Analysis and Synthesis of Logics

APPLIED LOGIC SERIES

VOLUME 35

SCOPE OF THE SERIES

Logic is applied in an increasingly wide variety of disciplines, from the traditional subjects of philosophy and mathematics to the more recent disciplines of cognitive science, computer science, artificial intelligence, and linguistics, leading to new vigor in this ancient subject. Springer, through its Applied Logic Series, seeks to provide a home for outstanding books and research monographs in applied logic, and in doing so demonstrates the underlying unity and applicability of logic.

The titles published in this series are listed at the end of this volume.

Analysis and Synthesis of Logics

How to Cut and Paste Reasoning Systems

by

Walter Carnielli
University of Campinas (UNICAMP), Brazil

Marcelo Coniglio
University of Campinas (UNICAMP), Brazil

Dov M. Gabbay
King's College, London, United Kingdom

Paula Gouveia
Instituto Superior Tecnico, Lisbon, Portugal

and

Cristina Sernadas
Instituto Superior Tecnico, Lisbon, Portugal

Springer

Library of Congress Control Number: 2008920294

ISBN 978-1-4020-6781-5 (HB)
ISBN 978-1-4020-6782-2 (e-book)

Published by Springer,
P.O. Box 17, 3300 AA Dordrecht, The Netherlands.

www.springer.com

Printed on acid-free paper

Contents

Preface

The aim of this book is to show how logics can be cut and paste in order to be applied to express and model problems in several distinct areas. The universal applicability of logic in both pure and applied science is a fact that defies philosophers. Contemporary logical research, however, has an undeniable tendency towards pluralism and compartimentation, as shown by the division of philosophical logic in areas and subfields. On the one hand, we have logics alternative to classic, such as many-valued logic, intuitionistic logic, paraconsistent logic. On the other hand, we also have logics complementary to classic, such as modal logics, and, in particular, temporal logic, epistemic logic, doxastic logic, erotetic logic, deontic logic, and so on.

Considering that reasoning is through process, this compartimentation, even if driven by methodological and technical reasons, has been said to be harmful to logic while a philosophical discipline. From this viewpoint, combinations of logics goes in the opposite direction of restoring the entirety of logic as wide theory of rationality, much in the same spirit to what happens in areas as algebraic geometry. Thus, from a philosophical perspective, logical combinations of tense and modality, for instance, may offer a better look to issues in the theory of causation and action. Combining temporal logic with alethic modal logics adds a temporal dimension to knowledge and belief.

Conceptually, the idea of looking to logic as an entirety instead of isolated fragments is not new. Philosophers and logicians from Ramón Lull (1235–1316), in his *Ars Magna*, to Gottfried W. Leibniz (1646–1716), with his *calculus ratiocinator* [179], have dreamed of building schemes or even machines that can reason by combining different logics or logic-like mechanisms that could cooperate instead of competing.

The activity of combining logics, as seen nowadays, offers an important tool for modularity. Rather than building a logic from scratch, it may be better for some applications to depend upon previous work on specialized topics. The underlying idea is that logics can be reusable, leading to a perspective gain with the resulting combined system. However, there are many technical difficulties if one is interested in the practical activity of combining logics. Symbols may mean different things in different logics. How is it possible to define the languages in order to compose them into an organic entity? Also, proofs and derivations can have different meanings in different logics. How to thread rules and derivation schemes of totally different nature?

Combining logics have also a surprising impact on philosophical questioning, an aspect that this book is not aimed to, but that should not be overlooked. An illustrative example is the well-known David Hume's objection from concluding a normative statement of the form "ought to be" starting from a descriptive statement of the form "what is" (a much discussed question in moral theory). So, for instance, from statements of fact, such as "emission of carbon dioxide is harmful to society", a statement of obligation such as "all nations ought to follow mandatory emission limitations" could not, according to an interpretation of Hume's ideas, be derived.

From the point of view of combining logics, this question is strictly connected to accepting properties of combining deontic and alethic logics, such as $p \rightarrow (Op)$, where O is the deontic obligation operator and p is an assertion. Such formula is what we call a "bridge principle". The term, meaning specifically a statement that binds factualities to norms, appears already in [3] and subsequently in [262].

In our treatment, by "bridge principles" we mean, in a wide sense, any new derivations among distinct logic operators (new in the sense of not being instances of valid derivations in the individual logics being combined).

Another much discussed thesis is the famous "ought-implies-can" thesis attributed to I. Kant, according to which the fact that we ought to do something implies that it has to be logically possible to do. This would be formalized through the following bridge principle: $(Op) \rightarrow (\Diamond p)$, where the diamond $\Diamond$ denotes the alethic "possibility" operator. Thus the principle means that if an assertion is obligatory then it must be possible. Other interpretations suggest that what Kant believed is that we cannot be obliged to do something if we are not capable of acting in that way. This would be formalized by the (non necessarily equivalent) bridge principle $(\neg\Diamond p) \rightarrow (\neg Op)$.

Not only bridge principles have an underlying conceptual meaning, but they also may emerge spontaneously, with surprising consequences as we show in many places of this book. The influence of bridge principles is yet perceived in the way the *collapsing problem* (a phenomenon of combining logics by which, for instance, intuitionistic collapses to classical logic) is solved (see Chapter 8).

This book intends to address the questions presented above in detail, presenting with the foremost rigor the issues of logical manipulation. The reader will learn here how to set up the syntactical dimension in detail, and how to define the semantics and the proof theory for recombinant logics. The impact of combination of logics in practice can only be assessed by people involved in the application domains. However, we believe that these techniques can be useful in fields such as computational linguistics, automated theorem proving, complexity and artificial intelligence. Other promising applications are in the areas of software specification, knowledge representation, architectures for intelligent computing and quantum computing, security protocols and authentication, secure computation and zero-knowledge proof systems and in the formal ethics of cryptographic protocols.

Combinations of temporal reasoning, reasoning in description logic, reasoning about space and distance are becoming a relevant toolbox in modeling multi-agent

systems. The resulting hybrid systems have the main advantage of combining logics which would be otherwise incompatible. Proof procedures with controlled complexity, model checking and satisfiability checking procedures can be obtained for a bigger logic from the respective procedures for the component logics.

But the reader should not think that combinations of logics is a topic restricted to applications outside logic. On the contrary, although we do not deal with this question in this book, the very idea of combining logics, as we see it, touches on more abstract domains as applied to the logical theory itself: for instance, as suggested in [233]. The idea of combining logics can be even useful to understand apparently far away topics such as Popper's structuralist theory of logic, as in [223], where an elementary theory of combining negations was developed.

In a rigorous way, the problem of combining logics can be seen as follows: given two logics $\mathcal{L}_1$ and $\mathcal{L}_2$ we want to combine them and obtain a new logic $\mathcal{L}$ satisfying certain requirements. In general, there are several mechanisms to combine the original logics. Choosing mechanism $\odot$, the new logic is $\mathcal{L} = \mathcal{L}_1 \odot \mathcal{L}_2$. That is, $\odot$ is an operator on some class of logics including $\mathcal{L}_1$ and $\mathcal{L}_2$. Different operators may lead to different resulting logics. Most of the operators provide an algorithmic construction of logic $\mathcal{L}$ by stating its language, semantic structures and/or deductive systems. Moreover, the construction of $\mathcal{L}$ usually is a minimal (or maximal) construction. The combined logic should extend the components in a controlled way, so that it does not include undesirable features.

All the mechanisms assume that the component logics are presented in the same way. In technical terms we say they are homogeneous. For instance, both of them are presented by Hilbert calculi. However, some assume that component logics need some preparation before being combined. For instance, assume that we say that component logics are endowed with an algebraic semantics. In this case, we have to say how the semantic structures of the component logics induce an algebraic semantics. In the book we deal with heterogeneous fibring in a moderated way in Chapter 3 and with heterogeneous fibring of deductive systems in Chapter 4.

One of the most challenging problems is related to proving transference results. That is, to investigate sufficient conditions for the preservation of properties, namely soundness, completeness, decidability, consistency, interpolation, from the components into the resulting logic.

Combination mechanisms can be extended to a finite number of components and sometimes even to an infinite number of components.

Among the different combination mechanisms we can refer to fibring which is one of the objects of this book. Fusion, if not historically the first, is the simplest method, and the best studied combination mechanism.

Combining logics in the perspective of this book does not mean only synthesizing or composing logics (which is called splicing), but is also intended to work in the opposite direction of decomposing logics, called splitting. Herein, we analyze the possible-translations semantics mechanism.

The idea of writing this book originated during *The Workshop on Combination of Logics: Theory and Applications* (CombLog04) [50, 52], held in the Center for

Logic and Computation, at the Department of Mathematics of IST, Technical University of Lisbon, Portugal, from July 28-30, 2004. Encouraged by the vigor of the field and by the interest triggered by this and several other conferences (such as [234, 112, 162, 9, 138, 10, 11]), we decided to accept the challenge to produce a book containing some basic ideas, methods and techniques that could help logicians, computer scientists and philosophers to have access to a general yet elementary theory of combinations of logics. The book intended to bring together a sample of results, problems and perspectives involving the idea of cutting and pasting logics, explaining when possible the role of the underlying constructions as universal arguments in the categorial sense.

We depart here from a basic universe of logic systems starting with propositional-based systems endowed with Hilbert calculi and ordered algebraic semantics. This basic setting is already rich enough to encompass interesting features of fibring with several applications and to provide the basic techniques for the trade of combining systems. Later on we extend the notion to the first-order and to the higher-order domains.

Chapter 1 is an introductory overview to the essential ingredients of composing and decomposing logics. In Section 1.1, we introduce the concept of consequence system as the basic abstraction to describe a logic system. In Section 1.2, we present the basic ideas about composing or splicing logics and decomposing or splitting logics. We also introduce a technical summary of some combination mechanisms like fusion, product and fibring by functions of modal logics. We also refer to Gödel provability logic as an illustration of a splitting mechanism. In Section 1.3, we provide a very brief introduction to algebraic fibring using Hilbert calculi. In Section 1.4, we sketch the splitting mechanism called possible-translations semantics.

Chapter 2 concentrates on fibring of propositional based logics presented as Hilbert calculi. Moreover, some preservation results are introduced. In Section 2.1, signatures and their fibring are presented. In Section 2.2, we dedicate our attention to the fibring of Hilbert calculi. We illustrate the concepts with several examples including classical logic, modal logics, intuitionistic logic, 3-valued Gödel and Łukasiewicz logics. In Section 2.3, we discuss several preservation results. Finally, Section 2.4 presents some final remarks.

Chapter 3 is dedicated to the fibring of semantics for propositional based logics. Ordered algebras are the basic semantic structures adopted. We also include the relationship to fusion and fibring by functions. Again some preservation results are given. In Section 3.1, we introduce the semantic structures and their fibring. We illustrate the concepts with several examples including classical logic, modal logics, intuitionistic logic, 3-valued Gödel and Łukasiewicz logics. In Section 3.2, we present the notions of logic system, soundness and completeness. In Section 3.3, we discuss the preservation of soundness and completeness properties. In Section 3.4, we establish the relationship between the present approach and fibring by functions. In Section 3.5 we present some final comments.

Chapter 4 is dedicated to the analysis of fibring of logics that are not presented in the same way. Two solutions are proposed. The first one is based on fibring of consequence systems and the second one on abstract proof systems. Some preservation results are established. Section 4.1 concentrates on fibring of consequence systems using a fixed point operator. Several examples are given for logics presented either in a proof-theoretic or a model-theoretic way. Section 4.2 focuses on the notion of abstract proof system and looks at the proof systems induced by Hilbert, sequent and tableau calculi. Moreover, it includes the notion of fibring of abstract proof systems. We also discuss some relationships between consequence systems and proof systems. In Section 4.3 we present some final remarks.

Chapter 5 studies composition of non-truth functional logics via fibring, an important extension of the theory, considering that many of the interesting logics for applications are not truth-functional. In Section 5.1, the notion of interpretation system presentation is introduced. In Section 5.2 the notions of unconstrained and constrained fibring of interpretation system presentations is defined. In Section 5.3 we again use Hilbert calculus as the suitable proof-theoretic notion. In Section 5.4 some preservation results are established, namely, the preservation of soundness and completeness. Section 5.5 discusses self-fibring in the context of non-truth-functional logics. In Section 5.6 we present some final comments.

Chapter 6 concentrates on fibring of first-order based logics. It can be seen as an extension of the fibring of propositional based logics, choosing particular powerset algebras as semantic structures. The running example is fibring of classical first-order logic and modal logic. In Section 6.1, first-order based signatures and the corresponding languages are introduced. Next, in Section 6.2, we present interpretation structures and interpretation systems. First-order Hilbert calculi are presented in Section 6.3. Section 6.4 introduces first-order logic systems. Then, in Section 6.5, we define fibring of first-order based logics. The preservation of completeness and other metatheorems by fibring is discussed in Section 6.6, where we also briefly sketch a proof of completeness for a particular class of first-order logic systems. In Section 6.7 we make some final remarks.

Chapter 7 deals with higher-order quantification logics. The semantic structures are generalizations of the usual topos semantics for higher-order logics. In Section 7.1 we introduce the relevant signatures. In Section 7.2 the Hilbert calculus is presented. In Section 7.3 is dedicated to setting up the semantic notions. Section 7.4 introduces the notion of logic system, and we briefly discuss some related notions such as soundness and completeness. The novelty here is that the usual notion of soundness must be modified in the present framework. In Section 7.5, a general completeness theorem is established. In Section 7.6, the notions of constrained and unconstrained fibring of logic systems are given, and it is shown that soundness is preserved by fibring and a completeness preservation result is obtained. In Section 7.7 we briefly discuss the main results described in the chapter.

In Chapter 8, we turn our attention to modulated fibring. This variant was developed to cope with collapsing problems: in some cases when two logics are combined one of them collapses with the other. We illustrate the concepts with

examples including propositional logic, intuitionistic logic, 3-valued Gödel and Łukasiewicz logics. In Section 8.1, we introduce the notions of modulated signature and modulated signature morphisms. In Section 8.2, we describe modulated interpretation structures, modulated interpretation systems and the corresponding morphisms. Next we present the notion of bridge between modulated interpretation systems. In Section 8.3, we define modulated Hilbert calculus and their morphisms. In Section 8.4 is dedicated to modulated logic systems and their corresponding morphisms. In Section 8.5, we establish soundness and completeness preservation results. Finally, Section 8.6 presents some final comments.

Chapter 9 introduces the problem of splitting logics, emphasizing the role of possible-translations semantics and contrasting with the previous chapters that deal with forms of splicing. In Section 9.1, a category of propositional based signatures suitable for splitting logics is introduced, as well as the corresponding category of consequence systems. In Section 9.2 the technique known as possible-translations characterization is analyzed, and some applications are given. In Section 9.3 two methods for combining matrix logics, plain fibring and direct union of matrices, are reviewed. Finally, Section 9.4 presents some final comments.

In Chapter 10 we discuss new tendencies on fibring. In Section 10.1, we motivate network fibring using modal logic. In Sections 10.2, 10.3, 10.4 and 10.5, some case-studies are introduced. Section 10.2 discusses integration of information flows by describing a system in which reasoning and proofs from different sources of information can be accommodated. In Section 10.3, we refer to some generalizations of logic input/output operations. We also discuss how to combine input/output operations into networks. In Section 10.4, we discuss the fibring of neural networks. In Section 10.5, we turn our attention to recursive Bayesian networks. In Section 10.6, the notion of self-fibring of networks is introduced. Section 10.7 presents some concluding remarks.

Finally, in Chapter 11 we first present a summing-up of the different techniques for combining logics presented in this book together with their main features. Then we move to a brief overview of applications where fibring can be directly used, as well as to emergent fields of application. It also includes an outlook of new research directions in both the existing combination mechanisms but also to new forms of combination.

We observe that most chapters of the book deal with combination of logics rather than with decomposition of logics. This happens because splitting mechanisms are not so well developed.

We assume that the readers are familiar with basic logic notions of classical propositional and first-order logics at the level of, for instance, [206] and [90] and propositional modal logics at the level of, for instance, [153]. Although not mandatory, a very basic knowledge of categories (see [186]) is useful for better understanding the minimality of the constructions.

The book is intended to be a research monograph for those that want to know the state-of-the art in composing and decomposing logics, that want to know about issues worthwhile to be pursued, as well as potential contemporary applications of

these techniques. If you are one of these we recommend that you have the patience to read the whole book. If you want to focus on particular aspects of combination of logics, we suggest several paths hoping that one of them is of your taste.

- If the reader is only interested in knowing what are the main issues in the combination of logics, we recommend Chapters 1, 2 and 3 which provide a basic account on consequence systems and the basic notions of propositional fibring;
- If you are curious about decomposition and its importance to non-truth functional logics you should read Chapters 1 and 9 and maybe it is useful reading Chapter 5;
- The reader interested in a more general form of fibring (capturing more propositional-based logics) and avoiding the well known collapsing problem should concentrate on Chapter 8, besides Chapters 1, 2 and 3;
- Someone with research interest in proof systems and how to combine different proof systems should read Chapters 1, 2 and 4;
- If your interests are in modal logic, we suggest you read Chapters 1, 2, 3, 4 and 6;
- If your interests are in hybrid logic and labeled deduction in general, we suggest you read Chapters 1, 2, 3, 4 and 10;
- If you are a first-order logician and have curiosity on combination of logics, we suggest you read Chapters 1, 2, 3 and, more importantly, Chapter 6;
- If you are a higher-order logician and want to grasp what is combination of logics, we suggest you read Chapters 1, 2, 3 and, of course, Chapter 7;
- If you are an intuitionistic logician and would like to know about combination of logics, we suggest you read Chapters 1, 2, 3 and Chapter 8;
- If you like to know the potential of combination in contexts that are not logical in nature you should read Chapter 10.

For a summing-up of the techniques used in the book, as well as some applications and topics for further research, we recommend Chapter 11.

Acknowledgments

The authors are in debt to their friends and colleagues that participated in FibLog project: Amílcar Sernadas, Carlos Caleiro, Jaime Ramos, João Rasga, Luís Cruz-Filipe and Paulo Mateus.

A special thanks is due to Amílcar Sernadas whose ideas and work is present in many of the chapters of the book. Also to Carlos Caleiro who collaborated in the

comparison between fibring by functions and algebraic fibring and in the fibring of non-truth functional logics, to Alberto Zanardo for his collaboration in the first proof of completeness preservation by fibring and fibring of first-order based logics, to João Rasga for his collaboration in the definition of modulated fibring and preservation of interpolation, to Luís Cruz-Filipe for his collaboration in heterogeneous fibring and to Víctor Fernández for his collaboration in the definition of plain fibring. We are thankful to Paulo Mateus for his help with security issues, in particular, related to zero-knowledge protocols. We are grateful to Amílcar Sernadas and Paulo Mateus for their advise on quantum related issues. We also acknowledge work with Luca Viganò although not directly reflected in this book. We are in debt to Miguel Dionísio for his precious help with several LaTeXproblems.

We would like to thank the Security and Quantum Information Security Group at Instituto de Telecomunicações (SQIG - IT) and Center for Logic and Computation (CLC) and all the colleagues there. This research was supported by Fundação para a Ciência e a Tecnologia (FCT) and FEDER, namely through CLC, IT and the FibLog POCTI/2001/MAT/37239, the QuantLog POCI/MAT/55796/2004 and the KLog PTDC/MAT/68723/2006 Projects. This research was also supported by The State of São Paulo Research Foundation (FAPESP), Brazil, through the Thematic Project number 2004/14107-2 ("ConsRel"). Walter Carnielli and Marcelo Coniglio also acknowledge support from The National Council for Scientific and Technological Development (CNPq), Brazil, through individual research grants.

Lisbon, July of 2007
The Authors

Chapter 1

Introductory overview

It is not an easy matter to trace down the origins of the idea of combining reasoning (in a schematic or semi-formal manner) and relations upon them (by means of diagram, rules or other mechanisms). In a sense this has origins in the history of European philosophy itself: Plato's dialogue *Sophist* inquires about the methods of philosophy, and part of his conclusions involves the limitations of common language, and the danger of using common language which may lead to fallacious conclusions. For example, Plato shows that "not being" is a form of "being", a confusion that common language cannot help to cope with.

On the other hand, logic is the branch of knowledge where language receives the highest systematized treatment. Concepts, such as time, belief, knowledge, inheritance, relevance and dependence, their mutual relations and the methods to draw conclusions from them can be expressed in the most convenient way. Contemporary logic has pushed this tendency to extremes, with not entirely positive consequences, in the opinion of critiques (see [267]). So, in a sense, if the logical analysis taken to an extreme has separated concepts and methods, we wonder why not join them together.

Under the light of today's logic, this can be achieved with greater accuracy. This is precisely the object of this book: how to cut and paste logics and how to use them. But the roots of the idea are much older. The Catalan mystic and logician Ramón Llull, born in 1235, used logic and mechanical methods based on symbolic notation and combinatorial diagrams to relate, he thought, all forms of knowledge. His main work, the *Ars Magna*, makes him, at the same time, a precursor of combinatorics and of the art of combining logics. The method of Llull consisted of a series of concentric circles with attributes such as *bonitas* (goodness), *magnitudo* (greatness), *eternitas* (eternity), and categories such as *flame*, *man*, *animal*, which could be combined with, for example *where?*, *why?* and *how?*. By combining them, one could (at least try) to solve riddles such as "Where does the flame go when a candle is put out?".

Llull inaugurated the idea of a "philosophical machine" and had a great influence on Gottfried Leibniz. According to Couturat in [64], Leibniz was the first to see explicitly the possibilities of applying Llull's methods to formal logic in his *Dissertatio de Arte Combinatoria*, where exhaustive techniques to combine premises and conclusions in the language of Aristotelian syllogisms was considered. Later on, the British economist and logician William Steven Jevons, famous for the invention of a "logic machine" to draw syllogistic conclusions, used similar ideas in his "logical alphabet". From some point on, the attempts to combine logical devices lead to the hardware side of constructing machines. The reader is invited to see the excellent book by Martin Gardner [121] for the account of the "demonstrator" of Charles Stanhope and of the logic machines of Allan Marquand.

If, in the question above, the terms are appropriately changed to *knowledge*, *belief*, *time*, *tense*, and the question by "Where does the learning go when knowledge is separated from time" we can foresee how combining logics would impact philosophical logic, knowledge representation, artificial intelligence, cognitive sciences and so many other areas of interest to philosophers, linguists and computer scientists.

The use of formal logic for representing conceptual reasoning favored the emergence of the so-called "non-classical" logics in the first quarter of the XX century, as opposed to "classical" logic, usually understood as two-valued propositional logic and predicate logic with equality. The label "non-classical" is far from appropriate, since, for example, the logical properties of necessity, possibility, impossibility, contingency and related concepts were extensively treated by Aristotle (see [135]) and other ancient authors. This tradition, including nowadays the logical properties of permission, prohibition, belief, knowledge, tense, and many other evolved to what is called *modal logic*. Besides modal, there are today dozens of such "non-classical" logics such as many-valued logics, paraconsistent logics, intuitionistic logics, dependence logics and relevant logics.

If logic is objective, how can there be so many logics? This intriguing question is posed in [88], and the proposed answer, in the same book, is that what one pays attention to, in reasoning, is what determines which logic is appropriate. So logic, as a discipline, is subject-matter dependent. Classical logic is appropriate to reasoning with purely mathematical concepts, such as points, lines, sets, numbers, groups, equations and topological spaces, where no direct influence of time, intention, intensity, purpose, cause or effect is taken into account. On the other hand, the so-called non-classical logics, emerge when specialized domains are taken into consideration, and one pays attention to specific constituents.

Thus, many-valued logics may be pertinent when one needs to pay attention to degrees of truth. Paraconsistent logics may be appropriate when one needs to reason under contradictions but avoiding trivial theories. Dependence logics may be suited if one prefers to see propositions as possessing referential content. Relevant or relevance logics may be apropos if one is interested in how assumptions are related to conclusions in derivations. Modal logics may be the right choice if one is engaged on reasoning with necessity, possibility, knowledge, belief, permission,

prohibition and obligation. Intuitionistic logic may be the case if one is occupied with some aspects of constructiveness in argumentation. And so on and so forth.

Now it seems immediately obvious that reality is many-faced. A concrete question may involve several aspects that one wishes to pay attention to, and a combination of pre-existing logics to reason with such a question would be the best decision. Instead of building a new logic from scratch, it may be wiser to depart from the assumption that logics can be reusable and assembled in new and convenient manners.

However, one must face the need to integrate distinct languages and inference engines. Different families of symbols have to be merged, and sometimes the same symbol in different logics has a different meaning. Moreover, derivations can have a completely different nature in different logics.

This chapter provides an overview, aiming to anticipate, in a very simplified form, some aspects that will be pursued in full detail in the next chapters. As an appetizer served before the function, it will show the issues in lesser content.

The contents of this chapter are as follows. In Section 1.1, we present consequence systems as an abstract way to present logics via a consequence operator. We also introduce the concept of morphism to relate consequence systems. In Section 1.2, we present the basic ideas about composing or splicing logics and decomposing or splitting logics. We also introduce a technical summary of the most relevant mechanisms for splicing and splitting, namely fusion, product and fibring by functions of modal logics and Gödel provability logic. In Section 1.3, we provide an introduction to algebraic fibring from a deductive perspective. We also motivate that our aim is to define composing and decomposing mechanisms as minimal or maximal constructions. The issues of this section are discussed in detail in most of the chapters of this book. In Section 1.4, we give an introduction to the splitting mechanism called possible-translations semantics. A deeper account of this technique is explored in Chapter 9.

1.1 Consequence systems

A fundamental question, previous to any attempt to combining logics, is how to define the logics which are to be combined. There are many authors that tried to answer this question. The interested reader can take a look at [103]. Herein, a logic is a consequence system following the formulation given by Alfred Tarski (see [258]). The quest for the abstract definition of logic, as a consequence operator, seems to go back to Bernard Bolzano (see [266] and also [231] in [118]).

A consequence system is usually a pair composed by a set, the language or the set of formulas, and a map indicating for each subset of formulas the respective consequences. Some requirements are imposed on the map depending on the objectives and the properties of the logic at hand. Consequence systems can be defined in a proof-theoretic way or in a model-theoretic way.

We start with some notation. Let S be a set. We denote by $\wp S$ the set of all subsets of S and by $\wp_{\text{fin}} S$ the set of all finite subsets of S.

Definition 1.1.1 A *consequence system* is a pair

$$\langle L, \mathsf{C} \rangle$$

where L is a set and $\mathsf{C} : \wp L \to \wp L$ is a map satisfying:

- $\Gamma \subseteq \mathsf{C}(\Gamma)$ extensiveness
- if $\Gamma_1 \subseteq \Gamma_2$ then $\mathsf{C}(\Gamma_1) \subseteq \mathsf{C}(\Gamma_2)$ monotonicity
- $\mathsf{C}(\mathsf{C}(\Gamma)) \subseteq \mathsf{C}(\Gamma)$ idempotence

$\triangledown$

The set L is the language, that is, the set of formulas. The map $\mathsf{C} : \wp L \to \wp L$ is a closure operator [198] called consequence operator. For each $\Gamma \subseteq L$, $\mathsf{C}(\Gamma)$ is the set of *consequences* of the set of *hypotheses or assumptions* Γ. Extensiveness means that an hypothesis in a set is a consequence of this set. Monotonicity states that a formula that is a consequence of a set of hypotheses is also a consequence of any bigger set of hypotheses. Idempotence means that we can use lemmas to obtain consequences of a set of formulas.

An alternative characterization of the consequence operator can be given.

Proposition 1.1.2 *A map $C : \wp L \to \wp L$ is a consequence operator if and only if it satisfies the following properties:*

- $\Gamma \subseteq C(\Gamma)$ extensiveness
- $(C(\Gamma_1) \cup C(\Gamma_2)) \subseteq C(\Gamma_1 \cup \Gamma_2)$ preservation of unions
- *If* $\Gamma_2 \subseteq C(\Gamma_1)$ *and* $\Gamma_3 \subseteq C(\Gamma_2)$ *then* $\Gamma_3 \subseteq C(\Gamma_1)$ transitivity.

Proof. Let C be a consequence operator. We start by proving preservation by unions. We have that $\Gamma_1 \subseteq \Gamma_1 \cup \Gamma_2$ and $\Gamma_2 \subseteq \Gamma_1 \cup \Gamma_2$. Then $\mathsf{C}(\Gamma_1) \subseteq \mathsf{C}(\Gamma_1 \cup \Gamma_2)$ and $\mathsf{C}(\Gamma_2) \subseteq \mathsf{C}(\Gamma_1 \cup \Gamma_2)$ by monotonicity and so $\mathsf{C}(\Gamma_1) \cup \mathsf{C}(\Gamma_2) \subseteq \mathsf{C}(\Gamma_1 \cup \Gamma_2)$. Now we prove transitivity. Assume that $\Gamma_2 \subseteq \mathsf{C}(\Gamma_1)$ and $\Gamma_3 \subseteq \mathsf{C}(\Gamma_2)$. Then, $\mathsf{C}(\Gamma_2) \subseteq \mathsf{C}(\mathsf{C}(\Gamma_1))$ by monotonicity. Hence, $\Gamma_3 \subseteq \mathsf{C}(\mathsf{C}(\Gamma_1))$ and so $\Gamma_3 \subseteq \mathsf{C}(\Gamma_1)$ by idempotence.

Assume now that C satisfies extensiveness, preservation of unions and transitivity. We start by proving monotonicity. Assume that $\Gamma_1 \subseteq \Gamma_2$. Then, $\Gamma_1 \cup \Gamma_2 = \Gamma_2$. Hence, $\mathsf{C}(\Gamma_1 \cup \Gamma_2) = \mathsf{C}(\Gamma_2)$. Therefore, by preservation of unions, $\mathsf{C}(\Gamma_1) \cup \mathsf{C}(\Gamma_2) \subseteq \mathsf{C}(\Gamma_2)$ and so $\mathsf{C}(\Gamma_1) \subseteq \mathsf{C}(\Gamma_2)$. Finally, we prove idempotence. Since $\mathsf{C}(\Gamma) \subseteq \mathsf{C}(\Gamma)$ and $\mathsf{C}(\mathsf{C}(\Gamma)) \subseteq \mathsf{C}(\mathsf{C}(\Gamma))$ then $\mathsf{C}(\mathsf{C}(\Gamma)) \subseteq \mathsf{C}(\Gamma)$ by transitivity. $\triangleleft$

This presentation of the consequence operator, namely transitivity, is closer to the initial notion given by Bolzano (see [266]).

It is worthwhile to observe that we do not have in general that $\mathsf{C}(\emptyset) = \emptyset$ and that $\mathsf{C}(\Gamma_1 \cup \Gamma_2) \subseteq \mathsf{C}(\Gamma_1) \cup \mathsf{C}(\Gamma_2)$. Hence, the consequence operator is not in general a Kuratowski operator (see [160]). For more details about the relationship between logic and closure spaces see [198]. See also [280].

A consequence system is said to be *compact* or *finitary* if

$$\mathsf{C}(\Gamma) = \bigcup_{\Phi \in \wp_{\mathrm{fin}}\Gamma} \mathsf{C}(\Phi).$$

Compact consequence systems are also known as *algebraic* systems (as in [77]).

Example 1.1.3 An example of a consequence system is $\langle L_{\mathbb{P}}, \mathsf{C}\rangle$ where $L_{\mathbb{P}}$ is the set of propositional formulas over the set $\mathbb{P}$ and $\mathsf{C}(\Gamma)$ is the set of all formulas that are derived from Γ using a Hilbert calculus for classical propositional logic.

Another example of a consequence system is $\langle L'_{\mathbb{P}}, \mathsf{C}'\rangle$ where $L'_{\mathbb{P}}$ is the set of modal propositional formulas over the set $\mathbb{P}$ and $\mathsf{C}'(\Gamma)$ is the set of all formulas that are derived from Γ using a Hilbert calculus for modal propositional logic. ▽

Another characterization of consequence operators can be given. But first we prove the following auxiliary result.

Lemma 1.1.4 *Let $\langle L, C\rangle$ be a consequence system. Then*

$$C(\Gamma) = C(C(\Gamma))$$

for every $\Gamma \subseteq L$.

Proof. Use the idempotence for one side and extensiveness and monotonicity conditions for the other. ◁

We note that this property is in fact idempotence in the usual algebraic sense.

Proposition 1.1.5 *The map $C : \wp L \to \wp L$ is a consequence operator if and only if the following condition holds: (a) $\Phi \subseteq C(\Gamma)$ if and only if (b) $C(\Phi) \subseteq C(\Gamma)$, for every $\Gamma, \Phi \subseteq L$.*

Proof. Assume that $\mathsf{C} : \wp L \to \wp L$ is a consequence operator. Assume also that $\Phi \subseteq \mathsf{C}(\Gamma)$. By monotonicity $\mathsf{C}(\Phi) \subseteq \mathsf{C}(\mathsf{C}(\Gamma))$ and so $\mathsf{C}(\Phi) \subseteq \mathsf{C}(\Gamma)$ follows by idempotence. Suppose that $\mathsf{C}(\Phi) \subseteq \mathsf{C}(\Gamma)$, then using extensiveness $\Phi \subseteq \mathsf{C}(\Phi)$, we get $\Phi \subseteq \mathsf{C}(\Gamma)$.

Conversely, assume that the condition holds. We prove that C is a consequence operator. Extensiveness: from $\mathsf{C}(\Gamma) \subseteq \mathsf{C}(\Gamma)$ we get $\Gamma \subseteq \mathsf{C}(\Gamma)$ using the fact that (b) implies (a). Monotonicity: using extensiveness $\Gamma_1 \subseteq \Gamma_2 \subseteq \mathsf{C}(\Gamma_2)$ and so $\mathsf{C}(\Gamma_1) \subseteq \mathsf{C}(\Gamma_2)$ using the fact that (a) implies (b). Idempotence: from $\mathsf{C}(\Gamma) \subseteq \mathsf{C}(\Gamma)$ we get $\mathsf{C}(\mathsf{C}(\Gamma)) \subseteq \mathsf{C}(\Gamma)$, using the fact that (a) implies (b). ◁

We observe that consequence systems do not cover every possible logic that one can think. Namely, the concept leaves outside logics where sets of formulas are not considered like for instance in linear logic [125]. It seems that for covering this kind of logics one needs a more general notion.

Consequence systems can be related. We introduce a weakness relation between consequence systems.

Definition 1.1.6 The consequence system $\langle L, \mathsf{C} \rangle$ is *weaker* than consequence system $\langle L', \mathsf{C}' \rangle$, written

$$\langle L, \mathsf{C} \rangle \leq \langle L', \mathsf{C}' \rangle$$

if $L \subseteq L'$ and $\mathsf{C}(\Gamma) \subseteq \mathsf{C}'(\Gamma)$ for every subset Γ of L. ▽

We also say that a consequence system $\langle L, \mathsf{C} \rangle$ is *partially weaker* than consequence system $\langle L', \mathsf{C}' \rangle$, written

$$\langle L, \mathsf{C} \rangle \leq_p \langle L', \mathsf{C}' \rangle$$

if $L \subseteq L'$ and $\mathsf{C}(\emptyset) \subseteq \mathsf{C}'(\emptyset)$. When C is a syntactic operator, this means that all theorems of C are also theorems of C' and when C is a semantic operator, this means that all valid formulas of C are also valid formulas of C'. Of course, if $\mathsf{C} \leq \mathsf{C}'$ then $\mathsf{C} \leq_p \mathsf{C}'$ but not the other way around.

Example 1.1.7 Recall the consequence systems $\langle L_{\mathbb{P}}, \mathsf{C} \rangle$ and $\langle L'_{\mathbb{P}}, \mathsf{C}' \rangle$ presented in Example 1.1.3. We have that $\langle L_{\mathbb{P}}, \mathsf{C} \rangle \leq \langle L'_{\mathbb{P}}, \mathsf{C}' \rangle$. ▽

We can also relate consequence systems that have completely different languages using morphisms.

Definition 1.1.8 A *consequence system morphism* $h : \langle L, \mathsf{C} \rangle \to \langle L', \mathsf{C}' \rangle$ is a map $h : L \to L'$ such that

$$h(\mathsf{C}(\Gamma)) \subseteq \mathsf{C}'(h(\Gamma))$$

for every $\Gamma \subseteq L$. ▽

That is, the image of every consequence of a set of formulas is a consequence of the image of the set. We observe that the converse is not always true, namely when $h : L \to L'$ is not injective. Consequence systems and their morphisms constitute a category **Csy**.

Note that when $\langle L, \mathsf{C} \rangle \leq \langle L', \mathsf{C}' \rangle$ then there is a consequence system morphism from $\langle L, \mathsf{C} \rangle$ to $\langle L', \mathsf{C}' \rangle$ where the map from L to L' is just an inclusion.

An alternative characterization is as follows. We start by introducing some notation. Given a map $h : L \to L'$, we denote by $h^{-1}(\Gamma')$ the set of formulas $\{\gamma \in L : h(\gamma) \in \Gamma'\}$ for each $\Gamma' \subseteq L'$.

Proposition 1.1.9 *Let* $\langle L, \mathsf{C}\rangle$ *and* $\langle L', \mathsf{C}'\rangle$ *be consequence systems. A map between formulas* $h : L \to L'$ *is a consequence system morphism* $h : \langle L, \mathsf{C}\rangle \to \langle L', \mathsf{C}'\rangle$ *if and only if*

$$\mathsf{C}(h^{-1}(\Gamma')) \subseteq h^{-1}(\mathsf{C}'(\Gamma'))$$

for every $\Gamma' \subseteq L'$.

Proof. Assume that h is a consequence system morphism. Let $\varphi \in \mathsf{C}(h^{-1}(\Gamma'))$. Then $h(\varphi) \in h(\mathsf{C}(h^{-1}(\Gamma')))$ and since h is a morphism $h(\varphi) \in \mathsf{C}'(h(h^{-1}(\Gamma')))$. On the other hand, $h(h^{-1}(\Gamma')) \subseteq \Gamma'$ hence, by monotonicity, $\mathsf{C}'(h(h^{-1}(\Gamma'))) \subseteq \mathsf{C}'(\Gamma')$. Therefore, $h(\varphi) \in \mathsf{C}'(\Gamma')$ and so $\varphi \in h^{-1}(\mathsf{C}'(\Gamma'))$.
Assume now that $\mathsf{C}(h^{-1}(\Gamma')) \subseteq h^{-1}(\mathsf{C}'(\Gamma'))$ for every Γ'. Let $\varphi \in \mathsf{C}(\Gamma)$. Then $\varphi \in \mathsf{C}(h^{-1}(h(\Gamma)))$. By the hypothesis, $\mathsf{C}(h^{-1}(h(\Gamma))) \subseteq h^{-1}(\mathsf{C}'(h(\Gamma)))$, hence $\varphi \in h^{-1}(\mathsf{C}'(h(\Gamma)))$ (since $\Gamma \subseteq h^{-1}(h(\Gamma))$ and so $\mathsf{C}(\Gamma) \subseteq \mathsf{C}(h^{-1}(h(\Gamma)))$ by monotonicity), then $h(\varphi) \in h(h^{-1}(\mathsf{C}'(h(\Gamma))))$ and so $h(\varphi) \in \mathsf{C}'(h(\Gamma))$. ◁

This characterization is of course related to the notion of continuous map in topological spaces (see [160]).

It is worthwhile to say what is the union of consequence systems. We will see below that most combination mechanisms (including for instance fibring) do not correspond to the union of consequence systems.

Definition 1.1.10 Let $\{\mathsf{C}_i\}_{i\in I}$, where $\mathcal{C}_i = \langle L_i, \mathsf{C}_i\rangle$, be a family of consequence systems. Their *union* is the following consequence system:

$$\mathcal{C} = \langle \bigcup_{i\in I} L_i, \mathsf{C}\rangle$$

where $\mathsf{C}(\bigcup_{i\in I}\Gamma_i)$ is $\bigcup_{i\in I}\mathsf{C}_i(\Gamma_i)$. ▽

Instead of presenting consequence via an operator we can look at consequence as a binary relation between the set of all sets of formulas and the set of formulas. Given a binary relation $S \subseteq A \times B$, we may write aSb whenever $\langle a, b\rangle \in S$.

Then, a *consequence relation* over L is a set

$$\mathsf{R} \subseteq \wp L \times L$$

satisfying the following postulates: (i) if $\varphi \in \Gamma$ then $\Gamma\mathsf{R}\varphi$; (ii) if $\Psi\mathsf{R}\varphi$ e $\Gamma\mathsf{R}\psi$ for every $\psi \in \Psi$ then $\Gamma\mathsf{R}\varphi$; (iii) if $\Gamma_1\mathsf{R}\varphi$ and $\Gamma_1 \subseteq \Gamma_2$ then $\Gamma_2\mathsf{R}\varphi$. A consequence operator $\mathsf{C} : \wp L \to \wp L$ induces a consequence relation R_C such that, for every $\Gamma \subseteq L$ and every $\varphi \in L$,

$$\Gamma\mathsf{R}_\mathsf{C}\varphi \text{ if and only if } \varphi \in \mathsf{C}(\Gamma).$$

A consequence relation R induces a consequence operator C_R such that, for every $\Gamma \subseteq L$,

$$\mathsf{C}_\mathsf{R}(\Gamma) = \{\varphi \in L \ : \ \Gamma\mathsf{R}\varphi\}.$$

It is worth noting that the map $\mathsf{C} \to \mathsf{R}_{\mathsf{C}}$ is the inverse of the map $\mathsf{R} \to \mathsf{C}_{\mathsf{R}}$, and vice-versa. Thus, a consequence system can be defined indistinctly as a pair $\langle L, \mathsf{C} \rangle$ such that C is a consequence operator over L, or as a pair $\langle L, \mathsf{R} \rangle$ such that R is a consequence relation over L. Note that $\langle L, \mathsf{R} \rangle$ is weaker than $\langle L', \mathsf{R}' \rangle$ if and only if $L \subseteq L'$ and $\mathsf{R} \subseteq \mathsf{R}'$.

Sometimes we may write Γ^R instead of $\mathsf{C}_{\mathsf{R}}(\Gamma)$.

For the purpose of most combination mechanisms, the consequence systems are such that their language is always generated from a family of connectives. Hence, we do not include L in the definition of a consequence system, but instead we use a signature C defining the family of connectives in each case. As an illustration we define propositional based signatures and the induced languages.

Definition 1.1.11 A *propositional based signature* is any family of sets

$$C = \{C_k\}_{k \in \mathbb{N}}$$

such that $C_i \cap C_j = \emptyset$ if $i \neq j$. ▽

The elements of the set C_k are called *k-ary connectives* or connectives of *arity* k. In particular, the elements of C_0 are called *constants*.

We will consider unions of propositional signatures. Given propositional signatures C' and C'', their union is the signature

$$C' \cup C''$$

where $(C' \cup C'')_k = C'_k \cup C''_k$ for each $k \in \mathbb{N}$. We use $C' \setminus C''$ to denote the signature such that $(C' \setminus C'')_k = C'_k \setminus C''_k$ for each $k \in \mathbb{N}$.

Given two signatures C e C', we say that C *is contained in* C', denoted by

$$C \leq C'$$

if $C_k \subseteq C'_k$, for every $k \in \mathbb{N}$. In some situations, we would like to include a set $\mathbb{P}$ of propositional constants. Then we would say that $\mathbb{P} \subseteq C_0$.

We assume that Ξ is a set of schema variables. Schema variables play an important role when combining logics, in particular for deduction, as we explain in Chapter 2.

Definition 1.1.12 Let $C = \{C_k\}_{k \in \mathbb{N}}$ be a signature. The propositional based *language generated by* C is the set $L(C)$ defined as the least set $L(C)$ that satisfies the following properties:

- $\Xi \subseteq L(C)$;
- $(c(\varphi_1, \ldots, \varphi_k)) \in L(C)$ whenever $k \in \mathbb{N}$, $c \in C_k$ and $\varphi_1, \ldots, \varphi_k \in L(C)$. ▽

The elements of $L(C)$ are called *formulas* over C. In particular, $C_0 \subseteq L(C)$.

Typical deductive systems such as Hilbert, tableau, sequent and natural deduction calculi induce consequence systems. We can look at each kind of deductive system as a presentation of a consequence system. Observe that it is quite common to work with these presentations instead of working with the consequence systems themselves. That is the case of most chapters in this book.

We illustrate how a Hilbert calculus induce a consequence system. We need to define Hilbert calculus, substitution and derivation.

Definition 1.1.13 A *Hilbert calculus* is a pair

$$H = \langle C, R \rangle$$

such that:

- C is a signature;
- R is a set inference rules, that is, a set of pairs $\langle \Delta, \psi \rangle$ where $\Delta \subseteq L(C)$ is a finite set and $\psi \in L(C)$. ▽

When $\Delta = \emptyset$ we say that the inference rule is an axiom. Otherwise it is said to be a rule. Sometimes when introducing axioms we may, for simplicity, indicate only the formula. We now define the notion of derivation in a Hilbert calculus. Before we have to introduce the notion of substitution.

The objective of a substitution is to replace schema variables by formulas. A *substitution* is a map

$$\sigma : \Xi \to L(C).$$

Substitutions can be extended to formulas in a natural way. We denote by $\sigma(\varphi)$ the formula that results from substituting each schema variable ξ in φ by $\sigma(\xi)$. Moreover, substitutions can be extended to sets of formulas. We denote by $\sigma(\Gamma)$ the set of formulas $\{\sigma(\gamma) : \gamma \in \Gamma\}$.

Definition 1.1.14 A *derivation* in H from a set $\Gamma \subseteq L(C)$ is a sequence

$$\varphi_1 \dots \varphi_n$$

such that for $i = 1, \dots, n$ each φ_i is either an element of Γ or there is a substitution σ and an inference rule $\langle \Delta, \psi \rangle$ in H such that $\sigma(\psi)$ is φ_i and $\sigma(\delta)$ is φ_j for some $j < i$, for every $\delta \in \Delta$.

We also say that φ_n is *derived from* Γ and use the following notation

$$\Gamma \vdash_H \varphi_n.$$

▽

A Hilbert calculus H induces a consequence system

$$\mathcal{C}(H) = \langle C, \vdash_H \rangle$$

where, for each $\Gamma \subseteq L(C)$, $\Gamma^{\vdash_H}$ is the set $\{\varphi \in L(C) : \Gamma \vdash_H \varphi\}$. Observe that this consequence system is compact. It is also structural in the following sense.

A consequence system is said to be *structural* if, for every substitution σ we have that:

$$\sigma(\mathsf{C}(\Gamma)) \subseteq \mathsf{C}(\sigma(\Gamma)).$$

That is, if a consequence system $\mathcal{C}$ is structural, then σ is a consequence system endomorphism (from $\mathcal{C}$ to $\mathcal{C}$).

Consequence systems that are compact and structural are called *standard* in the terminology of [280]. The notions of compact, structural and standard consequence systems can be expressed in terms of consequence relations in the obvious way.

Similarly, semantic entailments are also consequence operators. For instance, propositional entailment associated with valuations and modal entailment associated to Kripke structures are examples of consequence operators. Semantic structures can also be seen as presentations of the semantic entailment.

Consequence operators are relevant in all chapters of the book. In Chapter 2, we introduce a Hilbert consequence operator generated by a Hilbert calculus. In Chapter 3, we introduce a semantic consequence operator. We also introduce the notion of soundness, as saying that the set of Hilbert consequences is included in the set of semantic consequences, and the notion of completeness, as stating that the set of semantic consequences is included in the set of Hilbert consequences. In Chapter 5, consequence systems are used in another way. When considering logics presented in a different way, say one with a Hilbert calculus and the other via a sequent calculus, we can define their fibring by fibring the induced consequence systems. We will also define other relations between consequence systems in Chapter 9.

1.2 Splicing and splitting

To start with, it is convenient to keep in mind that, in order to combine logics, we intend to depart from simple logics to obtain a more complex one. Following the terminology of [47], this process, by which a bunch of logics is synthesized forming a new logic, is called *splicing logics*. A prototypical case of splicing is the method of *fibring*, introduced in [104] (see also [108]). On the other hand, we may think about an analytic procedure that permits us to decompose a given logic into simpler components. This kind of process was called *splitting* in [47]. A prototypical case of splitting occurs when one succeeds in describing a given logic in terms of simpler components by means of translating the original logic into a collection of simpler, auxiliary logics, using what is called *possible-translations semantics*. This mechanism is introduced in [45] and further developed in [196] and [46].

We may thus consider two complementary approaches in the field of combining logic systems:

- Splicing, combining or composing logics: a bottom–up, synthetic approach presented in Figure 1.1. There are several methods to combine logics. Each method can be seen as an operation on some class of logics. In the figure, we start with logics $\mathcal{L}_1$ and $\mathcal{L}_2$ in some class of logics, and using the binary operation $\odot$ we obtain a new logic $\mathcal{L} = \mathcal{L}_1 \odot \mathcal{L}_2$.

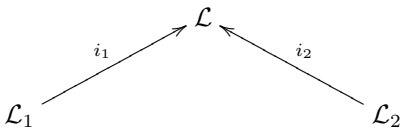

Figure 1.1: $\mathcal{L}$ is synthesized from $\mathcal{L}_1$ and $\mathcal{L}_2$

  The arrows i_1 and i_2 indicate that the component logics should be related with their combination. In general, these arrows induce consequence system morphisms from the component logics to their combination, meaning that derivation and entailment are preserved.

- Splitting, decombining or decomposing logics: a top–down, analytic approach presented in Figure 1.2. In this case we start with logic $\mathcal{L}$ and try to find an operation $\odot$ and components $\mathcal{L}_1$ and $\mathcal{L}_2$ such that $\mathcal{L} = \mathcal{L}_1 \odot \mathcal{L}_2$. The arrows f_1 and f_2 indicate that the given logic should be related with the components. In general, these arrows induce consequence system morphisms from the given logic to the components.

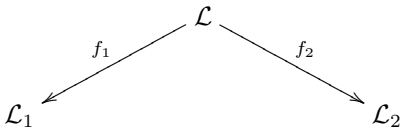

Figure 1.2: $\mathcal{L}$ is analyzed into $\mathcal{L}_1$ and $\mathcal{L}_2$

Splicing can be applied to more than two logics. It is worthwhile noting that splitting can involve an infinite number of components as we discuss below.

The known splicing mechanisms, as we will see throughout the book, have an important property. Once we choose a mechanism and the components $\mathcal{L}_1$ and $\mathcal{L}_2$, the resulting logic $\mathcal{L}$ is, in most cases, immediately defined. On the other hand, the known splitting mechanisms are different in this respect. Once we choose a mechanism and a logic $\mathcal{L}$, one can have several possibilities of choosing the components.

The splicing and the splitting mechanisms can be combined as we explain in Chapter 11 where we briefly discuss some applications. There we will discuss how one can use both of them if there is need to do so.

Among the most challenging problems in combination of logics we can refer to preservation of properties. For example, it is interesting to investigate if a logic resulting from a combination has a certain property, assuming that the original logics have that property. In many cases, one has to impose sufficient conditions for the preservation. The most relevant preservation results are related to soundness and completeness, interpolation and decidability.

Several combination mechanisms have been investigated, namely fusion of modal logics, product of modal logics, fibring by functions of modal logics, algebraic fibring, temporalization and parameterization, synchronization, institutions and parchments. They are targeted to different classes of logics and assume different degrees of abstraction. They all have in common that, in the end, we are dealing with consequence systems either with a syntactic or with a semantic nature. That is, in all the cases there are consequence operators that are extensive, monotonic and idempotent.

In this section, we only briefly describe three combination mechanisms: fusion, product and fibring by functions of modal logics. They will be used as examples in other chapters. For a complete description of fusion and product we refer to [113] and for a in depth treatment of fibring by functions see [108]. Algebraic fibring will be discussed in many guises throughout the book. The other combination mechanisms were triggered by applications in software specification and will be discussed in Chapter 11.

1.2.1 Fusion of modal logics

Fusion of normal modal logics (see [259]) is a binary operation on the class of normal modal logics endowed with Kripke semantics (introduced by Samuel Kripke in [174]) and Hilbert calculi, that we now explain with some detail. Recall that a Kripke structure is a triple

$$\langle W, R, V \rangle$$

where W is a non-empty set (the set of worlds), $R \subseteq W^2$ is a binary relation (the accessibility relation) and $V : \mathbb{P} \to \wp W$ is a map (the valuation).

Consider two normal modal logics $\mathcal{L}'$ and $\mathcal{L}''$ with the following characteristics:

- both have the same set $\mathbb{P}$ of zero-ary connectives (propositional constants), a unary connective $\neg$ and a binary connective $\Rightarrow$;
- a unary connective $\Box'$ and $\Box''$ for the logics $\mathcal{L}'$ and $\mathcal{L}''$, respectively;
- M' and M'' are classes of Kripke structures for $\mathcal{L}'$ and $\mathcal{L}''$, respectively;
- the Hilbert calculi H' and H'' include, besides the propositional part, the following axioms and rules:

- $\langle \emptyset, ((\Box'(\xi_1 \Rightarrow \xi_2)) \Rightarrow ((\Box'\xi_1) \Rightarrow (\Box'\xi_2)))\rangle$ K axiom for $\mathcal{L}'$;
- $\langle \emptyset, ((\Box''(\xi_1 \Rightarrow \xi_2)) \Rightarrow ((\Box''\xi_1) \Rightarrow (\Box''\xi_2)))\rangle$ K axiom for $\mathcal{L}''$;
- $\langle \{\xi\}, (\Box'\xi)\rangle$ necessitation rule for $\mathcal{L}'$;
- $\langle \{\xi\}, (\Box''\xi)\rangle$ necessitation rule for $\mathcal{L}''$.

The fusion of $\mathcal{L}'$ and $\mathcal{L}''$ is a normal bimodal logic $\mathcal{L}$ with two boxes that behave independently, except when otherwise imposed. That is, $\mathcal{L}$ is characterized as follows:

- a set $\mathbb{P}$ of zero-ary connectives (propositional constants), a unary connective $\neg$, a binary connective $\Rightarrow$ and two unary connectives $\Box'$ and $\Box''$;
- M is the class of all Kripke structures of the form $\langle W, R', R'', V\rangle$ where $\langle W, R', V\rangle$ and $\langle W, R'', V\rangle$ are Kripke structures of $\mathcal{L}'$ and $\mathcal{L}''$, respectively;
- the Hilbert calculus H includes all the rules of the Hilbert calculi of the original logics and hence the following ones:
  - $\langle \emptyset, ((\Box'(\xi_1 \Rightarrow \xi_2)) \Rightarrow ((\Box'\xi_1) \Rightarrow (\Box'\xi_2)))\rangle$ K axiom for $\mathcal{L}'$;
  - $\langle \emptyset, ((\Box''(\xi_1 \Rightarrow \xi_2)) \Rightarrow ((\Box''\xi_1) \Rightarrow (\Box''\xi_2)))\rangle$ K axiom for $\mathcal{L}''$;
  - $\langle \{\xi\}, (\Box'\xi)\rangle$ necessitation rule for $\mathcal{L}'$;
  - $\langle \{\xi\}, (\Box''\xi)\rangle$ necessitation rule for $\mathcal{L}''$.

Each model of the fusion corresponds to a model $\langle W, R', V'\rangle$ in $\mathcal{L}'$ and to a model $\langle W, R'', V''\rangle$ in $\mathcal{L}''$. That is, in the fusion we do not include models where the set W (of worlds) is different. In technical words, each model of the fusion should have as a reduct a model of $\mathcal{L}'$ and a model of $\mathcal{L}''$.

We briefly describe how a formula in the fusion is evaluated. Given the model $\langle W, R', R'', V\rangle$ in the fusion and $w \in W$ we have that the formula $(\Diamond'(\Box''p))$ is satisfied by $\langle W, R', R'', V\rangle$ at w, denoted by

$$\langle W, R', R'', V\rangle, w \Vdash (\Diamond'(\Box''p))$$

if there is $z \in W$ such that:

- $\langle W, R', R'', V\rangle, z \Vdash (\Box''p)$ and $wR'z$;
- $N_z \subseteq V(p)$ where $N_z = \{u \in W : zR''u\}$.

We refer to Figure 1.3 for details, where φ'_p is $(\Diamond'(\Box''p))$ and φ''_p is $(\Box''p)$.

Observe that the formula

$$(\Box''((\Box'(\xi_1 \Rightarrow \xi_2)) \Rightarrow ((\Box'\xi_1) \Rightarrow (\Box'\xi_2))))$$

is a theorem of the fusion. That is, we can derive this formulas from the K axiom for $\mathcal{L}'$ and the necessitation rule for $\mathcal{L}''$.

We synthesize the properties of fusion in the following way:

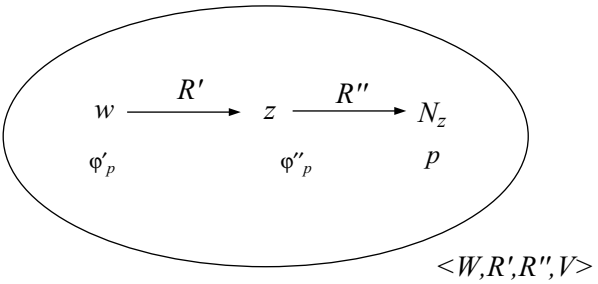

Figure 1.3: Evaluating the formula $(\Diamond'(\Box''p))$ in a fusion structure

- *homogeneous combination mechanism at the deductive level*: both original logics are presented by Hilbert calculi;
- *homogeneous combination mechanism at the semantic level*: both original logics are presented by Kripke structures;
- *algorithmic combination of logics at the deductive level*: given the Hilbert calculi for the original logics, we know how to define the Hilbert calculus for the fusion;
- *algorithmic combination of logics at the semantic level*: given the classes of Kripke structures for the original logics, we know how to define the class of Kripke structures for the fusion.

The algorithmic nature of fusion also means that no interaction is stated between $\Box'$ and $\Box''$. We will see in Chapter 2 and in Chapter 3 that fusion is a canonical construction in the sense that it is minimal in some class of logics.

It is easy to conclude that the definition of $\mathcal{L}'$ above induces a consequence system

$$\mathcal{C}(H') = \langle L(C'), \vdash_{H'} \rangle$$

where C' is the signature of $\mathcal{L}'$ and $(\Gamma')^{\vdash_{H'}}$ is the set of formulas that can be derived from Γ', using the Hilbert calculus for $\mathcal{L}'$. In a similar way, we can define $\mathcal{C}''(H'') = \langle L(C''), \vdash_{H''} \rangle$ and $\mathcal{C}(H) = \langle L(C' \cup C''), \vdash_H \rangle$. Then we have:

$$\mathcal{C}(H') \leq \mathcal{C}(H) \text{ and } \mathcal{C}(H'') \leq \mathcal{C}(H).$$

The semantic characterizations of $\mathcal{L}'$, $\mathcal{L}''$ and $\mathcal{L}$ also induce consequence systems. Again, the consequence systems for $\mathcal{L}'$ and $\mathcal{L}''$ are weaker than the one for $\mathcal{L}$.

At first sight, this may seem a very simple combination mechanism. However, it is interesting enough for seeing that preservation of properties is not an easy issue. An example of a preservation problem can be presented in the following way. Assume that $\mathcal{L}'$ and $\mathcal{L}''$ are weakly complete logics with respect to the class of

frames (that is, every valid formula is a theorem). Is the logic $\mathcal{L}$ resulting from the fusion also weakly complete with respect to the class of fusion frames? In fusion, there are preservation results for weak completeness, uniform Craig interpolation (for theoremhood) and decidability (see [281, 169]).

Fusion of non-normal modal logics was also investigated in [92], namely discussing preservation of weak completeness via a technique that extends the one used in the normal case.

It is worthwhile noting that there is no notion of fusion of a normal modal logic with a non-normal modal logic. Such a combination can, however, be defined in the context of algebraic fibring.

It is also worthwhile mentioning that the interested reader should also consult [99] where the notion of fusion was anticipated through some examples of combining alethic and deontic logics with philosophical interest.

1.2.2 Product of modal logics

We now concentrate on another mechanism of combination: the product of logics. The product of modal logics is a binary operation that is very useful when one wants, for example, to represent time-space information. Products were introduced in [235, 236]. We consider the same setting as we did for fusion of modal logics. The signature and the semantic counterparts of the product of $\mathcal{L}'$ and $\mathcal{L}''$ is as follows:

- a set $\mathbb{P}$ of zero-ary connectives (propositional constants), a unary connective $\neg$, a binary connective $\Rightarrow$ and two unary connectives $\Box'$ and $\Box''$;
- M is the class of product structures of the form

$$\langle W' \times W'', \overline{R}', \overline{R}'', V' \times V'' \rangle$$

  where $\langle W, R', V' \rangle$ and $\langle W, R'', V'' \rangle$ are Kripke structures of $\mathcal{L}'$ and $\mathcal{L}''$, respectively and where $\overline{R}', \overline{R}'' \subseteq (W' \times W'')^2$ are defined as follows:

  - $\langle w'_1, w'' \rangle \overline{R}' \langle w'_2, w'' \rangle$ if $w'_1 R' w'_2$;
  - $\langle w', w''_1 \rangle \overline{R}'' \langle w', w''_2 \rangle$ if $w''_1 R'' w''_2$;
  - $(V' \times V'')(p) = V'(p) \times V''(p)$.

The striking aspect about products is that some modal formulas are valid in every product frame. Namely the following will show that some interaction exists between $\Box'$ and $\Box''$ and $\Diamond'$ and $\Diamond''$ (recall that $\Diamond'\varphi$ is an abbreviation of $(\neg(\Box'(\neg\varphi)))$ and similarly with respect to $\Diamond''$):

- $((\Diamond'(\Diamond'' p)) \Rightarrow (\Diamond''(\Diamond' p)))$ commutativity 1;
- $((\Diamond''(\Diamond' p)) \Rightarrow (\Diamond'(\Diamond'' p)))$ commutativity 2;