non-commutative and the commutative composition of events. Indeed, commutative composition provides the diagonal of a diamond. Such a diagonal event should be distinguished from other events. Provided that a system already depicts two concurrent events, their step provides no additional information regarding concurrency, or connectedness of the system. This section will be devoted to formalising, and proving this last statement.

In order to do so, the structure of diamonds is studied, as seen in the Greechie representation of the logic. In the following, states of the logic will be considered as the cliques of incompatibility of the atoms which intersect every block of the logic as in theorem 4.3.1. Events will therefore be seen as the symmetric differences of these. For the basic notions on graph theory, see [4].

Definition 4.3.6 (Induced Subgraph)**.** *Let* $G_\perp(L) = (\mathcal{A}(L), \perp\restriction_{\mathcal{A}(L)\times\mathcal{A}(L)})$ *be the* Orthogonality Graph *of a Regional Logic* L*. Given a subset of atoms* $A \subseteq \mathcal{A}(L)$*,* $(A, \perp\restriction_{A^2})$ *is called the* subgraph *of* $G_\perp(L)$ induced *by* A*.*

Recall that each maximal clique of $G_\perp(L)$ corresponds to the set of atoms of a block (maximal Boolean sublogic) of L.

A *bipartite graph* is a graph whose set of vertices can be partitioned in two classes, such that no two vertices in the same class are bound by an edge (see [4]). A state, seen as a subgraph of $G_\perp(L)$, has the property that no pair of its vertices is in $\perp$. It will be seen that this implies that, for each event e, its set of pre-conditions and its set of post-condition, form a bipartite graph $({}^\bullet e, e^\bullet)$, when considered as the subgraph of $G_\perp(L)$ induced by $\nu_{\mathcal{A}}(e)$.

Proposition 4.3.5. *Let* $e \in E$*, and* $G_\perp(L)$ *as in Definition 4.3.6. Then* $({}^\bullet e \cap \mathcal{A}(L), e^\bullet \cap \mathcal{A}(L))$ *is a bipartition of* $(\nu_{\mathcal{A}}(e), \perp\restriction_{\nu_{\mathcal{A}}(e)^2})$*.*

Proof. $e \in E$ implies $\exists s, s' \in \mathcal{S}(L) : s[e\rangle s'$. Then ${}^\bullet e \subseteq s$ and $e^\bullet \subseteq s'$. ${}^\bullet e \subseteq s \Rightarrow \forall a_1, a_2 \in {}^\bullet e : a_1, a_2 \in s$. From Theorem 4.3.1, it follows that $a_1 \not\# a_2$, so in particular $(a_1, a_2) \notin \perp$. The result is analogous for $e^\bullet$ and s'. □

The converse, however, is not always the case. Here follows an example of a logic, in which there is a bipartite subgraph of $G_\perp(L)$ which does not correspond to an event.

Example 4.3.4. *In Figure 4.9, consider* $(\{a_1, b_1, g_1, a_2, b_2, g_2\})$ *as an induced subgraph. In this case,* $(\{a_1, b_1, g_1\}, \{a_2, b_2, g_2\})$ *forms a bipartition, but there is no event* $e = \langle\{a_1, b_1, g_1\}, \{a_2, b_2, g_2\}\rangle$*, since no state enables it. Such a state*

would be a maximal clique of $\not\$$, but as seen in Example 4.3.2, $\{a_1, b_1, g_1\}$ does not intersect every block of L, and is therefore not a state.

In the following, the properties of diamonds, as seen on $G_\perp(L)$, are studied. The next result will exhibit a property that any pair of concurrent events must satisfy.

Proposition 4.3.6. *Let e_1 and e_2 be two concurrent events. Then $\forall a_1 \in {}^\bullet e_1, \forall a_2 \in e_2^\bullet : a_1 \not\$ a_2$, and symmetrically $\forall a_3 \in e_1^\bullet, \forall a_4 \in {}^\bullet e_2 : a_3 \not\$ a_4$.*

Proof. Since e_1 and e_2 are concurrent, they must form a diamond. Namely, $\exists s, s_1, s_2, s' \in \mathcal{S}(L) : s[e_1\rangle s_1[e_2\rangle s' \wedge s[e_2\rangle s_2[e_1\rangle s'$. Hence, $e_1^\bullet \cup {}^\bullet e_2 \subseteq s_1$, and $e_2^\bullet \cup {}^\bullet e_1 \subseteq s_2$. Theorem 4.3.1 then yields the result. □

This last result might become clearer when looking at a Greechie diagram.

Example 4.3.5. *Consider the events $e_1 = \langle\{a_1\}, \{a_2\}\rangle$, and $e_2 = \langle\{b_1\}, \{b_2\}\rangle$ from Figure 4.9. They are both enabled at state $s = \{a_1, b_1, c_3\}$. The existence of the state $s_1 = \{a_2, b_1, c_3\}$ implies that $a_2 \in e_1^\bullet$, and $b_1 \in {}^\bullet e_2$ satisfy $a_2 \not\$ b_1$. Symmetrically, the state $s_2 = \{a_1, b_2, c_3\}$ provides $a_1 \not\$ b_2$.*

Now, the much stronger converse result shall be proven. Namely, it is shown that given an event, seen as the bipartite induced subgraph of $G_\perp(L)$, it can be determined whether it is the step of two concurrent events, and if so, actually reconstruct the corresponding diamond.

Lemma 4.3.2. *Connected components of the subgraph of $G_\perp(L)$ induced by an event e, form a set of pairwise concurrent events, the step of which is e.*

Proof. Let $e \in E$, and suppose that there is a partition $\{a_i\}_{i\leq n}$ of ${}^\bullet e$, and a partition $\{b_i\}_{i\leq n}$ of $e^\bullet$ such that $a_i \cup b_i$ are the connected components of $(\nu_{\mathcal{A}}(e), \perp\restriction_{\nu_{\mathcal{A}}(e)^2})$, for $i \leq n$. Proceed by induction over the states. For the base case, consider $s, s' \in \mathcal{S}(L) : s[e\rangle s'$. Now, let $j \leq n$, then $(s \setminus a_j) \cup b_j$ must be a clique of $\not\$$, since otherwise there would be an $r_1 \in \bigcup_{i\neq j} a_i$ and an $r_2 \in b_j$ such that $r_1 \perp r_2$, contradicting that $a_j \cup b_j$ is a connected component for $\perp$.

Now suppose $(s \setminus a_j) \cup b_j$ is not a state. Then there must be a block B of $G_\perp(L)$ such that $((s \setminus a_j) \cup b_j) \cap B = \emptyset$. $s \in \mathcal{S}(L)$ implies that $\exists r \in s \cap B$, so it must be $r \in a_j$. Analogously, $s' \in \mathcal{S}(L)$ implies that $\exists r' \in s' \cap B$, so it must be $r' \in \bigcup_{i\neq j} b_i$. Clearly, $r, r' \in B \to r \,\$\, r'$, and ${}^\bullet e \cap e^\bullet = \emptyset \to r \neq r'$, so $r \perp r'$. This contradicts the fact that $a_j \cup b_j$ is a connected component. Then $s_j = (s \setminus a_j) \cup b_j$ must be a state, and there must be an $e_j = \langle a_j, b_j\rangle \in E$ such that $s[e_j\rangle s_j$.

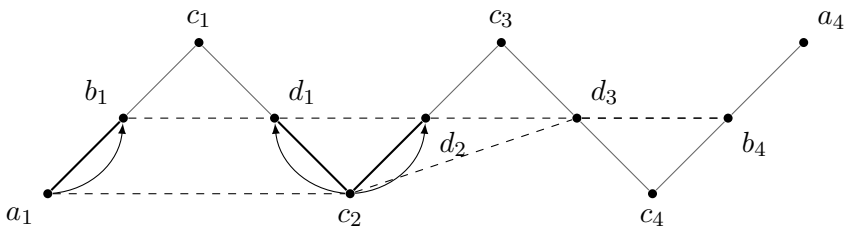

Figure 4.13: The block diagram depicted in Figure 4.11 (p.113). Events are seen as symmetric differences of the cliques of atoms that generate states as their up-closures.

The induction step is absolutely analogous. One only needs to consider the event $e'_j = \langle \bigcup_{i \neq j} a_i, \bigcup_{i \neq j} b_j \rangle$, noting that $s_j[e'_j\rangle s'$.

The result holds for any choice of $j \leq n$ at any induction step, hence all permutations of n provide a sequential decomposition of the event e. Note that, by construction, and for any $j \leq n$, the events e_j are pairwise independent, and the existence of the state s provides that they must be concurrent. It should be clear that $\forall j_1 \neq j_2 : e_{j_1} \oplus e_{j_2} = e_{j_2} \oplus e_{j_1} \in E$. It follows from associativity of $\oplus$ that $\bigoplus_{i \leq n} e_i = e$. □

This lemma represents the main technical contribution of this section, and most of the results henceforth presented are consequences of it. Indeed, it is a powerful tool which will permits to decompose events, as the step of a family of pairwise concurrent events.

Example 4.3.6. *With reference to Figure 4.13, the states are seen as the cliques of atoms that generate them. The arrows represent an event labelling the transition that goes from the state* $\uparrow\{a_1, c_2, d_3, b_4\}$ *to the state* $\uparrow\{b_1, d_2, d_2, d_3, b_4\}$. *It also labels, for example, the one that goes from state* $\uparrow\{a_1, c_2, c_4\}$ *to state* $\uparrow\{b_1, d_2, d_2, c_4\}$. *When the atomic neighbourhood of the event is not connected for* $\perp$, *it has two connected components* $\{a_1, b_1\}$, *and* $\{c_2, d_1, d_2\}$. *Then the two parts of the event corresponding to each component can be fired independently, the events are concurrent.*

In order to formalise the notion of minimality of the events which are no further decomposable in this way, a partial order of events.

Definition 4.3.7 (Partial Order of Events)**.** *Let* E *be a set of events. Define* $< \subseteq E^2$ *as* $\forall e_1, e \in E : e_1 < e$ *iff* $\exists e_2 \in E : e_1 \oplus e_2 = e_2 \oplus e_1 = e$.

Proposition 4.3.7. $(E, <)$ *is a strict partial order*

Proof. $<$ is irreflexive since, by convention, $e_\emptyset \in E$ is not considered. $<$ is transitive, as a consequence of Proposition 4.3.6, and independence of concurrent events. $<$ is antisymmetric, since $e_1 \oplus e_2 = e_2 \oplus e_1 = e$, and $e_3 \oplus e = e \oplus e_3 = e_1$ imply that $e_1 \oplus e_2 \oplus e_3 = e_1$, and so $e_3 = e_2^{-1}$. Hence $s[e\rangle$ implies that $s[e_2\rangle$ and $s[e_2^{-1}\rangle$, but then there must be an $s' \in \mathcal{S}(L) : s'[e_2\rangle s[e_2\rangle$, which contradicts the separation axioms in Definition 2.2.8 (p.38). □

An event is *minimal* if it is minimal with respect to this partial order. From Lemma 4.3.2, it should be clear that an event is minimal if and only if it is connected, as a bipartite induced subgraph of $G_\perp(L)$. The main result presented in this section is that it is sufficient to consider minimal events when synthesising a transition system, in order for it to be connected, but also to depict all the concurrency encoded in the logic. This last notion will be formalised by comparing the set of regions of such a transition system, and that of the canonical synthesised system, which is saturated with events.

Definition 4.3.8. *Call $M \subseteq E$ the set of events which are minimal in $(E, <)$.*

It is first shown that M is sufficient to generate the regions of the transition system synthesised from L. The idea is rather simple. When two concurrent events are in E, then the separation axioms (see Definition 2.2.8 p.38) corresponding to their step are redundant.

Definition 4.3.9. *Let $A = (Q, E, T)$ be a transition system, and $G \subseteq E$ be a subset of events of A. Define T_G as the set of all transitions of A labelled by some element of G. Then*

$$A \setminus G = (Q, E \setminus G, T \setminus T_G)$$

Clearly, $A \setminus G$ is a, possibly non-connected, transition system.

Lemma 4.3.3. *Let $A = (Q, E, T)$ be a condition/event transition system, $s_0, s_1, s_2, s_3 \in Q$, and $e_1, e_2, d \in E$, such that $s_0[e_1\rangle s_1[e_2\rangle s_3$, $s_0[e_2\rangle s_2[e_1\rangle s_3$, and $s_0[d\rangle s_3$. Then A and $A \setminus \{d\}$ have the same set of regions.*

Proof. Any region of A is also a region of $A \setminus \{d\}$, since the latter has been obtained by removing an event. Suppose now that r is a region of $A \setminus \{d\}$, but not of A. This implies that there are two transitions in A, say (s_1, d, s_2) and (s_3, d, s_4), with different crossing relations with respect to r. Suppose that $s_1 \in r$, $s_2, s_3, s_4 \notin r$ (all other combinations can be dealt with in a similar way). By hypothesis, ${}^\bullet d = {}^\bullet e_1 \cup {}^\bullet e_2$, and $d^\bullet = e_1^\bullet \cup e_2^\bullet$. Hence, e_1 and e_2 are enabled at s_1 and at s_3, and form two diamonds, as shown in Figure 4.12.

Suppose that s_5 is in r. Then, e_1 is orthogonal to r, and $e_2 \in {}^\bullet r$; but this leads to a contradiction, since $s_7 \notin r$, and (s_7, e_2, s_4) does not cross the border of r. On the other hand, $e_5 \notin r$ contradicts the hypothesis that $s_3 \notin r$, because in this case e_1 should leave r. □

This last result implies, in particular, that minimal events convey all the information regarding concurrency, all other events being steps of these. This is formalised in the following theorem.

Theorem 4.3.2. *Let L be a rich, and regular logic. Let $A = (\mathcal{S}(L), E, T)$ be the saturated transition system synthesised from L. Let M be the set of minimal events in $(E, <)$. Then the Regional logic $\mathcal{R}(A)$ is isomorphic to $\mathcal{R}((\mathcal{S}(L), M, T_M))$.*

Proof. The proof follows an induction over the set of states. Starting from A, apply Lemma 4.3.3 to show that for any diagonal d, $\mathcal{R}(A) \simeq \mathcal{R}(A \setminus \{d\})$. In the inductive step, the hypothesis will be that $\mathcal{R}(A \setminus D) \simeq \mathcal{R}(A)$, then for any step d in $E \setminus D$, Lemma 4.3.3 provides $\mathcal{R}(A \setminus (D \cup \{d\})) \simeq \mathcal{R}(A)$. Clearly, when D contains all the steps of E, then $E \setminus D = M$. □

Theorem 4.3.3. *Let M be a set of minimal events in $(E, <)$, and let T_M be the set of all transitions carrying some label in M, then the graph representation of the transition system $(\mathcal{S}(L), M, T_M)$ is connected.*

Proof. By construction, the graph associated with the saturated synthesised transition system $(\mathcal{S}(L), E, T)$ is a complete graph. It is therefore connected. Now let $s, s' \in \mathcal{S}(L)$. The proof follows an induction. Suppose $[s, s']$ is not a minimal event. Then Lemma 4.3.2 shows that $\exists s_{1/2} \in \mathcal{S}(L) : s[[s, s_{1/2}]\rangle s_{1/2}[[s_{1/2}, s']\rangle s'$. For the inductive step, assume there is a path $s[e_1\rangle s_1[e_2\rangle \ldots [e_i\rangle s_i[e_{i+1}\rangle s'$. Now, from Lemma 4.3.2 it holds that $\forall e_{j+1} \notin M : \exists s_{j1/2}$ such that $s_j[[s_j, s_{j1/2}]\rangle s_{j1/2}[[s_{j1/2}, s_{j+1}]\rangle s_{j+1}$.

Clearly,$s[e_1\rangle s_1[e_2\rangle \ldots\ s_j[[s_j, s_{j1/2}]\rangle s_{j1/2}[[s_{j1/2}, s_{j+1}]\rangle s_{j+1} \ldots\ [e_i\rangle s_i[e_{i+1}\rangle s'$ is a path connecting s and s'. The induction will then end with a path connecting s and s' where all arcs correspond to minimal events. □

A direct consequence of this last theorem, is that the transition system built as the saturated transition system but depicting only minimal events satisfies the first axiom of Definition 2.2.3.

Chapter 5

A Logic for Concurrent Systems

Characterising the class of orthomodular partial orders that arise as the structure of observable properties of distributed systems is the main goal in this line of research. The problem is of particular relevance for the purpose of this work. When an orthomodular partial order is not suitable for representing local states of a system, the transition system built from it might present less concurrent behaviour than intended. In [7] the problem was state, of whether given an orthomodular partial order, then its saturated transition system carries a set of regions isomorphic to it, whenever it arises as the set of regions of some elementary transition system. It could be said that such an orthomodular partial order is *stable* under synthesis of a transition system. In such a situation, the two orthomodular partial orders that specify the boundaries for the behaviour of the saturated transition system turn out to be isomorphic. Since the saturated transition system presents all the synchronisations specified by the logic it is built from, and its set of regions specify the concurrency it carries, if the two were isomorphic they would fully characterise the way it can be distributed. When an orthomodular poset is stable, its saturated transition system, as well as the system of minimal events presented in the last chapter, can be distributed precisely as specified by the partial order.

However, not all orthomodular posets are stable. In particular, if an orthomdular poset is not rich, then the synthesised transition system may not have enough states to distinguish regions as subsets of these. If it is not regular, then the synchronisations specified by its orthogonality relation will force the saturated system to have more sequential components than the ones specified by the order. The fact that all regional partial orders are rich derives

from classical results in the theory. Order structures of elementary regions were also shown to be regular and to satisfy the so called triple intersection property. The conjecture about stability under synthesis turned into a problem of characterisation. Indeed, one searches for non-general properties of orthomodular partial orders, which always hold when these arise as sets of elementary regions. Ideally these properties would help to prove that ordered sets of regions are stable under synthesis.

In particular, in [7] richness was exploited to show that to every element of an orthomdular partial order, there corresponds a region of its saturated transition system, providing an injection of the first structure into the partial order of regions of the system, which preserves the order. From regularity, it was also derived that the compatibility relation must be preserved by this injection, making it an actual embedding. In this way, in order to prove the conjecture, it would be sufficient to show that such an embedding is surjective, thus providing the desired isomorphism. In other terms, one would like to show that if an orthomodular partial order is stable under synthesis, then no additional property than the specified is actually observable on the synthesised system.

In this chapter, another property of orthomodular posets, named ETI, is presented. It follows from the idea that the incompatibility relation translates concurrency on the system, and that this concurrency must be expressed by events. After formalising this notion, it will be shown to hold on every regional partial order. Furthermore, it is shown that with this additional condition, the embedding between the two logics of the stability problem preserves incompatibility. Equivalently, such an embedding reflects compatibility, a condition that ranks it up from simple embedding to strong embedding. Of course, not every embedding is strong, but every isomorphism is. In this way, the class of stable orthomodular partial orders is narrowed down by the ETI property. Indeed, every regional partial order satisfies it.

Intuitively, just like regularity forces the saturated transition system to present the specified sequential components, ETI imposes concurrency to be present as specified by incompatibility. In particular, two incompatible regions correspond to properties which are observable from different sequential components. When the embedding preserves incompatibility, two elements of a logic specified to be in distinct sequential components will comply with the specification as regions of the saturated transition system. They would correspond to properties observable from different localities. Thus, if ETI holds in the specification, incompatibility can be used to represent independence of local states. The embedding being strong, this independence is ensured to hold in the saturated transition system. As a consequence, minimal regions

of the specification, remain minimal in the transition system. They properly contain no new region. Still, it is not guaranteed that no spurious synchronisations may arise. Incompatibility, as a binary relation only ensures that pairwise independence is preserved by the system. In proving that no unspecified property is observable in the saturated system, one would try to extend these notions of incompatibility and independence to arbitrary subsets of regions. However, the range of possible combinations of this binary relation leaves the problem of fully characterising stable orthomodular posets out of the scope of this work.

Stability is proved,in the last section of this chapter, for restricted classes of logics, in which the incompatibility relations follows given patterns. It is first shown for trivial relations of incompatibility. Indeed, an empty incompatibility encodes the fact that the system should present no concurrency. It is further shown, that the parallel composition specifications of sequential components remains stable under synthesis. It finally proved for independent sequential components synchronising only through the same single local state.

5.1 Stability of Regional Logics

In the present section, The problem of whether the embedding of a logic L into the ordered set of regions of the transition system constructed from L is an isomorphism is exposed. Some counter-examples are proposed for the general case, and the problem is restated by requiring of L to be a regional logic.

5.1.1 Non-Regional Orthomodular Posets

Given a logic L, one can build its saturated transition system $A(L)$. One can then obtain a new logic as $\mathcal{R}(A(L))$. Just like a transition system is isomorphic to the case graph of its saturated net system, one would expect L to be isomorphic to $\mathcal{R}(A(L))$; this is in general not the case.

It is conjectured that when a logic L arises a the set of regions of a condition/event system, then it is isomorphic to $\mathcal{R}(A(L))$. In fact, all known examples of regional logics are isomorphic to the The present section displays some examples in which the saturated transition system obtained from a logic L does not have a regional logic isomorphic to L. This should provide some intuition on how to search for axioms that should be satisfied by regional logics, in order for isomorphism to exist.

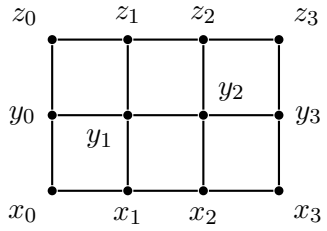

Figure 5.1: Block diagram representation of a logic with 3 maximal Boolean algebras of 4 atoms interfacing with 4 Boolean algebras of 3 atoms.

Example 5.1.1. *Consider the logic L represented in Figure 5.1. It has 7 blocks. The horizontal blocks are characterized by the letters x_i, y_i, z_i, and the vertical ones by the indices $i \in \{0,1,2,3\}$. The states of this logic can be identified on the block diagram by using Theorem 4.3.1. A state is determined by its set of atoms. They are a collection of atoms α which intersects each block at a singe element. Hence, α should contain enough atoms to represent every letter, and every index. Let $k_1 \in \{0,1,2,3\}$, and suppose $\alpha \cap \{x_i\}_{i\leq 3} = x_{k_1}$. Then $y_{k_1} \perp x_{k_1} \to y_{k_1} \notin \alpha$, and analogously for z_{k_1}. So let $k_2 \in \{0,1,2,3\} \setminus \{k_1\}$, and suppose $\alpha \cap \{y_i\}_{i\leq 3} = y_{k_2}$. For each $k_3 \in \{k_1, k_2\}$, either $z_{k_3} \perp x_{k_1}$, or $z_{k_3} \perp y_{k_2}$, and so $z_{k_3} \notin \alpha$. Then pick $k_3 \in \{0,1,2,3\} \setminus \{k_1, k_2\}$, and suppose $\alpha \cap \{z_i\}_{i\leq 3} = z_{k_3}$. The remaining index $k_4 \in \{0,1,2,3\} \setminus \{k_1, k_2, k_3\}$ corresponds to a block, and so it must be $\alpha \cap \{x_{k_4}, yk_4, zk_4\} \neq \emptyset$. But $x_{k_4} \perp x_{k_1}$, $y_{k_4} \perp y_{k_2}$, and $z_{k_4} \perp z_{k_3}$, and so α can not represent a state. In particular this logic has no states. As a consequence $\mathrm{A}(L) = (\emptyset, \emptyset, \emptyset)$ is the trivial transition system, and its set of regions is $\mathcal{R}(\mathrm{A}(L)) = \{\emptyset\}$. This degenerate case is, according to the first axiom of Definition 2.3.6, not even a logic, although some authors agree in including it in the class. In any case $L \not\simeq \mathcal{R}(\mathrm{A}(L))$.*

The last example showed a logic with no states, an extreme case of non rich logic. Its concrete representation is degenerate, and justifies the necessity of regional logics being rich.

The following example displays a non regular logic, with the aim of justifying the regularity requirement for regional logics.

Example 5.1.2. *Consider the set $\Omega = \{i \in \mathbb{N} \mid 1 \leq k \leq 8\}$. Let $\Delta = \{S \subseteq \Omega \mid \exists k \in \mathbb{N} : |S| = 2{*}k\}$ be the collection of subsets of Ω with even cardinality. The pair (Ω, Δ) is a concrete logic, therefore Δ forms a rich logic. The axioms of Definition 4.2.5 are easily verified to hold. This logic, however, is not regular. Consider the elements $a = \{1,2,3,4\}$, $b = \{1,2,5,6\}$, and $c = \{1,3,5,7\}$ of Δ.*

They are pairwise compatible. Consider, for example $a \cap b = \{1,2\} \in \Delta$*, then the set* $\{\{3,4\},\{1,2\},\{5,6\}\}$ *is an orthogonal covering of* $\{a,b\}$*, and so* $a \,\$\, b$*. Analogously,* $a \,\$\, c$*, and* $b \,\$\, c$*. If* Δ *were regular it should be hold that* $a \,\$\, (b \vee c)$*. This is not the case, since* $b \vee c = \{1,2,3,5,6,7\}$*, and* $a \cap (b \vee c) = \{1,2,3\}$ *has odd cardinality, and is not an element of* Δ*. The set* $\{a, b \vee c\}$ *does not admit an orthogonal covering, and so the logic fails to be regular. As a matter of fact, there is no collection of pairwise orthogonal elements such that their joins generate all three* a*,* b*, and* c*.*

Now, one can still try to synthesise a transition system as described in the last chapter. Since (Ω, Δ) *is a concrete logic, the set of states of* Δ *is known to be isomorphic to* Ω*. Let* $x, y \in \Omega$*, and consider the corresponding states of* Δ*. Out of Theorem 4.3.1, these are characterised by the atoms of their support, and so it is sufficient to consider* $s_x = \{a \in \mathcal{A}(\Delta) \mid x \in a\}$*, and* $s_y = \{a \in \mathcal{A}(\Delta) \mid y \in a\}$*. Note that the atoms of* Δ *are the subsets of* Ω *consisting of two elements, so* $s_x \setminus s_y = s_x \setminus \{x, y\}$*. Suppose* $x', y' \in \Omega$ *are such that* $s_{x'} \setminus s_{y'} = s_x \setminus \{x, y\}$*. Clearly, they must be* $x' = x$ *and* $y' = y$*. Hence the transition* $(s_x, \langle s_x \setminus s_y, s_y \setminus s_x \rangle, s_y) \in T(\Delta)$ *is the only one carrying this label. Thus, all labels on the saturated transition system are different, and the uniform crossing property imposes no constraints on the regions:* $\mathcal{R}(\mathrm{A}(\Delta)) = 2^{\Omega}$*, the Boolean algebra with 256 elements. In particular, all singletons are regions, and* $\Delta \neq \mathcal{R}(\mathrm{A}(\Delta))$*. This logic is not stable.*

Note that in this last example, the fact that the synthesis of a transition system fails to provide a set of regions isomorphic to the logic does not depend on the cardinality of the carrier. This motivates the following example.

Example 5.1.3. *Consider the set* $\Omega = \{i \in \mathbb{N} \mid 1 \leq k \leq 6\}$*. Let* $\Delta = \{S \subseteq \Omega \mid \exists k \in \mathbb{N} : |S| = 2 * k\}$ *be the collection of subsets of* Ω *with even cardinality. Just like in Example 5.1.2,* Δ *forms a rich logic. Note that, in this case, the synthesis procedure is analogous to the one in that example, and the saturated transition system built on* Δ *will have no repeated labels. In this case, as well, its set of regions will be the Boolean algebra of the parts of* Ω*,* $2^{\Omega} = \mathcal{R}(\mathrm{A}(\Delta)) \neq \Delta$*.* Δ *is not stable.*

However, in this case, Δ *is regular. To see this consider three pairwise compatible elements* a*,* b*, and* c *of* Δ*. Then, in particular, there is an orthogonal covering* $\{\tilde{b}, d, \tilde{c}\}$ *for* $\{b, c\}$*. Since* (Ω, Δ) *is concrete,* $d = b \cap c$ *must have even cardinality or be* $\emptyset$*. The cardinality of* d *admits 4 values. If* $|d| = 6$*, then clearly* $|b| = |d| = |c|$*, and so* $b = d = c = \Omega$*, and so* $a \subset \Omega$ *implies that* $a \,\$\, b \vee c$*. If* $|d| = 4$*, Then either* $b = d$*, or* $c = d$*, since the only element of* Δ *strictly greater than* d *is* Ω*. Suppose* $b = d$*, then* $b \vee c = c$*, and so* $a \,\$\, b \vee c$*.*

If $|d| = 0$*, then* $b = \tilde{b}$*, and* $c = \tilde{c}$*, so* $b \perp c$*. Hence,* $b \cap c = \emptyset$ *and since* $a \$ b$ *and* $a \$ c$*,* $a \cap b \in \Delta$*, and* $a \cap c \in \Delta$*. Therefore,* $a \cap (b \cup c) = (a \cap b) \cup (a \cap c)$ *must have even cardinality and* $a \$ (b \cup c)$*. Now, suppose* $|d| = 2$*, and suppose that* $a \cap d$ *has an odd cardinality. Then* $a \$ b$ *implies there must be an* $x \in (b \cap a) \setminus c$*, and analogously, a* $y \in (c \cap a) \setminus b$*. Then both* b *and* c *have at least a cardinality of* $|d|+1$*, and since they are in* Δ*, it follows that these must be either* 4 *or* 6*. If* c *has cardinality* 6*, then* $b \leq c$*, and so* $b = d$*, contradicting that* $a \$ b$*. If* b *has cardinality* 6*, then analogously,* $c = d$ *is a contradiction. So suppose that* $|b| = |c| = 4$*. Then* $|b \cap c| = 2$ *means that* $b \cup c = \Omega$*, and then* $|a| = |a \cap b| + |a \cap c| - |a \cap d|$ *must be odd, contradicting* $a \in \Delta$*.*

Hence (Ω, Δ) *is regular.*

This last example is the most relevant for the characterisation of regional logics. It shows that being rich and regular is not sufficient to be regional, and guides the research of properties of regional logics.

There is an apparent pattern in the last two examples, they are constructed as the subsets of even cardinality of a set of even cardinality. Note that, when one considers either the subsets of odd cardinality, or a carrier of odd cardinality, the resulting construction fails to be a logic.

Suppose that the carrier Ω has an odd number of elements. Then considering subsets of even cardinality would contradict the second axiom of Definition 4.2.5. Indeed the resulting poset would not be complemented. In fact, suppose that Ω has odd cardinality. In order for (Ω, Δ) to be complemented, Δ could not consist of the subsets of Ω of even cardinality, but then the disjoint union of two elements would not be part of Δ, thus contradicting the third axiom of Definition 4.2.5.

Continuing with the pattern of the two last examples, it is natural to consider the following example:

Example 5.1.4. *Consider* $\Omega = \{1, 2, 3, 4\}$*, and* $\Delta = \{S \subseteq \Omega \mid |S| = 2\} \cup \{\emptyset, \Omega\}$*, the collection of subsets of even cardinality in* Ω*. This logic is trivially rich, and regular. Each element is only compatible with its complement and the trivial elements. Hence, states are obtained by selecting one element of each pair of complements, and* Ω*. Since there are three such pairs, there must be* 2^3 *states.*

Consider a region $r \in \mathcal{R}(\mathrm{A}(\Delta))$*, and suppose it is not the extension of an element of* Δ*. Then for each* $x \in \Delta$ *there is either a state in* r *which does not contain* x*, or a state containing* x *which is not in* r*. Let* $x \in \Delta$ *and suppose* $\exists s_1 \in \mathcal{S}_{x'} \cap r$*. Then, out of Theorem 4.3.1,* $s_2 = \uparrow((s \cap \mathcal{A}(\Delta) \setminus x') \cup \{x\})$ *is a state, and* $[s_1, s_2] = \langle\{x'\}, \{x\}\rangle$*. This event has four instances on the*

transition system, one for each selection of atoms among the two remaining pairs. Then r must contain the four states that enable it, which coincide with the extension of x'. Furthermore, none of remaining states can belong to r, since they backward enable $[s_1, s_2]$. Then r must coincide with the extension of x'.

Now suppose there is a state $s_1 \in \mathcal{S}_x$ which is not in r. Then $s_1 \in \mathcal{S}_x \cap r'$, and the previous argument applies, so that r' is the extension of x. Then clearly, $r = \mathcal{S}_{x'}$.

Hence, $\Delta = \mathcal{R}(\mathrm{A}(\Delta))$.

This last example shows that the family of concrete logics built as the subsets of even cardinality of a set of even cardinality does not have a consistent behaviour regarding the synthesis of transition systems. The logic of the last example is regional, however, it is not representable as the regions of a transition system which has Ω as set of states.

5.1.2 A Characterisation Problem

The following problems arise naturally.

1. Given a logic L, build the condition/event transition system $\mathrm{A}(L)$, and consider its regional logic $\mathcal{R}(\mathrm{A}(L))$. Is $\mathcal{R}(\mathrm{A}(L))$ isomorphic to L?.

2. Given a condition/event transition system A_0, construct the saturated transition system associated with its regional logic $\mathrm{A}(\mathcal{R}(A_0))$. Is $\mathrm{A}(\mathcal{R}(A_0))$ isomorphic to A_0?

The second question is shown to have a negative answer in the general case. The following example serves as the proof.

Example 5.1.5. *Consider the condition/event transition system $A = (Q, E, T)$ shown in Figure 5.2. Its regions are the trivial ones, $\emptyset$ and Q, plus $x = \{1, 2, 5\}$, $y = \{1, 2, 6\}$, $z = \{1, 3, 5\}$, $w = \{1, 3, 6\}$ and the respective complements $\{3, 4, 6\} = (\{1, 2, 5\})'$, $\{3, 4, 5\} = (\{1, 2, 6\})'$, $\{2, 4, 6\} = (\{1, 3, 5\})'$ and $\{2, 4, 5\} = (\{1, 3, 6\})'$. The corresponding concrete logic is represented in Figure 5.2 and its states are formed by choosing exactly one element from each complementary pair of disjoint non-trivial regions. Hence, there are sixteen states. When constructing the saturated transition system from the logic $\mathcal{R}(A) = (\mathcal{R}(A), \subseteq, (.)', \emptyset, Q)$, that six out of the sixteen states in $\mathcal{S}(\mathcal{R}(A))$ correspond to the original states of A. However, for each state $q \in Q$, $\mathcal{R}_q$ is*

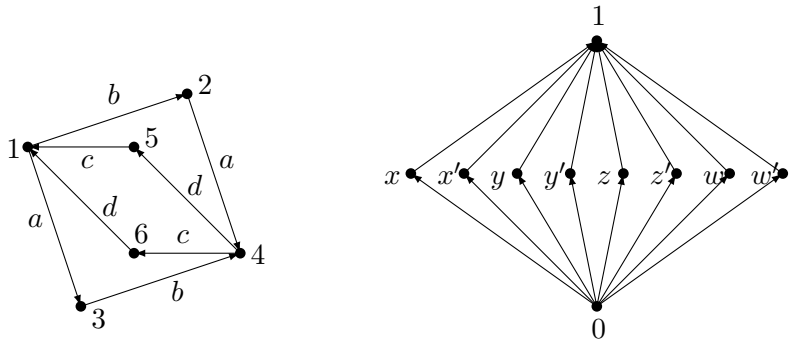

Figure 5.2: A condition/event transition system and its regional logic.

a state of $\mathcal{R}(A)$*, and labelling of transitions is preserved on the synthesised system. For example,* $\langle \mathcal{R}_1 \setminus \mathcal{R}_2, \mathcal{R}_2 \setminus \mathcal{R}_1 \rangle = \langle \mathcal{R}_3 \setminus \mathcal{R}_4, \mathcal{R}_4 \setminus \mathcal{R}_3 \rangle$ *is the image of* b *by the embedding* $A \hookrightarrow \mathrm{A}(\mathcal{R}(A))$*.*

Hence, in the general case a condition/event transition system is not isomorphic to the saturated transition system built from its regional logic. However, it has been shown to be a subsystem of it. Indeed, given a condition/event transition system $A = (Q, E, T)$, consider the mapping

$$\begin{aligned} \psi : Q &\to \mathcal{S}(\mathcal{R}(A)) \\ q &\mapsto \mathcal{R}_q \end{aligned}$$

Proposition 4.2.6 shows that ψ maps states to states, and is well defined. Furthermore, it follows trivially from Axiom 1 of Definition 2.2.8 that it is injective. Now consider two transitions carrying the same label, $(q_1, e, q_2), (q_3, e, q_4) \in T$. Then Proposition 4.2.7 shows that $[\psi(q_1), \psi(q_2)] = [\mathcal{R}_{q_1}, \mathcal{R}_{q_2}] = [\mathcal{R}_{q_3}, \mathcal{R}_{q_4}] = [\psi(q_3), \psi(q_4)]$. Hence, labels are preserved by ψ. To show that ψ can be extended to a morphism of transition systems remains out of the scope of this work. The following result can be found in [7].

Theorem 5.1.1. *Let* A *be a condition/event transition system. Then* A *embeds into* $\mathrm{A}(\mathcal{R}(A))$*.*

In order for this embedding to be an isomorphism, A should be itself a saturated transition system. Problem 2 can be restated by requiring that A was built from a logic.

3. Given a logic L, is $\mathrm{A}(\mathcal{R}(\mathrm{A}(L)))$ isomorphic to $\mathrm{A}(L)$?

Naturally, this new problem reduces to Problem 1.

Conversely, the logic presented in the last section, in Example 5.1.3 is a rich and regular logic which is not stable, and so Problem 1 should as well be restated

> 4. Given a condition/event transition system A, is $\mathcal{R}(\mathrm{A}(\mathcal{R}(A)))$ isomorphic to $\mathcal{R}(A)$?

Clearly, the Problems 3 and 4 reduce to each other, if one further requires that the logic L in Problem 3 is regional.

Nevertheless, given a rich logic L, it can be shown to embed in $\mathcal{R}(\mathrm{A}(L))$. The mapping

$$\phi : L \to \mathcal{R}(\mathrm{A}(L)) \tag{5.1}$$

$$x \mapsto \mathcal{S}_x \tag{5.2}$$

was shown to be an injective morphism in [7]. The proof is here reported informally, and it is further proved that ϕ is an embedding.

It is first shown that $\mathcal{S}_x$ is a region of the saturated transition system. This is so when all transitions carrying the same label have a consistent orientation with respect to $\mathcal{S}_x$. Let $(s_1, [s_1, s_2], s_2), (s_3, [s_3, s_4], s_4) \in T(L)$ with $[s_1, s_2] = [s_3, s_4]$. Suppose $s_1 \in \mathcal{S}_x$ and $s_2 \notin \mathcal{S}_x$. Then $x \in s_1 \setminus s_2 = s_3 \setminus s_4$, and so $s_3 \in \mathcal{S}_x$, but $s_4 \notin \mathcal{S}_x$. Analogously one gets that if $s_1 \notin \mathcal{S}_x$ and $s_2 \in \mathcal{S}_x$, then $s_3 \notin \mathcal{S}_x$, and $s_4 \in \mathcal{S}_x$. Finally, if either $s_1 \in \mathcal{S}_x$ and $s_2 \in \mathcal{S}_x$, or $s_1 \notin \mathcal{S}_x$ and $s_2 \notin \mathcal{S}_x$, then it holds that $x \notin s_1 \setminus s_2 = s_3 \setminus s_4$, and $x \notin s_2 \setminus s_1 = s_4 \setminus s_3$. In any case, it holds that $x \in s_3 \to x \in s_4$ and $x \in s_4 \to x \in s_3$. Then $s_3 \in \mathcal{S}_x \leftrightarrow s_4 \in \mathcal{S}_x$. So $\mathcal{S}_x$ must be a region.

To see that ϕ is a morphism of logics, one must verify the axioms of Definition 2.3.8 (p.45). The first axiom is trivial since no state contains 0. This is clear from the first axiom in Definition 4.2.6 of state: $s(1) = 1$. Then every state satisfies that $s(1') = s(0) = 0$, and so $\mathcal{S}_0 = \emptyset$. The second axiom to verify is $\mathcal{S}_{x'} = \mathcal{S}(L) \setminus \mathcal{S}_x$. Note that $x \perp x'$, and from the Definition 4.2.6 of state, $\forall s \in \mathcal{S}(L) : s(x) + s(x') = s(x \vee x') = s(1) = 1$. Then clearly, either $x \in s$, or $x' \in s$, but never both. Finally, the last axiom requires that, given a collection of pairwise orthogonal elements, then the join of their images is the image of their joins. To see this, first note that if $x \perp y$, then $\mathcal{S}_x \cap \mathcal{S}_y = \emptyset$. Since states are upwards closed in L, it holds that any states containing x must contain all the elements above it. Analogously $\forall s \in \mathcal{S}_y : \forall z \in \uparrow\{y\} : s(z) = 1$. Then the least upper bound of x and y is the smallest element $x \vee y$ such that $s(x) = 1$ or $s(y) = 1$ implies $s(x \vee y) = 1$, and so $\mathcal{S}_x \cup \mathcal{S}_y \subseteq \mathcal{S}_{x \vee y}$.

Furthermore, let $s(x \vee y) = s(x) + s(y)$. It implies that either $s(x) = 1$ or $s(y) = 1$, so $\mathcal{S}_{x \vee y} \subseteq \mathcal{S}_x \cup \mathcal{S}_y$.

The fact that ϕ is injective follows directly from the Definition 4.2.7 of rich logic. Indeed, if $\mathcal{S}_x = \mathcal{S}_y$, then $\mathcal{S}_x \subseteq \mathcal{S}_y$, and $\mathcal{S}_x \subseteq \mathcal{S}_y$. Since L is rich, $x \leq y$ and $y \leq x$ so $x = y$.

A consequence of this result is that ϕ is an injective morphism. The following proposition shows that its inverse, restricted to its image, is also a morphism. As a consequence, ϕ is an embedding.

Proposition 5.1.1. *Let L be a rich and regular logic, and $\phi : L \to \mathcal{R}(\mathrm{A}(L))$ be defined by $\phi(x) = \mathcal{S}_x$ for every $x \in L$. Then $\tilde{\psi} \equiv \phi^{-1} \restriction_{\phi(L)}$ is a logic morphism.*

Proof. Clearly, $\forall x \in L : \tilde{\psi}(\mathcal{S}_x) = x$. Now, since L is rich, the only element whose associated set of states is the bottom element, hence $\tilde{\psi}(\emptyset) = 0$. Also, $\tilde{\psi}(\mathcal{S}(L) \setminus \mathcal{S}_x) = \tilde{\psi}(\mathcal{S}_{x'}) = x'$, so it preserves orthocomplements. Finally, consider two disjoint $\mathcal{S}_x, \mathcal{S}_y$. Then $\mathcal{S}_x \subseteq \mathcal{S}(L) \setminus \mathcal{S}_y = \mathcal{S}_{y'}$, and since L is rich, $x \perp y$. From point 2. in Definition 4.2.6 it stands that $\{s \in \mathcal{S}(L) \mid s(x \vee y) = 1\} = \{s \in \mathcal{S}(L) \mid s(x) = 1\} \cup \{s \in \mathcal{S}(L) \mid s(y) = 1\} = \mathcal{S}_x \cup \mathcal{S}_y$. Hence $\tilde{\psi}(\mathcal{S}_x \cup \mathcal{S}_y) = x \vee y$. □

Theorem 5.1.2. *Let L be a logic, then*

$$\begin{aligned} \phi : L &\to \mathcal{R}(\mathrm{A}(L)) \\ x &\mapsto \mathcal{S}_x \end{aligned}$$

is an embedding of logics.

The missing part of the proof that ϕ is an isomorphism is its surjectiveness. Note that, since the logic of Example 5.1.3 is not isomorphic to the set of regions of its saturated transition system. Additional conditions are certainly to be required on the logic. This chapter explores the properties of regional logics, in search for a condition that would allow to solve Problem 4.

As a technical remark, the following lemma gives a sufficient condition for the morphism ϕ to be surjective. It requires of a logic to be finite, which is reasonable when considering the regions of finite transition systems.

Lemma 5.1.1. *Let L be a rich and finite logic, and ϕ as in Equation* (5.1)*. Suppose that $\forall r \in \mathcal{R}(\mathrm{A}(L)) \setminus \{\emptyset\} : \exists x \in L \setminus \{0\} : \phi(x) \subseteq r$. Then ϕ is surjective.*

Proof. The proof follows an induction. Let $r = r_0 \in \mathcal{R}(\mathrm{A}(L))$. There is an $x_1 \in L \setminus \{0\} : \phi(x_1) \subseteq r_0$. Since ϕ is rich, it maps regions to regions, and out of orthomodularity of L , the relative complement of $\phi(x_1)$ in r_0 is a region, put $r_1 = r_0 \setminus \phi(x_1) \in \mathcal{R}(\mathrm{A}(L))$. Now suppose $r_i = r \setminus (\bigcup_{j \leq i} \phi(x_j)) \in \mathcal{R}(\mathrm{A}(L))$, then $\exists x_{i+1} \in L \setminus \{0\} : \phi(x_{i+1}) \subseteq r_i$. Again, $r_{i+1} = r_i \setminus \phi(x_{i+1}) \in \mathcal{R}(\mathrm{A}(L))$, and furthermore $r_{i+1} = r \setminus (\bigcup_{j \leq i+1} \phi(x_j))$. Now $\forall i \in \mathbb{N} : x_i \neq 0$, and ϕ being an injective morphism of logics, it must hold that $\phi(x_i) \neq \emptyset$. Hence $\forall i \in \mathbb{N} : r_{i+1} \subsetneq r_i$. Then, out of finiteness of L, $\mathcal{S}(L)$ must also be finite. So there must be an $n \in \mathbb{N}$ such that $r_n = \emptyset$, and $r_{n-1} = \phi(x_n)$. Note that $\forall i \leq n : \phi(x_i) \subseteq r = r_0$, and so $r_n = r \setminus (\bigcup_{j \leq n} \phi(x_j)) = \emptyset$ implies that $r = \bigcup_{j \leq n} \phi(x_j)$

Finally, note that by construction, $\forall i \leq n : \forall j < i : \phi(x_i) \cap \phi(x_j) = \emptyset$. Thus, $\{\phi(x_i)\}_{i \leq n}$ is a disjoint collection of regions of $\mathcal{R}(\mathrm{A}(L))$. Since ${\phi^{-1}}\restriction_{\phi(L)}$ is a morphism, out of Proposition 2.3.1,it preserves orthogonality. It follows that $\{x_i\}_{i \leq n}$ is collection of pairwise orthogonal elements of L, and so $\bigvee_{i \leq n} x_i \in L$. Since ϕ is a morphism $\phi(\bigvee_{i \leq n} x_i) = \bigcup_{i \leq n} \phi(x_i) = r$. □

With this result, surjectiveness of ϕ would be proven by showing that ϕ maps some non-trivial element inside every non-empty region.

5.2 ETI and Strong embedding

This section investigates properties of regional logics, which are not common to all logics. Richness and regularity exclude counterexamples to the stability conjecture. It is shown that in [11] that regional logics satisfy the properties called TIP. This section introduces a new property verified by all regional logics, which is necessary for a logic to be stable. It can be used to determine subsets of events which are sufficient for the synthesised transition system built on a given logic to present the same regions as the saturated one. Finally, this property is used to show that the embedding of a logic in the poset of regions of its saturated transition system is strong.

5.2.1 Properties of Regional Logics

The results of this section can be found in [14], where the property ETI was introduced.

The property TIP results from the translation in the abstract setting of quantum logic of a property of regions of transition systems, named *triple intersection property* and which is expressed by the following lemma, the proof of which can be found in [11].

Lemma 5.2.1. *Let A be a condition/event transition system, and let $a, b, c \in \mathcal{R}(A)$ be such that: $a \cap b = b \cap c = c \cap a$, then $z = a \cap b \cap c \in \mathcal{R}(A)$.*

The triple intersection property (TIP) for a logic L is then obtained by considering for any element in L the set of two-valued states selecting that element, and from the fact that this set identifies a region of the condition/event transition system $\mathrm{A}(L)$ synthesised from L.

Definition 5.2.1 (Triple Intersection Property)**.** *A logic L is* TIP *if*

$$\forall a, b, c \in L : \mathcal{S}_a \cap \mathcal{S}_b = \mathcal{S}_b \cap \mathcal{S}_c = \mathcal{S}_c \cap \mathcal{S}_a \quad \Rightarrow \quad \exists z \in L : \mathcal{S}_z = \mathcal{S}_a \cap \mathcal{S}_b \cap \mathcal{S}_c$$

A new property is investigated, called ETI. As well as TIP, also ETI is inspired by a property of regions. If q_1, q_2, q_3, and q_4 are distinct states, so that (q_1, e, q_2), and (q_3, e, q_4) are two different transitions in a condition/event transition system $A = (Q, E, T)$, then $\mathcal{R}(A)$ must contain two incompatible regions, one containing q_1 and q_2 but not q_3, the other containing q_1 and q_3 but not q_2. Hence, events with multiple occurrences, and pairs of incompatible regions are related. In this sense, the events of an abstract logic L are said to *testify incompatibility*; L is said to be ETI if, for any pair of incompatible regions, the set $E(L)$ contains an element witnessing this incompatibility.

Definition 5.2.2 (Events Testify Incompatibility)**.** *A logic L is* ETI *if* $\forall a, b \in L : a \not\$ b \Rightarrow$

$$\exists s_1 \in \mathcal{S}_a \cap \mathcal{S}_b, s_a \in \mathcal{S}_a \cap \mathcal{S}_{b'}, s_b \in \mathcal{S}_b \cap \mathcal{S}_{a'}, s_0 \in \mathcal{S}_{a'} \cap \mathcal{S}_{b'} : \; s_a \setminus s_1 = s_0 \setminus s_b$$

When L is ETI, it is ensured that concurrency among any two regions is described by incompatibility.

Example 5.2.1. *An example of a concrete logic which is neither* TIP *nor* ETI *is the logic of the subsets of even cardinality of $\{1, 2, 3, 4, 5, 6\}$, already seen in Example 5.1.3 (p.127). This logic is regular and rich, but not regional as discussed in [7]. In order to see that L is not* TIP*, consider the elements in L : $\{1, 2\}, \{1, 3\}, \{1, 4\}$, then $\mathcal{S}_{\{1,2\}} \cap \mathcal{S}_{\{1,3\}} = \mathcal{S}_{\{1,3\}} \cap \mathcal{S}_{\{1,4\}} = \mathcal{S}_{\{1,4\}} \cap \mathcal{S}_{\{1,2\}} = \{\delta_1\}$, where δ_1 is the two-valued state of L selecting all the elements containing 1. It is then immediate to see that there is no z in L such that $\mathcal{S}_z = \{\delta_1\}$. L is not* ETI *since the symmetric differences among the six states of the associated transition systems are all different, and then it is not possible for a pair of incompatible elements of L to find pairs of equal symmetric differences.*

Any regional logic is TIP, as shown in [11]. Although it is not yet known if the two properties, ETI and TIP, coincide, it can be proven that ETI implies TIP.

Theorem 5.2.1. *Let L be a* ETI *logic. Then L is* TIP.

Proof. By contradiction, let L be ETI and not TIP. Not TIP means: $\exists a, b, c \in L:\ \mathcal{S}_a \cap \mathcal{S}_b = \mathcal{S}_b \cap \mathcal{S}_c = \mathcal{S}_c \cap \mathcal{S}_a \neq \emptyset$ and $\forall z \in L:\ \mathcal{S}_z \neq \mathcal{S}_a \cap \mathcal{S}_b \cap \mathcal{S}_c$. There are two cases.

First case: $a \,\$\, b$. Then since $\mathcal{S}_a \cap \mathcal{S}_b \neq \emptyset$, $a \not\perp b$ and then there exist three mutually orthogonal elements $\hat{a}$, $\hat{b}$ and x in L such that $a = \hat{a} \vee x$ and $b = \hat{b} \vee x$. This implies $\mathcal{S}_a$ to be the disjoint union of $\mathcal{S}_{\hat{a}}$ and $\mathcal{S}_x$, and $\mathcal{S}_b$ to be the disjoint union of $\mathcal{S}_{\hat{b}}$ and $\mathcal{S}_x$, yielding the contradiction: $\exists x : \mathcal{S}_x = \mathcal{S}_a \cap \mathcal{S}_b = \mathcal{S}_c \cap \mathcal{S}_a$.

Second case: $a \not\$ b$. Then, since L is ETI, there are four states: $s_1 \in \mathcal{S}_a \cap \mathcal{S}_b$, $s_a \in \mathcal{S}_a \cap \mathcal{S}_{b'}, s_b \in \mathcal{S}_b \cap \mathcal{S}_{a'}$, $s_0 \in \mathcal{S}_{a'} \cap \mathcal{S}_{b'}$ such that: $s_a \setminus s_1 = s_0 \setminus s_b$. Then $s_1 \in \mathcal{S}_a \cap \mathcal{S}_b$ implies $s_1 \in \mathcal{S}_c$. Since $\mathcal{S}_c$ is a region in $\mathcal{R}(\mathrm{A}(L))$, then either $s_a \in \mathcal{S}_c$, or $s_b \in \mathcal{S}_c$. In any case this contradicts the hypothesis $\mathcal{S}_a \cap \mathcal{S}_b = \mathcal{S}_b \cap \mathcal{S}_c = \mathcal{S}_c \cap \mathcal{S}_a$. □

It will now be shown, that all regional logics are ETI.

Theorem 5.2.2. *Let $A = (Q, E, T)$ be a generalised condition/event transition system. Then $\mathcal{R}(A)$ is* ETI.

Proof. Let $a, b \in \mathcal{R}(A)$ be such that $a \not\$ b$. Since $a \cap b = \emptyset$ would imply $a \perp b$, and $a \cap b \in \mathcal{R}(A)$ would mean $a \,\$\, b$, there must be an event $e \in E$, and two transitions $(s_1, e, s_2), (s_3, e, s_4) \in T$ such that one of them crosses the border of $a \cap b$, and the other does not. Suppose, without loss of generality, that $s_1 \in a \cap b$, and $s_2 \notin a \cap b$, then clearly either $s_2 \in a \setminus b$ or $s_2 \in b \setminus a$. After symmetry of $\not\$$, the two must be equivalent, so assume the first case holds. Clearly, (s_1, e, s_2) crosses the border of b, which is a region, hence it must be that $s_3 \in b$ and $s_4 \notin b$. (s_3, e, s_4) does not cross the border of $a \cap b$, so clearly $s_3 \in b \setminus a$, but neither does (s_3, e, s_4) cross the border of a, hence $s_4 \notin a \cup b$. Then $s_1 \in a \cap b, s_2 \in a \setminus b, s_3 \in b \setminus a$ and $s_4 \notin a \cup b$, furthermore, $e \in E$ implies that $s_2 \setminus s_1 = s_4 \setminus s_3$. □

5.2.2 $\mathcal{D}$-completeness, and ETI-completeness

The results of this section can be found in [13]

This section explores how the property ETI can be used in order to reduce the set of events considered in the synthesis of a transition system from a

logic. In order to do so, $\mathcal{D}(L)$ is defined as the set of 4-tuples of states of a logic L which identify concurrency between potential events.

Definition 5.2.3. *Let L be a logic.*
$\mathcal{D}(L) = \{(s_1, s_2, s_3, s_4) \mid s_i \in \mathcal{S}(L)$ *for* $i = 1, \cdots, 4, s_1 \neq s_2, s_1 \neq s_3, s_1 \setminus s_2 = s_3 \setminus s_4\}$.

As stated in the following Lemma, any 4-tuple in $\mathcal{D}(L)$ identifies three other 4-tuples in $\mathcal{D}(L)$ which can be considered equivalent regarding incompatibility of regions.

Lemma 5.2.2. *If* $(s_1, s_2, s_3, s_4) \in \mathcal{D}(L)$,
then $\{(s_2, s_1, s_4, s_3), (s_1, s_3, s_2, s_4), (s_3, s_1, s_4, s_2)\} \subseteq \mathcal{D}(L)$,
and $\{(s_3, s_4, s_1, s_2), (s_4, s_3, s_2, s_1), (s_2, s_4, s_1, s_3), (s_4, s_2, s_3, s_1)\} \subseteq \mathcal{D}(L)$.

Proof. It is first proved that $s_1 \setminus s_2 = s_3 \setminus s_4$ and $s_1 \neq s_2$ imply $s_2 \setminus s_1 = s_4 \setminus s_3$, and $s_1 \setminus s_3 = s_2 \setminus s_4$, and $s_3 \setminus s_1 = s_4 \setminus s_2$. Any two-valued state s satisfies $s(a) = 1 \Leftrightarrow s(a') = 0$. Hence $s_1 \setminus s_2 = s_3 \setminus s_4$ implies $s_2 \setminus s_1 = s_4 \setminus s_3$.

It will now be proved that $s_1 \setminus s_2 = s_3 \setminus s_4$ and $s_1 \neq s_2$ imply $s_1 \setminus s_3 = s_2 \setminus s_4$. It will be shown that $s_1 \setminus s_3 \subseteq s_2 \setminus s_4$, the proof of the containment in the other direction being analogous. From $x \in s_1 \setminus s_3$ and $s_1 \setminus s_2 = s_3 \setminus s_4$, it follows: $x \notin s_3$, $x \notin s_1 \setminus s_2$, and then $x \in s_2 \cap s_1$ and from $s_2 \setminus s_1 = s_4 \setminus s_3$ also $x \notin s_4$, i.e. : $x \in s_2 \setminus s_4$. Finally, by symmetry of the equality in $s_1 \setminus s_2 = s_3 \setminus s_4$, one obtains $(s_1, s_2, s_3, s_4) \in \mathcal{D}(L) \Leftrightarrow (s_3, s_4, s_1, s_2) \in \mathcal{D}(L)$, which by analogy, completes the proof for the four remaining cases. □

The set $\mathcal{D}(L)$ contains all the configurations of the saturated net system which translate incompatibility of L into concurrency of events. As such, they allow for preservation of this incompatibility.

Consider the synthesis of a generalised condition/event transition system $A_0 = (\mathcal{S}(L), E_0, T_0)$, in which, although the sets of states of $A(L)$ and A_0 coincide, the set of events, as symmetric differences, is restricted: $E_0 \subsetneq E$. It is still required that all transitions in $T(L)$ carrying a label in E_0 to be present in $T_0 \subsetneq T(L)$. This notion is formalised in the following definition.

Definition 5.2.4. *Let $A = (Q, E, T)$, and $A_0 = (Q_0, E_0, T_0)$ be two generalised condition/event transition system. Let $A_0 \prec A$ denote that: $Q = Q_0$, $E_0 \subseteq E$, $T_0 \subseteq T$ and $\forall e \in E_0 : (s, e, t) \in T \Rightarrow (s, e, t) \in T_0$.*

Under these assumptions, a sufficient condition on E_0 can be presented for A_0 to carry the same information as $A(L)$ regarding concurrency.

However, according to the definition of region, an event e might prevent a subset of states from being a region. So withdrawing an event from E might allow for more regions, but never exclude an existing region. This idea is formalised in the following lemma.

Lemma 5.2.3. *Let L be a logic, and $A := A(L) = (\mathcal{S}(L), E(L), T(L))$ its saturated transition system. Let $A_0 \prec A$. Then $\mathcal{R}(A) \subseteq \mathcal{R}(A_0)$.*

Proof. Let $r \in \mathcal{R}(A)$. Then each $e \in E(L)$ crosses r uniformly. Since $E_0 \subseteq E(L)$, the same holds for each event in E_0. Hence r is a region in A_0. □

The property the set of events E_0 has to fulfil for $\mathcal{R}(A) = \mathcal{R}(A_0)$ to hold is as follows. A set of events E_0 is called *$\mathcal{D}$-complete* if for any set of equivalent 4-tuples in $\mathcal{D}(L)$, there is at least a label in E_0 corresponding to one of the symmetric differences associated with those tuples.

Definition 5.2.5 ($\mathcal{D}$-complete System)**.** *Let $E(L) = \{[s_1, s_2] \mid s_1, s_2 \in \mathcal{S}(L), s_1 \neq s_2\}$. A set $E_0 \subseteq E(L)$ is* $\mathcal{D}$-complete *if for any $(s_1, s_2, s_3, s_4) \in \mathcal{D}(L)$, it holds:*

$$[s_1, s_2] \in E_0 \ \vee [s_2, s_1] \in E_0 \ \vee [s_1, s_3] \in E_0 \vee [s_3, s_1] \in E_0.$$

Theorem 5.2.3. *Let L be a logic, and $A := A(L) = (\mathcal{S}(L), E(L), T(L))$ its synthesised saturated transition system. Let $A_0 = (\mathcal{S}(L), E_0, T_0)$ be a generalised condition/event transition system such that $A_0 \prec A$. If E_0 is $\mathcal{D}$-complete, then $\mathcal{R}(A) = \mathcal{R}(A_0)$.*

Proof. Lemma 5.2.3 already provided $\mathcal{R}(A) \subseteq \mathcal{R}(A_0)$. It now suffices to prove $\mathcal{R}(A_0) \subseteq \mathcal{R}(A)$.

Let r be a region in A_0, and suppose that it is not a region in A. Then there is an event $e \in E(L)$ which does not cross the border of r uniformly in A. Without losing in generality, assume that there $(1, e, 2), (3, e, 4) \in T(L)$, with $1, 2, 3 \in r$ and $4 \notin r$. Then the four states $1, 2, 3$, and 4 form a diamond. Since E_0 is $\mathcal{D}$-complete, it contains e_0 corresponding to a pair of parallel edges in $(1, 2, 3, 4)$. So either $\{(1, e_0, 2), (3, e_0, 4)\}, \{(2, e_0, 1), (4, e_0, 3)\}$, $\{(1, e_0, 3), (2, e_0, 4)\}$, or $\{(3, e_0, 1), (4, e_0, 2)\} \subseteq T_0$. In any of these cases, one of the transitions crosses the border of r while the other does not, thus r can neither be a region of A_0. Note that, for any of the possible configurations in which e prevents r from being a region in A, the four possibilities for e_0 are the same, and any of them still prevents r from being a region in A_0. □

A set of events $E_0 \subseteq E(L)$ is called ETI-*complete* if for any pair a and b of incompatible elements of L there is a label in E_0, with at least two occurrences

in T_0, such that either it crosses a and is parallel to b, or it crosses b and is parallel to a.

Definition 5.2.6 (ETI-complete System)**.** *$E_0 \subseteq E$ is* ETI-complete *if* $\forall a, b \in L : (a \not\parallel b) \Rightarrow$
$(\exists [s_i, s_j] \in E_0, \exists\ s_1 \in \mathcal{S}_a \cap \mathcal{S}_b, s_a \in \mathcal{S}_a \cap \mathcal{S}_{b'}, s_b \in \mathcal{S}_b \cap \mathcal{S}_{a'}, s_0 \in \mathcal{S}_{a'} \cap \mathcal{S}_{b'} :$
$(s_a, s_1, s_0, s_b) \in \mathcal{D}(L)$ *and* $[s_i, s_j] = [s_a, s_1] \ \vee [s_i, s_j] = [s_1, s_a] \ \vee [s_i, s_j] = [s_b, s_1] \ \vee [s_i, s_j] = [s_1, s_b]$).

Clearly, if L is ETI and $a \not\parallel b$, the existence of a 4-tuple of states in $\mathcal{D}(L)$ with equal symmetric differences is provided. For ETI-completeness, only one corresponding event is required in E_0.

If L is not ETI, then there exists no E_0 which is ETI-complete, and then, from the following proposition, there is no E_0 which is $\mathcal{D}$-complete.

Proposition 5.2.1. *Let L be a* ETI *logic. Let $A := A(L) = (\mathcal{S}(L), E(L), T(L))$ be the corresponding saturated CE transition system, and $A_0 \prec A$ be such that E_0 is $\mathcal{D}$-complete. Then E_0 is* ETI*-complete.*

Proof. Let $a, b \in L : a \not\parallel b$, since L is ETI there must be $s_1 \in a \cap b, s_2 \in a \setminus b, s_3 \in b \setminus a$ and $s_4 \notin a \cup b$ such that $s_2 \setminus s_1 = s_4 \setminus s_3$. Clearly, $(s_1, s_2, s_3, s_4) \in \mathcal{D}(L)$, and since E_0 is $\mathcal{D}$-complete, there must be an event $e_0 \in E_0$, and a couple of transitions in T_0 among the following: $\{(s_1, e_0, s_2), (s_3, e_0, s_4)\}$, $\{(s_2, e_0, s_1), (s_4, e_0, s_3)\}, \{(s_1, e_0, s_3), (s_2, e_0, s_4)\}$, or $\{(s_3, e_0, s_1),\ (s_4, e_0, s_2)\}$. Now, $E_0 \subseteq E(L)$, so in particular $e_0 \in E(L)$ implying, by Lemma 5.2.2, that either $s_2 \setminus s_1 = s_4 \setminus s_3$ and $s_1 \setminus s_2 = s_3 \setminus s_4$, or $s_3 \setminus s_1 = s_4 \setminus s_2$ and $s_1 \setminus s_3 = s_2 \setminus s_4$. □

This exercise in restricting the set of events considered in the synthesis, in a manner analogous to the one in Chapter 4, shows that the ETI property should be sufficient to convey the incompatiblity to the regions of its synthesised transition system, through concurrency of the events of the latter.

5.2.3 Strong Embedding

The results of this section can be found in [14]

The setting of Section 5.1.2, is recalled. Let L be a rich logic, and consider

$$\phi : L \to \mathcal{R}(\mathrm{A}(L))$$
$$x \mapsto \mathcal{S}_x$$

then ϕ is an embedding of logics.

Now, ϕ being an embedding means that L and $\mathcal{R}(\mathrm{A}(L))$ would be isomorphic if ϕ was surjective. It has also been seen in the previous section that L being ETI is a necessary condition for that. It shall now be seen that if L is ETI, the embedding satisfies a stronger property, required (but not sufficient) for ϕ to be an isomorphism.

It is well known that logic morphisms preserve order, orthogonality and compatibility [51]. Since ϕ is an embedding it shall also reflect these relations. However, in general, this is only true when considering them restricted to the image $\phi(L)$. Indeed, if ϕ is an embedding then L is isomorphic to $\phi(L)$, but the lack of surjectiveness might, in the general case, allow the images of two incompatible elements to be compatible. The next example shall explain this notion.

Example 5.2.2. *Consider the logic $L = \{0, u, u', v, v', 1\}$, and the Boolean logic B whose atoms are $\{a_1, a_2, a_3, a_4\}$ (Figure 5.3). Then the mapping given by $\phi(u) = a_1 \vee a_2$, $\phi(u') = a_3 \vee a_4$, $\phi(v) = a_1 \vee a_3$, $\phi(v') = a_2 \vee a_4$ is indeed an embedding. Since $(a_1 \vee a_2)' = a_3 \vee a_4$ and $(a_1 \vee a_3)' = a_2 \vee a_4$, the sublogic $L_1 = \{0, a_1 \vee a_2, a_3 \vee a_4, a_1 \vee a_3, a_2 \vee a_4, 1\}$ of B is isomorphic to L. However, when considered in the whole of B, one can see that a_1 is both $a_1 \leq a_1 \vee a_2$ and $a_1 \leq a_1 \vee a_3$, with $\{a_1, a_2, a_3\}$ mutually orthogonal in B. Thus $\phi(u) \$ \phi(v)$, whereas $u \not\$ v$. This does not prevent ϕ from being an embedding because, in fact $a_1, a_2, a_3 \notin L_1$.*

This example should justify the following definition (see [51]).

Definition 5.2.7 (Strong Embedding)**.** *Let $\phi : L_1 \to L_2$ be an embedding between logics. Then ϕ is said to be a* strong embedding *if*

$$\forall a, b \in L_1 : a \$ b \Leftrightarrow \phi(a) \$ \phi(b)$$

For instance, a logic with incompatible elements cannot embed strongly into a Boolean logic.

The next result shows that a logic L being ETI is a sufficient condition for $\phi : L \to \mathcal{R}(\mathrm{A}(L))$ to be a strong embedding.

Theorem 5.2.4. *Let L be a rich and regular logic. If L is* ETI*, then the embedding $\phi : L \to \mathcal{R}(\mathrm{A}(L))$ defined as $\phi(x) = \mathcal{S}_x$ is strong.*

Proof. Theorem 5.1.1 shows that ϕ preserves compatibility, it will therefore be sufficient to prove that it also preserves incompatibility. So let $a, b \in L$ satisfy $a \not\$ b$. Since L is ETI, $\exists s_1 \in \mathcal{S}_a \cap \mathcal{S}_b, s_a \in \mathcal{S}_a \cap \mathcal{S}_{b'}, s_b \in \mathcal{S}_b \cap \mathcal{S}_{a'}, s_0 \in \mathcal{S}_{a'} \cap \mathcal{S}_{b'} : s_a \setminus s_1 = s_0 \setminus s_b$. Then $e = [s_1, s_a] \in E(L)$ will be a label in the saturated

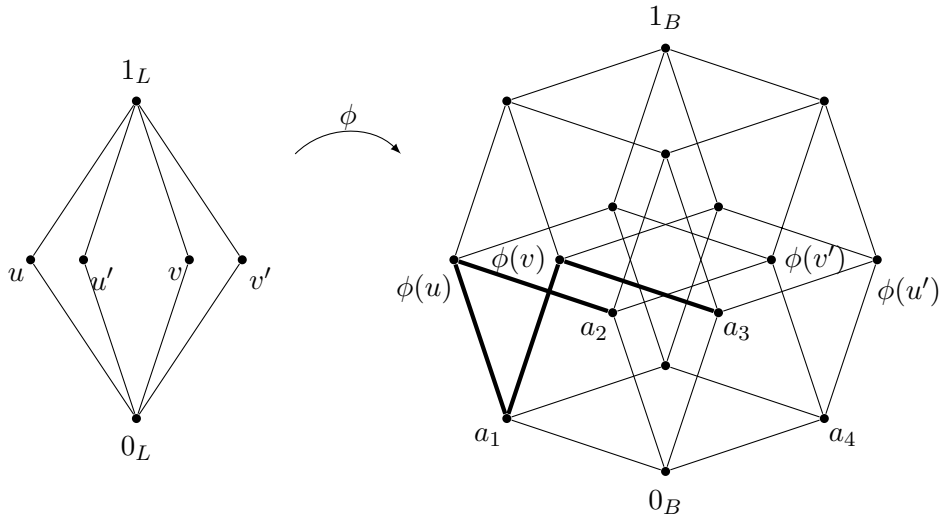

Figure 5.3: An example of embedding which is not strong.

transition system $\mathrm{A}(L)$, and the transitions $(s_1, e, s_a), (s_b, e, s_0) \in T(L)$ will prevent $\mathcal{S}_a \cap \mathcal{S}_b$ from being a region. Indeed, (s_1, e, s_a) crosses the border of $\mathcal{S}_a \cap \mathcal{S}_b$, whereas (s_b, e, s_0) does not. Since $\mathcal{R}(\mathrm{A}(L))$ is a concrete logic, it holds that $\mathcal{S}_a = \phi(a) \not\$ \phi(b) = \mathcal{S}_b$. □

This result implies, in particular, that if new regions are produced by the synthesis procedure, these cannot be contained in the image by ϕ of an atom. Orthocomplementation implies therefore that they cannot contain the images of coatoms (maximal elements except for the top). Thus, the possible lack of surjectiveness of ϕ is narrowed down.

5.3 Stable Classes of Logics

In this section a few subclasses of concrete logics will be studied to show that they are stable. The presented results can be found in [14].

5.3.1 Boolean Algebras

To start with the simplest example, let L be a finite Boolean logic with k atoms. Then L is a regular rich logic, isomorphic to the power set of $\{1, \cdots, k\}$, in which singletons correspond to atoms. L has exactly k states, each corresponding to $\uparrow\{x\}$, where x is an atom of L.

This implies that all the ordered symmetric differences between states differ in at least one atom, so that each transition in $\mathrm{A}(L)$ carries a unique label, and all subsets of states are regions. Hence, L and $\mathcal{R}(\mathrm{A}(L))$ are isomorphic, and L is stable.

5.3.2 $\{0,1\}$-pasting

The next case under consideration is that of a logic obtained as the so-called $\{0,1\}$-pasting of two (or more) logics. In plain words, the $\{0,\ 1\}$-pasting of L_1 and L_2 is the disjoint union of L_1 and L_2, but for identification of 0_1 with 0_2, and 1_1 with 1_2 (see Figure 5.5).

Definition 5.3.1 ($\{0\text{-}1\}$-Pasting of Logics)**.** *Let L_1 and L_2 be two disjoint logics with $0_i, 1_i$ for $i = 1, 2$ the respective least and greatest elements. Define on $L_1 \cup L_2$ the equivalence relation*

$$\sim := \{(a, a) \mid a \in L_1 \cup L_2\} \cup \{(0_1, 0_2), (1_1, 1_2)\}.$$

Then the $\{0\text{-}1\}$-pasting *of L_1 and L_2, denoted by $L_1 \,||\, L_2$ is given by the set $(L_1 \cup L_2)/\!\sim$, together with the partial order obtained as the union of the partial orders of L_1 and L_2, up to $\sim$.*

The $\{0\text{-}1\}$-pasting of L_1 and L_2 is again a logic ([51], Prop. 1.2.6). Without loss of generality, in what follows the notation concerning the $\{0\text{-}1\}$-pasting of L_1 and L_2 will be simplified by indicating by the same element 0 and, respectively, 1 the bottom and top elements in *both* L_1 and L_2.

Remark 5.3.1. *As a useful consequence, the set of states $\mathcal{S}(L_1 \,||\, L_2)$ of $L_1 \,||\, L_2$, consists in sets obtained by taking the union $s_1 \cup s_2$ for any pair of states $s_1 \in \mathcal{S}(L_1)$ and $s_2 \in \mathcal{S}(L_2)$.*

The $\{0,\ 1\}$-pasting of logics is related to a construction on transition systems. Given two transition systems, A_1 and A_2, with disjoint sets of events, a new one can be built by putting them side-by-side and letting them work in parallel. States of this new transition system are pairs of "local" states of the two components. The result will be called the *parallel product* of A_1 and A_2, and denote it by $A_1 \,||\, A_2$.

Definition 5.3.2 (Parallel Composition of Systems)**.** *Let $A_i = (Q_i, E_i, T_i)$ be a condition/event transition system for $i = 1, 2$, with $E_1 \cap E_2 = \emptyset$. Define*

$$A_1 \,||\, A_2 = (Q_1 \times Q_2, E_1 \cup E_2, T)$$

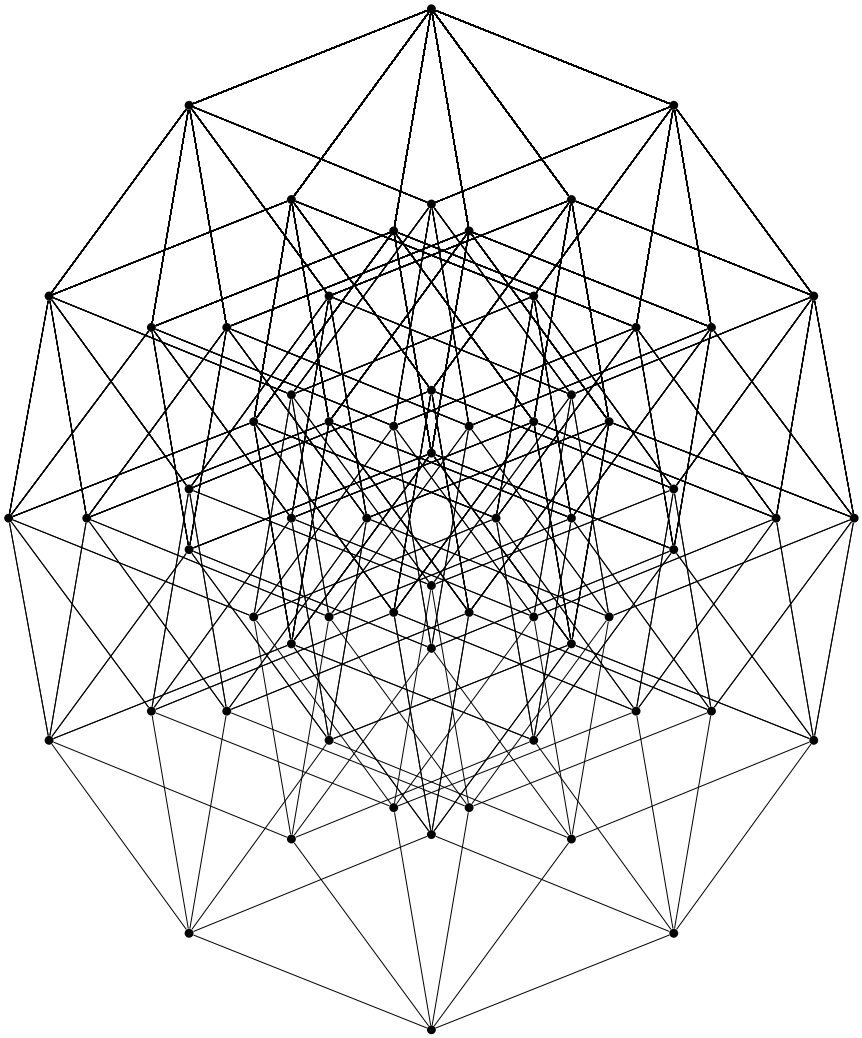

Figure 5.4: Hasse diagram of a Boolean algebra with 6 atoms.

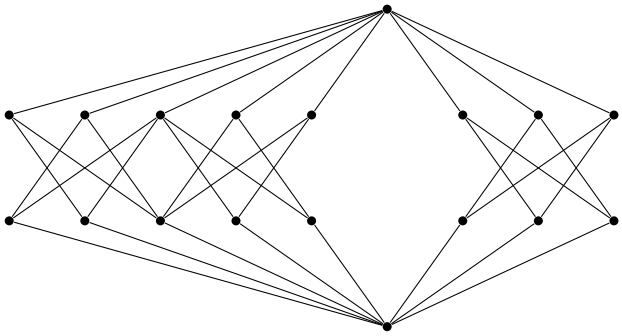

Figure 5.5: $\{0,1\}$-pasting of the logic in Figure 4.1, and a Boolean algebra with 3 atoms.

where

$$\begin{aligned} T = &\{((q_1,q_2),e,(q_1',q_2)) \mid (q_1,e,q_1') \in T_1, q_2 \in Q_2\} \quad \cup \\ &\{((q_1,q_2),e,(q_1,q_2')) \mid (q_2,e,q_2') \in T_2, q_1 \in Q_1\} \end{aligned}$$

The next lemma shows that the parallel product of transition systems and the $\{0,1\}$-pasting of logics are strictly related.

Proposition 5.3.1. *Let $A_i = (Q_i, E_i, T_i)$ be a condition/event transition system for $i = 1,2$, with $E_1 \cap E_2 = \emptyset$. Then $\mathcal{R}(A_1 \,||\, A_2)$ and $\mathcal{R}(A_1) \,||\, \mathcal{R}(A_2)$ are isomorphic logics.*

Proof. By construction, if there is a transition $((q_1,q_2),e,(q_1',q_2))$ in T, then, for each q_2' in Q_2, T contains a transition $((q_1,q_2'),e,(q_1',q_2'))$. Hence, for each region r of A_1, the set $r \times Q_2$ is a region of $A_1 \,||\, A_2$, and, for each region r of A_2, the same holds for the set $Q_1 \times r$. For any region r of $A_1 \,||\, A_2$, the projection of its states on the first component must be a region of A_1 (and symmetrically for the projection on the second component), because in any transition only one of the two components of a state will change; hence, the full set of non-trivial regions of $A_1 \,||\, A_2$ is given by $\{r \times Q_2 \mid r \in \mathcal{R}(A_1)\} \cup \{Q_1 \times r \mid r \in \mathcal{R}(A_2)\}$. □

Proposition 5.3.2. *Let L_1 and L_2 be stable regional logics. Then $L = L_1 \,||\, L_2$ is a stable regional logic.*

Proof. By Theorem 4.2.2, L can be identified with the isomorphic concrete logic where each x in L is represented by $\mathcal{S}_x$. Then, as discussed in Section 5.1.2, $L \subseteq \mathcal{R}(\mathrm{A}(L))$.

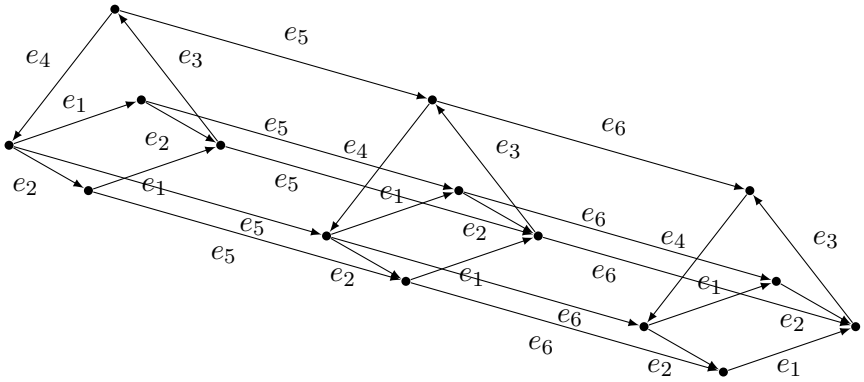

Figure 5.6: A condition/event transition system such that its poset of regions is the one depicted in Figure 5.5. It is the product of the condition/event transition system of Figure 2.7 and a sequence of three states.

Remark 5.3.1 implies that

$$\mathcal{S}(L) = \{s_1 \cup s_2 \mid s_1 \in \mathcal{S}(L_1), s_2 \in \mathcal{S}(L_2)\}.$$

Since $s_1 \cap s_2 = \{1\}$ for each choice of s_1 and s_2 above, $\mathcal{S}(L)$ can be represented as as $\mathcal{S}(L_1) \times \mathcal{S}(L_2)$.

A transition system can now be defined on $\mathcal{S}(L)$, by taking a subset of the events and transitions of the saturated transition system. The idea is to choose only the "local" transitions, namely transitions that change only one component of a state. Define

$$\begin{aligned} E_M = &\{[(s_1, s_2), (s_1', s_2)] \mid s_1, s_1' \in \mathcal{S}(L_1), s_2 \in \mathcal{S}(L_2)\} \quad \cup \\ &\{[(s_1, s_2), (s_1, s_2')] \mid s_1 \in \mathcal{S}(L_1), s_2, s_2' \in \mathcal{S}(L_2)\} \end{aligned}$$

All the transitions corresponding to labels in E must be considered.

$$\begin{aligned} T_M = &\{((s_1, s_2), [(s_1, s_2), (s_1', s_2)], (s_1', s_2)) \mid s_1, s_1' \in \mathcal{S}(L_1), s_2 \in \mathcal{S}(L_2)\} \quad \cup \\ &\{((s_1, s_2), [(s_1, s_2), (s_1, s_2')], (s_1, s_2')) \mid s_1 \in \mathcal{S}(L_1), s_2, s_2' \in \mathcal{S}(L_2)\}. \end{aligned}$$

The transition system $\mathrm{A}_M(L) = (\mathcal{S}(L), E_M, T_M)$ is a restriction of $\mathrm{A}(L)$, as such, its set of regions is a superset of $R(\mathrm{A}(L))$.

On the other hand, A_M is isomorphic to $\mathrm{A}(L_1) \,||\, \mathrm{A}(L_2)$. Apply Proposition 5.3.1, and the hypothesis of stability of L_1 and L_2 to derive

$$\mathcal{R}(\mathrm{A}_M(L)) = \mathcal{R}(\mathrm{A}(L_1) \,||\, \mathrm{A}(L_2)) = \mathcal{R}(\mathrm{A}(L_1)) \,||\, \mathcal{R}(\mathrm{A}(L_2)) = L_1 \,||\, L_2 = L$$

and

$$L \subseteq \mathcal{R}(\mathrm{A}(L)) \subseteq \mathcal{R}(\mathrm{A}_M(L)) = L$$

so that $L = \mathcal{R}(\mathrm{A}(L))$, and L is stable.

□

The construction and the argument above can be generalised to the case of the $\{0,1\}$-pasting of K logics by noting that $L_1 \,||\, L_2 \,||\, L_3$ is isomorphic to $L_1 \,||\, (L_2 \,||\, L_3)$.

5.3.3 Logics with Centers

Let us now suppose that L is a rich, regular quantum logic, which is the union of Boolean algebras such that their pairwise intersections are all the same Boolean algebra. More formally, $L = \bigcup_{i \leq n} B_i$, where $\{B_i\}_{i \leq n}$ is a finite family of finite Boolean algebras, and there is a Boolean algebra B such that $\forall i \neq j : B_i \cap B_j = B$. B corresponds to what is called the *centre* of L in [51], it is a sublogic of any B_i. An example of such a logic with $n = 2$ and $B = \{0, x, x', 1\}$ has been given in Figure 4.1 (p.87). The synthesis of its saturated transition system was discussed in 4.2.1 and is depicted in Figure 4.7 (both p.106).

It is proved that such logics are stable, for which it is required that the centre B contains at least an atom of L. This is, however, always the case, as shown in the following lemma.

Lemma 5.3.1. *Let $\{B\} \cup \{B_i\}_{i \leq n}$ be a finite family of finite Boolean algebras, and L be a logic such that $L = \bigcup_{i \leq n} B_i$, and $\forall i \neq j : B_i \cap B_j = B$. Then there is at least one atom of B which is an atom of L.*

Proof. Note that by construction, each B_i is a maximal Boolean subalgebra of L. Hence, for any $x_i \in B_i \setminus B$, and $x_j \in B_j \setminus B : i \neq j \Rightarrow x_i \not\$ x_j$. The proof proceeds by reductio ad absurdum, so suppose that no atom of B is an atom of L. Let x be an atom of B, then neither x nor x' are atoms of L. Then $\exists\, x_1 \in L \setminus B : x_1 < x$, (hence $x' < x_1'$). Suppose w.l.g. that $x_1 \in B_1$. Since $x_1 \notin B$ it holds that $\forall j \neq 1 : x_1 \notin B_j$, and so $x_1' \notin B_j$. It holds, in particular for $j = 2$. On the other hand, $B \subsetneq B_2$ and $x \notin \{0_L, 1_L\}$ imply that $\exists x_2 \in B_2 \setminus B : x \,\$\, x_2$, hence $x_2 \notin B_1$. there are two cases. Either $x \perp x_2$, or $\exists y_2 \in B_2 : y_2 = x \wedge x_2$ with $y_2 \neq 0_{B_2} = 0_L$, and since x is an atom of B: $y_2 \notin B_1$. From $x \perp x_2$, it follows that $x_2 \leq x' \leq x_1'$, so in particular $x_1' \,\$\, x_2$, hence $x_1 \,\$\, x_2$, which is in contradiction with $(x_1 \notin B_2) \wedge (x_2 \notin B_1)$. If $y_2 = x \wedge x_2$, then $x' < x_1'$ and $x' < y_2'$ with $x_1' \in B_1 \setminus B$ and $y_2' \in B_2 \setminus B$. Since, by hypothesis, x' is not an atom of L, there must be an atom y of L

such that $y < x'$, and $y \notin B$. Now, there must be some i such that $y \in B_i \setminus B$. If $i = 1$ then $y \notin B_2$, so $y \not\frown y'_2$, but $y < y'_2$, which is a contradiction. If $i \neq 1$ then $y \notin B_1$, and inconsistency follows from $y < x'_1$ and $y \not\frown x'_1$. □

The following result can now be proved.

Proposition 5.3.3. *Let $\{B\} \cup \{B_i\}_{i \leq n}$ be a finite family of finite Boolean algebras,and L be a logic such that $L = \bigcup_{i \leq n} B_i$, and $\forall i \neq j : B_i \cap B_j = B$. Then L is stable.*

Proof. Let $\mathcal{A}_L$ denote the atoms of L. Lemma 5.3.1 shows, that some atom of B is in $\mathcal{A}_L$. Since $\forall i \leq n : B \subseteq B_i$, any pair of atoms $x \in B \cap \mathcal{A}_L$, $y \in \mathcal{A}_L$ are orthogonal. Hence, a state containing an atom of L belonging to B will contain no other atom in L. This allows us to partition the states of L into two classes. On one hand, the class S_B of states that contain exactly one atom $x \in \mathcal{A}_L \cap B$. Since the family $\{B_i \setminus B\}$ is pairwise disjoint, the class S of remaining states will contain exactly one atom in each of the set differences $B_i \setminus B$.

Now, the carrier of $\mathrm{A}(L)$, is the disjoint union of S and S_B: $\mathcal{S}(L) = S \cup S_B$. Denote the states in S by $\uparrow\{x_i\}_{i \leq n}$ where each x_i is an atom of B_i, and the states in S_B by $\uparrow\{x\}$ where x is an atom of both B and L.

Let $EM(L) := \{\langle \uparrow\{x\} \setminus \uparrow\{y\}, \uparrow\{y\} \setminus \uparrow\{x\}\rangle \mid \exists i \leq n : x, y \in B_i \setminus B \text{ are atoms } \}$ be the set of events associated with local transitions among states of S, and define $TM(L)$ as the set of all transitions in $\mathrm{A}(L)$ with labels in $EM(L)$. Finally, define $\mathrm{A}_M(L) = (\mathcal{S}(L), EM(L), TM(L))$. Clearly, $\mathrm{A}_M(L)$ is a generalised transition system, which is a restriction of $\mathrm{A}(L)$. In particular, every region of $\mathrm{A}(L)$ is also a region of $\mathrm{A}_M(L)$.

$\mathrm{A}_M(L)$ is not connected because the states $\uparrow\{x\} \in S_B$ are all isolated.

Each transition in $\mathrm{A}(L)$ starting from, or leading to, any $\uparrow\{x\}$ carries a unique label, which has no other occurrence. Hence, the singleton formed by this state is a region in $\mathrm{A}(L)$, as it is in $\mathrm{A}_M(L)$. Furthermore, such singleton is disjoint to any other minimal region $\mathrm{A}(L)$, and so the corresponding region is orthogonal to all other atoms of $\mathcal{R}(\mathrm{A}(L))$. It therefore belongs to its centre. In fact, all subsets of S_B are regions of $\mathrm{A}(L)$, and as a power set, they form a Boolean algebra.

Now, each transition $t \in TM(L)$ corresponds to an ordered pair of states of S. For each such transition $t = (\uparrow\{x_i\}_{i \leq n}, e, \uparrow\{y_i\}_{i \leq n})$, there is an index $i \leq n$ such that $\forall j \neq i : x_j = y_j$. In this way, two transitions $t = (\uparrow\{x_i\}_{i \leq n}, e, \uparrow\{y_i\}_{i \leq n})$ and $\tilde{t} = (\uparrow\{u_i\}_{i \leq n}, e, \uparrow\{v_i\}_{i \leq n})$ carry the same label $e \in EM(L)$ if and only if there is an $i \leq n$, such that $x_i = u_i$, $y_i = v_i$, and $\forall j \neq i : x_j = y_j$ and $u_j = v_j$.

For each $j \leq n$, let A_j be the sequential transition system synthesised from the Boolean algebra generated by the atoms of $B_j \setminus B$. Then $EM(L)$ prevents any subset of S, which is not the disjoint union of sets of the form $\{\uparrow\{x_i\}\} \times \Pi_{j \neq i} A_j$ from being a region.

In this way, there is a natural bijection from the atoms of L to the atoms of $\mathcal{R}(\mathrm{A}(L))$, which maps the $x \in B$ to the singletons $\{\uparrow\{x\}\}$, and the $x_i \in B_i \setminus B$ to $\{\uparrow\{x_i\}\} \times \Pi_{j \neq i} A_j$. Such a bijection preserves orthogonality and incompatibility, so that the logics generated by these atoms are isomorphic.

□

Chapter 6

Conclusions and Further Research

This work has tackled the problem of distributing systems and the processes they execute. The adopted approach is characterized by the consideration of observability as a central notion. Algebraic structures allowing to handle the idea of consistent observations have been studied considering how they can be applied to the models of either processes, or systems. The models at stake are taken from Petri net theory, and among these, elementary and condition/event systems are chosen. The reason for the choice of these models is that, as Petri net models, they allow for an accurate analysis of concurrency. Moreover, elementary and condition/event systems are suitable to handle the notion of observation in logical terms.

In the case of processes, the notion of subprocess introduced in [10] allows one to determine which parts of the process can indeed be executed independently. This is done by assuming that the flow of information in the process induces the causal dependence among the occurrences that compose it. Then the causal independence can be interpreted as a lack of information flow. Hence, an observer could not have access to the information about both a subprocess, and the subprocesses which are causally independent from it. These subprocesses are described as subsets of the partial order which represents the causal dependencies of the occurrences of elements of a system. These subsets, when ordered by inclusion, form an orthomodular lattice.

The features of this structure have been exploited in order to define, for each process, an abstraction of it. The elements of this abstraction are the minimal subprocesses described in [10]. This abstraction is shown to accurately represent the two main features regarding distributability. First, when

two parts of the process can be executed independently from each other. This possibility is represented in the abstraction. Hence, the abstraction determines which elements of the process can be run remotely from each other. Second, under the assumption that the system responsible for running a process does not have unlimited resources, the abstraction depicts all the flow of information among the parts of the process. In some sense, the abstraction isolates the parts of the system which must be executed together, representing them as single elements, and gathers the information about their independence, or their interactions.

In order to analyse how a system can be distributed, the approach follows that of [7], which evolves, again, around the notion of observability. It is not studied how to distribute the system in space, by splitting its components in disjoint parts. Instead, the performed analysis provides parts of the system which can be observed from a given location. In this way, the notion of sequential component arises as including, not only a localised part of the system, but also all its interactions with the rest of the system. It is assumed that the actions performed by the system are only observable by means of their effect on its states. The theory of Petri net synthesis technically motivates the fact that only local states are observable. Furthermore, this theory allows to show that all observable properties can be implemented as local states of the system, they are identified with the regions of the transition system modelling the behaviour of a net system. Regions, when ordered by set inclusion, form an orthomodular poset, which represents the logical structure of the properties which are observable on the system. The idea that observations can only be performed locally motivates the choice of this structure as a specification of the sequential components of the system. In [7], it was shown how to build a transition system from a given specification in these terms, and that the regions of this system contain all the elements of the specification. Intuitively, this second kind of synthesis produces the model of a system in which sequential components are specified by the orthomodular poset. It is not shown however, that no more sequential components than the ones specified arise in the system.

The contribution of this work in this line of research has been of two types. First, the transition system synthesised from an orthomodular poset has been provided a structure, which binds the specification of the sequential components of the system with the concurrency it depicts. This structure provides mathematical tools to analyse how the information about the allowed observations on the system induces concurrency on it. This has provided insight on the problem of characterising the class of orthomodular posets which are suitable specifications of the sequential components. Although the full

characterisation remains a long term objective, the class has been restricted by a property named ETI (Events Testify Incompatibility).

It is shown that this property is sufficient to guarantee that pairwise independent observations will correspond to local states which are implementable remotely from one another. Since global states of the system will be composed of groups of such local states, the full characterisation then reduces to scaling this pairwise correspondence to the global states of the system.

The orthomodular lattices that arise in the study of non-sequential processes are a subclass of orthomodular posets. Since orthomodular posets arise in the study of distributed systems, as the structure of elementary regions of transition systems, it is natural to wonder about their correspondence.

An interesting aspect of this consideration is that they display concurrency dually. Orthomodular posets are, in general, characterised by the notions of orthogonality and incompatibility.

In the case of systems, orthogonality translates mutual exclusion of observations. Two observations are orthogonal when they imply each other's negation. Observations are identified with potential local states of the system. When considering all possible behaviour in a system, two local states can only be mutually exclusive if they belong to the same component, they depend on each other. In the case of processes, however, orthogonality expresses the exact opposite, that two occurrences are independent from each other. There is no flow of information among orthogonal parts of the process, and so intuitively, these can be run from different locations. Consider the maximal sets of pairwise orthogonal minimal elements. They generate the Boolean sublogics of the orthomodular poset. These represent, in the case of systems, a sequential component. In the case of processes, they represent the occurrence of an element in each of them.

Dually, by selecting a minimal element in each sequential component of a system, one obtains a global state of the system. Such a collection of minimal elements represents the local states that compose the global one. The local states in such a collection are, in particular, pairwise incompatible. Incompatibility in this sense translates the belonging to different sequential components. Two incompatible elements of the orthomodular poset obtained from a system are local states that can evolve independently from each other. Incompatibility, in the lattice obtained from a process is to be interpreted as causal dependence. And in fact, the selection of one minimal element from each Boolean algebra produces a line on the underlying partial order, a maximal sequential subprocess. Naturally, such a line corresponds to a part of the process that is required to happen in a given order. The duality between the two orthomodular posets that arise when analysing concurrency in processes and

systems motivates the study of this correspondence. In this line of research, the main challenge is to make the two models comparable. Indeed, since one is described on processes, and the other on systems, it is natural to try to transport them in both directions. In trying to transport the orthomodular lattice defined on processes to the systems that execute them, the first issue one faces is the multiplicity of the processes a system can run. In general, to one system there might correspond several processes, and hence several different orthomoduar lattices. This question was already tackled in [6], where the closure operator is extended to unfoldings. Unfoldings are compact representations of all the processes a system can run. However, the problem of going from the set of occurrences of elements of a system to the system itself is not trivial, and in particular, in the process, orthomodularity is lost.

The problem of transporting the orthomodular poset of local states of a system to the framework of processes is not less challenging. Indeed, this model represents which local states belong to the same sequential component, but carries no information about the orientation of their causal dependences. In this line, an appropriate approach to try to transport the orthomodular poset of regions to the realm of processes would be to investigate the possible orderings admitted among sets of pairwise orthogonal elements. In this line, it seems that some results from graph theory could be applied, to be noted in particular, the work on comparability graphs of, for example [43]. However, such an ordering does not seem possible in general in orthmodular posets, and so these results would depend on properties particular to regional structures. This consideration calls for the full characterisation of the class of the latter.

Naturally, this characterisation problem is the most relevant continuation of the present work. Regarding this line of research, it is to be noted that with the results provided in this work, the characterisation reduces to extending the correspondence between concurrency of events, and incompatibility of local states from pairs of elements to full global states. The way local states compose global states, however, strongly depends on the instance of orthomodular poset under consideration, and the possible combinations grow very fast, even on small models. It would be suitable to identify a property of regional partial orders, that would narrow down the possibilities. Such a property would be, in some sense, complementary to the one presented in this work, in fully characterising the class of regional logics.

Finally, another open line of research is investigating the composition of regional logics. As specifications of the sequential components of the system, the modes of communication of the latter are specified by the way they overlap. It would be interesting to define an operation which, given the specifications of the structure of sequential components of the system, and a specification

of an interface, merges them into one single model that would represent the two parts interacting as specified. Regarding such a composition, the main advantage of the approach followed in this work, is that the components of the system, and the modes of interaction, admit a formalisation in the same terms, thus reducing such a composition to the composition of two models with respect to a third, all three formalised with the same mathematical apparatus. Such a construction is well known in the literature, as amalgam. To this aim, two approaches seem equally promising. The first one involves representability of orthomodular posets as subsets of their sets of states. This relates it closely to the composition of systems. The second would exploit representability in terms of atoms, thus relating it closely to the theory of test spaces. In both cases, the main challenge is to show that the result of such a composition operation lies in the same class of models as its operands. This is known not to be the case in general, and intuitively this means that not every such composition would result in a regional poset.

CETERUM CENSEO MATROIDEM QUEMDAM ESSE

Bibliography

[1] Adrián Puerto Aubel. Concurrency-preserving minimal process representation. In Peter Höfner, Damien Pous, and Georg Struth, editors, *Relational and Algebraic Methods in Computer Science - 16th International Conference, RAMiCS 2017, Lyon, France, May 15-18, 2017, Proceedings*, volume 10226 of *Lecture Notes in Computer Science*, pages 242–257, 2017.

[2] E. Badouel, L. Bernardinello, and P. Darondeau. *Petri Net Synthesis*. Texts in Theoretical Computer Science. An EATCS Series. Springer, 2015.

[3] E. Badouel, B. Caillaud, and P. Darondeau. Distributing finite automata through petri net synthesis. *Formal Aspects of Computing*, 13:447–470, 2002.

[4] C. Berge. *Graphs and Hypergraphs*. North-Holland mathematical library. Amsterdam, 1973.

[5] L. Bernardinello. Synthesis of net systems. In M. A. Marsan, editor, *Application and Theory of Petri Nets 1993, 14th International Conference, Chicago, Illinois, USA, June 21-25, 1993, Proceedings*, volume 691 of *Lecture Notes in Computer Science*, pages 89–105. Springer, 1993.

[6] L. Bernardinello, C. Ferigato, S. Haar, and L. Pomello. Closed sets in occurrence nets with conflicts. *Fundamenta Informaticae*, 133(4):323–344, 2014.

[7] L. Bernardinello, C. Ferigato, and L. Pomello. An algebraic model of observable properties in distributed systems. *Theoretical Computer Science*, 290(1):637–668, 2003.

[8] L. Bernardinello and L. Pomello. A category of transition systems and its relations with orthomodular posets. In I. Prívara and P. Ruzicka,

editors, *Mathematical Foundations of Computer Science 1997, 22nd International Symposium, MFCS'97, Bratislava, Slovakia, August 25-29, 1997, Proceedings*, volume 1295 of *Lecture Notes in Computer Science*, pages 139–148. Springer, 1997.

[9] L. Bernardinello, L. Pomello, and S. Rombolà. Orthomodular lattices in occurrence nets. In G. Franceschinis and K. Wolf, editors, *Applications and Theory of Petri Nets, 30th International Conference, PETRI NETS 2009, Paris, France, June 22-26, 2009. Proceedings*, volume 5606 of *Lecture Notes in Computer Science*, pages 163–182. Springer, 2009.

[10] L. Bernardinello, L. Pomello, and S. Rombolà. Closure operators and lattices derived from concurrency in posets and occurrence nets. *Fundamenta Informaticae*, 105(3):211–235, 2010.

[11] L. Bernardinello, L. Pomello, and S. Rombolà. On orthomodular posets generated by transition systems. *Electronic Notes Theoretical Computer Science*, 270(1):147–154, 2011.

[12] L. Bernardinello, L. Pomello, and S. Rombolà. Orthomodular algebraic lattices related to combinatorial posets. In S. Bistarelli and A. Formisano, editors, *Proceedings of the 15th Italian Conference on Theoretical Computer Science, Perugia, Italy, September 17-19, 2014.*, volume 1231 of *CEUR Workshop Proceedings*, pages 241–245. CEUR-WS.org, 2014.

[13] Luca Bernardinello, Carlo Ferigato, Lucia Pomello, and Adrián Puerto Aubel. Synthesis of transition systems from quantum logics. *Fundam. Inform.*, 154(1-4):25–36, 2017.

[14] Luca Bernardinello, Carlo Ferigato, Lucia Pomello, and Adrián Puerto Aubel. On stability of regional orthomodular posets. *T. Petri Nets and Other Models of Concurrency*, 13:52–72, 2018.

[15] Luca Bernardinello, Carlo Ferigato, Lucia Pomello, and Adrián Puerto Aubel. On the decomposition of regional events in elementary systems. In Wil M. P. van der Aalst, Robin Bergenthum, and Josep Carmona, editors, *Proceedings of the International Workshop on Algorithms & Theories for the Analysis of Event Data 2018 Satellite event of the conferences: 39th International Conference on Application and Theory of Petri Nets and Concurrency Petri Nets 2018 and 18th International Conference on Application of Concurrency to System Design ACSD 2018, Bratislava, Slovakia, June 25, 2018.*, volume 2115 of *CEUR Workshop Proceedings*, pages 39–55. CEUR-WS.org, 2018.

[16] E. Best and P. Darondeau. Petri net distributability. In E. M. Clarke, I. Virbitskaite, and A. Voronkov, editors, *Perspectives of Systems Informatics - 8th International Andrei Ershov Memorial Conference, PSI 2011, Novosibirsk, Russia, June 27-July 1, 2011, Revised Selected Papers*, volume 7162 of *Lecture Notes in Computer Science*, pages 1–18. Springer, 2011.

[17] E. Best and C. Fernandez. *Nonsequential Processes–A Petri Net View*, volume 13 of *EATCS Monographs on Theoretical Computer Science*. Springer-Verlag, 1988.

[18] G. Birkhoff. *Lattice Theory*. American Mathematical Society; 3rd Ed., 1979.

[19] G. Birkhoff and J. Von Neumann. The logic of quantum mechanics. *Annals of Mathematics*, 37(4):823–843, 1936.

[20] A. M. Borzyszkowski and Philippe Darondeau. Transition systems without transitions. *Theoretical Computer Science*, 338(1):1 – 16, 2005.

[21] W. Brauer, W. Reisig, and G. Rozenberg, editors. *Petri Nets: Central Models and Their Properties, Advances in Petri Nets 1986, Part II, Proceedings of an Advanced Course, Bad Honnef, Germany, 8-19 September 1986*, volume 255 of *Lecture Notes in Computer Science*. Springer, 1987.

[22] G. Bruns and J. Harding. Amalgamation of ortholattices. *Order*, 14(3):193–209, Sep 1997.

[23] J. Carmona, J. Cortadella, V. Khomenko, and A. Yakovlev. Synthesis of asynchronous hardware from petri nets. In J. Desel, W. Reisig, and G. Rozenberg, editors, *Lectures on Concurrency and Petri Nets, Advances in Petri Nets [This tutorial volume originates from the 4th Advanced Course on Petri Nets, ACPN 2003, held in Eichstätt, Germany in September 2003. In addition to lectures given at ACPN 2003, additional chapters have been commissioned]*, volume 3098 of *Lecture Notes in Computer Science*, pages 345–401. Springer, 2003.

[24] E. M. Clarke, O. Grumberg, and D. A. Peled. *Model checking*. MIT Press, 2001.

[25] G. Coulouris, J. Dollimore, and T. Kindberg. *Distributed Systems. Concepts and Design. 3rd edition.* Addison-Wesley, 01 2000.

[26] J. C. Dacey. Orthomodular Spaces. Phd thesis, Univ. Massachusetts, Amherst, Mass., 1968.

[27] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, 1990.

[28] Luca de Alfaro and Thomas A. Henzinger. Interface automata. In A. Min Tjoa and Volker Gruhn, editors, *Proceedings of the 8th European Software Engineering Conference held jointly with 9th ACM SIGSOFT International Symposium on Foundations of Software Engineering 2001, Vienna, Austria, September 10-14, 2001*, pages 109–120. ACM, 2001.

[29] J. Desel and W. Reisig. The synthesis problem of petri nets. *Acta Informatica*, 33(4):297–315, 1996.

[30] A. Ehrenfeucht and G. Rozenberg. Partial (set) 2-structures. part I: basic notions and the representation problem. *Acta Informatica*, 27(4):315–342, 1990.

[31] A. Ehrenfeucht and G. Rozenberg. Partial (set) 2-structures. part II: state spaces of concurrent systems. *Acta Informatica*, 27(4):343–368, 1990.

[32] B. Finkbeiner and S. Schewe. Uniform distributed synthesis. In *20th IEEE Symposium on Logic in Computer Science (LICS 2005), 26-29 June 2005, Chicago, IL, USA, Proceedings*, pages 321–330. IEEE Computer Society, 2005.

[33] R.J. Greechie. On the structure of orthomodular lattices satisfying the chain condition. *Journal of Combinatorial Theory*, 4(3):210 – 218, 1968.

[34] R.J. Greechie. Orthomodular lattices admitting no states. *Journal of Combinatorial Theory, Series A*, 10(2):119 – 132, 1971.

[35] J. Harding. Decompositions in quantum logic. *Transactions of the American Mathematical Society*, 348(5):1839–1862, 1996.

[36] J. Harding. *The source of the orthomodular law*, pages 555 – 586. Elsevier Science B.V., Amsterdam, 2007.

[37] R. I. G. Hughes. *The Structure and Interpretation of Quantum Mechanics*. Harvard University Press, 1989.

[38] L. Iturrioz. Orthomodular ordered sets and orthogonal closure spaces. *Portugaliae mathematica*, 39(1-4):477–488, 1980.

[39] G. Kalmbach. *Orthomodular Lattices*. Academic Press, New York, 1983.

[40] L. Lamport. Time, clocks, and the ordering of events in a distributed system. *Communications of the ACM*, 21(7):558–565, July 1978.

[41] F. Mattern. Virtual time and global states of distributed systems. In M. Cosnard et. al., editor, *Parallel and Distributed Algorithms: proceedings of the International Workshop on Parallel & Distributed Algorithms*, pages 215–226. Elsevier Science Publishers B. V., 1989.

[42] A. W. Mazurkiewicz. Trace theory. In Brauer et al. [21], pages 279–324.

[43] R. H. Möhring. *Algorithmic Aspects of Comparability Graphs and Interval Graphs*, pages 41–101. Springer Netherlands, Dordrecht, 1985.

[44] T. Murata. Petri nets: Properties, analysis and applications. *Proceedings of the IEEE*, 77(4):541–580, April 1989.

[45] M. Nielsen, G. D. Plotkin, and G. Winskel. Petri nets, event structures and domains, part I. *Theoretical Computer Science*, 13:85–108, 1981.

[46] C. A. Petri. Kommunikation mit Automaten. Dissertation, Schriften des IIM 2, Rheinisch-Westfälisches Institut für Instrumentelle Mathematik an der Universität Bonn, Bonn, 1962.

[47] C. A. Petri. Nicht-sequentielle Prozesse. ArbeitsbE.hte des IMMD 8, Universität Erlangen Nürnberg, 1976.

[48] C. A. Petri. General net theory. In B. Shaw, editor, *Computing System Design: Proc. of the Joint IBM University of Newcastle upon Tyne Seminar, Sep., 1976*, pages 131–169. University of Newcastle upon Tyne, 1977.

[49] A. Pnueli and R. Rosner. On the synthesis of a reactive module. In *Conference Record of the Sixteenth Annual ACM Symposium on Principles of Programming Languages, Austin, Texas, USA, January 11-13, 1989*, pages 179–190. ACM Press, 1989.

[50] A. Pnueli and R. Rosner. Distributed reactive systems are hard to synthesize. In *31st Annual Symposium on Foundations of Computer Science, St. Louis, Missouri, USA, October 22-24, 1990, Volume II*, pages 746–757. IEEE Computer Society, 1990.

[51] P. Pták and S. Pulmannová. *Orthomodular Structures as Quantum Logics*. Kluwer Academic Publishers, 1991.

[52] S. Pulmannov and Z. Rieanov. Block-finite atomic orthomodular lattices. *Journal of Pure and Applied Algebra*, 89(3):295 – 304, 1993.

[53] C. H. Randall and D. J. Foulis. Operational Statistics. I. Basic Concepts. *Journal of Mathematical Physics*, 13:1667–1675, November 1972.

[54] C. H. Randall and D. J. Foulis. Operational statistics. II. Manuals of operations and their logics. *Journal of Mathematical Physics*, 14:1472–1480, October 1973.

[55] C. H. Randall and D. J. Foulis. *The Operational Approach to Quantum Mechanics*, pages 167–201. Springer Netherlands, Dordrecht, 1979.

[56] W. Reisig and G. Rozenberg, editors. *Lectures on Petri Nets I: Basic Models, Advances in Petri Nets, the volumes are based on the Advanced Course on Petri Nets, held in Dagstuhl, September 1996*, volume 1491 of *Lecture Notes in Computer Science*. Springer, 1998.

[57] G. Rozenberg and J. Engelfriet. Elementary net systems. In Reisig and Rozenberg [56], pages 12–121.

[58] M. Silva. *Modeling, Analysis and Control of Discrete Event Systems as Petri Nets*, pages 1–12. Springer London, London, 2013.

[59] E. Smith and W. Reisig. *The Semantics of a Net is a Net*, pages 461–479. Springer Berlin Heidelberg, Berlin, Heidelberg, 1987.

[60] M. H. Stone. The theory of representation for boolean algebras. *Transactions of the American Mathematical Society*, 40(1):37–111, 1936.

[61] A. Wilce. *Test Spaces and Orthoalgebras*, pages 81–114. Springer Netherlands, Dordrecht, 2000.

[62] G. Winskel. Event structures. In Brauer et al. [21], pages 325–392.

[63] Wieslaw Zielonka. Notes on finite asynchronous automata. *ITA*, 21(2):99–135, 1987.