# Some Contributions to the Algebraic Theory of Automata

**Enric Cosme i Llópez**

Supervised by

**Adolfo Ballester-Bolinches**
**and Jean-Éric Pin**

Facultat de Ciències Matemàtiques
Universitat de València

This dissertation is submitted for the degree of
*International Doctor in Mathematics*

September 2015

*als meus pares*
*i a la meua germana*

*a Toni de l'Hostal*

# Agraïments

Els agraïments són complicats quan tens tant a agraïr. Més encara si és la part de la tesi que més expectatives sol endur-se i que amb més entusiasme es llig la gent. Representem la resta de la tesi com una meta elevada i no ho és tant; no és cap obra suprema, ni principi ni final de res. És treball fet, voluntats escrites i riuades de sentiments contraposats. Només cal pensar-la com un viatge honest cap endavant, cap al que anomenem ciència i que no deixa de ser una activitat humana, que beu de les nostres virtuts i dels nostres vicis. Per això són importants els agraïments, perquè és l'únic lloc on no es parla del que has fet, sinó de la gent que has conegut pel camí i que t'ha ajudat a arribar fins al final. La tesi, al cap i a la fi, sempre és allò que acompanya els agraïments. Qui ho negue menteix.

Agraïsc, així, tot el que he aprés del meu tutor, Adolfo Ballester. Agraïsc, la confiança que ha tingut en aquest projecte, la seua amistat, estima i generositat. A ell em vaig acostar per dir-li que volia fer autòmats i sense pegar-li moltes voltes mamprenguèrem este viatge. Pensat i fet. Com millor solen eixir les coses. Gràcies pels seus consells i les batalles quotidianes, per apretar i afluixar quan tocava. Ha sigut tot un privilegi vore el geni en marxa. Que siga així per molts anys! Gràcies també a Jean-Éric Pin per les orientacions i indicacions donades, per l'ajuda inestimable i per haver guiat aquest projecte i aportar tot el seu coneixement a l'elaboració dels treballs. *Merci beaucoup!*

A Jan Rutten per haver contribuït amb la seua gran capacitat, humana i professional, als treballs que ací es presenten. Trobar-se gent com ell alleugera el viatge i ho fa tot una miqueta més fàcil. Gràcies per la proximitat i confiança rebuda, per l'espenta que sap imprimir i per la cura amb què tracta els alumnes. *Heel hartelijk bedankt!* Gràcies també a la resta de la gent del CWI per fer-me sentir part d'aquell centre, per l'ambient de treball i les ganes de treballar que allí em vaig trobar. Gràcies a Marcello, Alexandra, Julian i Henning per la seua ajuda i comprensió. *Grazie mille! Muito obrigado! Muchas gracias! Vielen Dank!*

Gràcies al professor John Cossey, per haver-nos acollit a Austràlia i haver-nos ajudat a batallar amb semigrups, monoides i autòmats. Per la seua increïble capacitat per a escoltar i proposar idees. *Thank you so much!*

A la gent de Bogotà, per l'experiència tan gratificant que allí vaig viure. A *los pelaos juiciosos* de la Distrital per ser un exemple d'esforç i dedicació, per ensenyar-me també a valorar les matemàtiques com a motor de canvi, pel tracte proper i per descobrir-me el café. A la professora Verónica Cifuentes per l'acolliment que em va donar i la seua amistat.

Des d'ací una abraçada a la professora Paz Jiménez per l'ajuda que m'ha donat, per llegir-

se els meus apunts i per la col·laboració iniciada. Per les hores dedicades i per la seua forma de ser, pel treball que ja està en marxa i pel que ha de vindre. Gràcies també a la gent que m'ha acollit a Saragossa i m'ha fet sentir com a casa. Un record especial per a Javier Otal i José María Muñoz. *Muchas gracias! Muitas grazias!* Gràcies també a la gent de Navarra. A Immaculada Lizasoain i a Luis M. Ezquerro per la seua col·laboració i per acollir-me a Pamplona. *Muchas gracias! Eskerrik asko!*

A Ramon Esteban, gran investigador i millor persona, per la seua inestimable entrega i ajuda. Des d'ací un sentit homenatge a la seua capacitat per a treballar i emprendre. Per la seua amistat i proximitat. A Ramon li haurien de fer una rotonda a l'entrada del Campus! Amb llums i tot! Gràcies als seus consells i creativitat he aprés moltes coses útils i profitoses.

Vull donar també les gràcies a la resta dels membres del grup d'investigació, especialment a Francesca i Vicent, per l'ajuda rebuda i pels grans moments que hem compartit. Sense vosaltres haguera sigut més difícil i més avorrit. Gràcies també a tots els membres del Departament d'Àlgebra, especialment a Carolina, Joan i Azahara amb qui he compartit despatx, treball i hores de desfici. Tinc també un record agraït a Fran pels seus consells i ajuda.

Agraïsc a tots aquells que m'han contagiat la seua passió per les matemàtiques; a Gaspar i Bernardo, des del Serra Perenxisa, i a tot el professorat que m'he trobat a la Facultat de Matemàtiques, amb gran record per a Rafa, Francisca i els seus consells. Amb especial estima a Juan Climent per la seua amistat i per imprimir-li humanitat a la lògica i a totes les coses que fa. Per l'ajuda que m'ha oferit en aquests anys. Gràcies a ell vaig fer el camí de la lògica, per a continuar estudiant tot el que ell ens havia descobert. Al professorat de Barcelona, per descobrir-me un poc més la bellesa d'aquell món. Amb un record especial per a Ramon Jansana, per ser la primera persona a qui li vaig escoltar dir *coàlgebra.*

A tots els que han cregut en el projecte dels Predoc, una excusa com una altra per a juntar-nos tots, fer-nos visibles i moure la Facultat. A tots els Predoc, per la seua vitalitat i ganes de festa. No em puc queixar de res. A Juan Monterde i Rafa Crespo per la seua ajuda en aquest i molts altres més projectes.

Gràcies al Ministeri d'Economia i Competitivitat per l'ajuda econòmica rebuda mitjançant la concesió de la beca FPU. Gràcies igualment al CWI per l'ajuda rebuda per a l'estància a Àmsterdam. A la Universidad de Zaragoza, Universidad Pública de Navarra, Universidad Francisco José de Caldas, University of Canberra i Australian National University pel suport econòmic.

A Abel i Raffa —amics, artistes, mestres, templats, aseats, bonicos, treballadors— per les cerveses fetes! Als bufardos dels meus amics per les batalletes que portem darrere i per l'amistat que dura tants anys. A Carme, companya, per ser un suport essencial en aquests anys. Gràcies als meus pares per estimar-me tant, per ensenyar-me tant. A la meua família, als que estan i als que se n'han anat, pel recolçament que m'han donat sempre. A Torrent i a Alaquàs, els meus pobles.

Arre haca, anem al camp,
que no eł cànvie per un mestre!
Arre haca, anem al camp,
que no cànvie ês llibres per l'aixâ!

A carro i haca — Les Mãedéus

# Resum

En el present treball estudiarem els autòmats des d'una perspectiva tant algebraica com coalgebraica. Volem aprofitar la natura dual d'aquests objectes per a presentar un marc unificador que explique i estenga alguns resultats recents de la teoria d'autòmats.

Per tant, la secció 2 conté nocions i definicions preliminars per a mantenir el treball tan contingut com siga possible. Així, presentarem les nocions d'àlgebra i coàlgebra per a un endofunctor. També introduirem alguns conceptes sobre monoides i llenguatges. En aquest capítol també exposarem les nocions d'autòmats deterministes i no deterministes, homomorfismes i bisimulacions d'autòmats i productes i coproductes d'aquestes estructures. Finalment, recordarem algunes nocions bàsiques de teoria de reticles.

Des d'una perspectiva algebraica, els autòmats són àlgebres amb operacions unàries. En aquest context, una equació és simplement un parell de paraules. Direm que una equació és satisfeta per un autòmat si per a cada estat inicial possible els estats als quals s'arriba des de l'estat considerat sota l'acció de les dues paraules coincideix. Es pot provar que, per a un autòmat donat, podem construir el major conjunt d'equacions que aquest satisfà. Aquest conjunt d'equacions resulta ser una congruència en el monoide lliure associat a l'alfabet d'entrada i ens permet definir l'autòmat lliure, denotat per $\mathsf{free}$. Pel que respecta a la perspectiva coalgebraica, un autòmat és un sistema de transicions amb estats finals. Així, una coequació és un conjunt de llenguatges. Direm que una coequació és satisfeta per un autòmat, si per a cada observació possible (coloracions sobre els estats indicant-ne la finalitat o no), el llenguatge acceptat per l'autòmat es troba dins la coequació considerada. Intuïtivament, les coequacions poden ser pensades com comportaments o especificacions en el disseny que se suposa que una coàlgebra deu tindre. Com hem fet abans, per a un autòmat donat, podem construir el menor conjunt de coequacions que aquest satisfà. Aquest conjunt de coequacions resulta ser un subconjunt amb característiques ben determinades del conjunt de tots els llenguatges associats a l'alfabet d'entrada i ens permet definir l'autòmat colliure, denotat per $\mathsf{cofree}$. Provem, a més, que aquestes construccions basades en equacions i coequacions són functorials.

Al capítol 3 hem establert un nou resultat que presenta la dualitat entre *quocients de congruència* del monoide lliure i el seu conjunt de coequacions, que són àlgebres booleanes completes i atòmiques tancades sota derivació i que hem anomenat *preformacions de llenguatges*. Aquesta dualitat no imposa cap restricció en la grandària dels objectes, per tant, també s'aplica a objectes infinits. El capítol 3 està basat en els següents articles:

- [65] J.J.M.M. Rutten, A. Ballester-Bolinches, and E. Cosme-Llópez. **Varieties and covarieties of languages (preliminary version)**. In D. Kozen and M. Mislove, editors, Proceedings of MFPS XXIX, volume 298 of *Electron. Notes Theor. Comput. Sci.*, pages 7–28, 2013.

- [14] A. Ballester-Bolinches, E. Cosme-Llópez, and J. Rutten. **The dual equivalence of equations and coequations for automata**. *Information and Computation*, 244:49 – 75, 2015.

Aquesta dualitat és emprada en el capítol 4 per a presentar un nou apropament al teorema de varietats d'Eilenberg. En primer lloc presentem una descripció equivalent, basada en equacions i coequacions, de la noció original de varietat de llenguatges d'Eilenberg. Aquesta nova descripció és un dels millors exemples possibles del poder expressiu del functors free i cofree. Una adaptació adient d'aquestes construccions permet presentar un resultat de tipus Eilenberg per a formacions de monoides no necessàriament finits. En el nostre cas, primerament provem que les formacions de monoides estan en correspondència biunívoca amb les formacions de congruències. Un segon pas en la prova relaciona formacions de congruències amb formacions de llenguatges. Així, provem que tots tres conceptes són equivalents

| | | | | |
|---|---|---|---|---|
| Formacions<br>de monoides | ⇔ | Formacions<br>de congruències | ⇔ | Formacions<br>de llenguatges |

La primera correspondència pareix ser completament nova i relaciona formacions de monoides amb filtres de congruències per a cada monoide. L'última correspondència és un dels millors exemples on poder aplicar la dualitat presentada al capítol 3. A més, donem una aplicació d'aquestes equivalències per al cas dels llenguatges relativament disjuntius. Aquests teoremes poden ser adequadament modificats per a cobrir el cas de les varietats de monoides en el sentit de Birkhoff. Discutim aquest cas particular al final del capítol 4. Els resultats d'aquest capítol han estat enviats per a la seua possible publicació en una revista científica sota el títol

- [13] A. Ballester-Bolinches, E. Cosme-Llópez, R. Esteban-Romero, and J. Rutten. **Formations of monoids, congruences, and formal languages**. 2015. Enviat.

El capítol 5 està completament dedicat a l'estudi de l'objecte final associat als autòmats no deterministes. En general, les tècniques emprades en el capítol 5 difereixen de les presentades en els capítols 3 i 4. En conseqüència, al principi d'aquest capítol introduïm alguns conceptes preliminars sobre bisimulacions i objectes finals. El nostre resultat principal és presentat en el Teorema 5.17, que descriu l'autòmat final no determinista amb l'ajuda d'estructures basades en llenguatges. A continuació, relacionem altres descripcions de l'autòmat final no determinista amb la nostra construcció. El capítol 5 està basat en el següent article:

- [11] A. Ballester-Bolinches, E. Cosme-Llópez, and R. Esteban-Romero. **A description based on languages of the final non-deterministic automaton.** *Theor. Comput. Sci.*, 536(0):1 – 20, 2014.

Certament, els diferents punts de vista emprats en aquesta dissertació ja han estat explorats en alguns altres treballs. Per això, al final de cada capítol presentem un estudi detallat dels treballs relacionats i discutim les aportacions o millores realitzades en els resultats existents. Finalment, el capítol 6 presenta les conclusions i indica els treballs que caldrà realitzar en el futur. També presentem alguns del articles de recerca que es deriven de la realització d'aquest projecte.

La vida mai nostra.
Embriagadors els vins,
que bevem quan fugim
de la primera impressió.

Els vespres verds — Mishima

# Resumen

En el presente trabajo estudiaremos los autómatas desde una perspectiva tanto algebraica como coalgebraica. Queremos aprovechar la naturaza dual de estos objetos para presentar un marco unificador que explique y extienda algunos resultados recientes de la teoría de autómatas.

Por tanto, la sección 2 contiene nociones y definiciones preliminares para mantener el trabajo tan contenido como sea posible. Así, presentamos las nociones de álgebra y coálgebra para un endofuntor. También introducimos algunos conceptos sobre monoides y lenguajes. En este capítulo también introduciremos las nociones de autómatas deterministas y no deterministas, homomorfismos y bisimulaciones de autómatas y productos y coproductos de estas estructuras. Finalmente, recordaremos algunas nociones básicas de la teoría de retículos.

Desde una perspectiva algebraica, los autómatas son álgebras con operaciones unarias. En este contexto, una ecuación es simplemente un par de palabras. Diremos que una ecuación se satisface en un autómata si para cada estado inicial posible los estados alcanzados desde el estado considerado bajo la acción de las dos palabras coincide. Se puede probar que, para un autómata dado, podemos construir el mayor conjunto de ecuaciones que éste satisface. Este conjunto de ecuaciones resulta ser una congruencia en el monoide libre asociado al alfabeto de entrada y nos permite definir el autómata libre, denotado por free. Por lo que respecta a la perspectiva coalgebraica, un autómata es un sistema de transiciones con estados finales. Así, una coecuación es un conjunto de lenguajes. Diremos que una coecuación se satisface en un autómata, si para cada observación posible (coloraciones sobre los estados indicando la finalidad o no), el lenguaje aceptado por el autómata se encuentra dentro de la coecuación considerada. Intuitivamente, las coecuaciones pueden ser pensadas como comportamientos o especificaciones en el diseño que se supone que una coálgebra debe tener. Como hemos hecho antes, para un autómata dado, podemos construir el menor conjunto de coecuaciones que éste satisface. Este conjunto de coecuaciones resulta ser un subconjunto con características bien determinadas del conjunto de todos los lenguajes asociados al alfabeto de entrada y nos permite definir el autómata colibre, denotado por cofree. Probamos, además, que estas construcciones basadas en ecuaciones y coecuaciones son funtoriales.

En el capítulo 3 hemos establecido un nuevo resultado que presenta la dualidad entre *cocientes de congruencia* del monoide libre y su conjunto de coecuaciones, que son álgebras booleanas completas y atómicas cerradas bajo derivación y que hemos llamado *preformaciones de lenguajes.* Esta dualidad no impone ninguna restricción en el tamaño de los objetos, por

tanto, también se aplica a objetos infinitos. El capítulo 3 está basado en los siguientes artículos:

- [65] J.J.M.M. Rutten, A. Ballester-Bolinches, and E. Cosme-Llópez. **Varieties and covarieties of languages (preliminary version)**. In D. Kozen and M. Mislove, editors, Proceedings of MFPS XXIX, volume 298 of *Electron. Notes Theor. Comput. Sci.*, pages 7–28, 2013.

- [14] A. Ballester-Bolinches, E. Cosme-Llópez, and J. Rutten. **The dual equivalence of equations and coequations for automata**. *Information and Computation*, 244:49 – 75, 2015.

Esta dualidad se utiliza en el capítulo 4 para presentar un nuevo acercamiento al teorema de variedades de Eilenberg. En primer lugar presentamos una descripción equivalente, basada en ecuaciones y coecuaciones, de la noción original de variedades de lenguajes de Eilenberg. Esta nueva descripción es uno de los mejores ejemplos posibles del poder expresivo de los funtores free y cofree.

Una adaptación adecuada de estas construcciones nos permite presentar un resultado de tipo Eilenberg para formaciones de monoides no necessariamente finitos. En nuestro caso, primeramente probamos que las formaciones de monoides están en correspondencia biunívoca con las formaciones de congruencias. Un segundo paso en la prueba relaciona formaciones de congruencias y formaciones de lenguaje. Así, probamos que todos estos tres conceptos son equivalentes

$$\begin{array}{ccccc}\text{Formaciones} & & \text{Formaciones} & & \text{Formaciones}\\ \text{de monoides} & \Leftrightarrow & \text{de congruencias} & \Leftrightarrow & \text{de lenguajes}\end{array}$$

La primera correspondencia parece ser completamente nueva y relaciona formaciones de monoides con filtros de congruencias para cada monoide. La última correspondencia es uno de los mejores ejemplos de aplicación de la dualidad presentada en el capítulo 3. Además, damos una aplicación de estas equivalencias para el caso de los lenguajes relativamente disyuntivos. Estos teoremas pueden ser convenientemente modificados para cubrir el caso de las variedades de monoides en el sentido de Birkhoff. Discutiremos este caso particular al final del capítulo 4. Los resultados de este capítulo han sido enviados para su posible publicación en una revista científica bajo el título

- [13] A. Ballester-Bolinches, E. Cosme-Llópez, R. Esteban-Romero, and J. Rutten. **Formations of monoids, congruences, and formal languages**. 2015. Enviado.

El capítulo 5 está completamente dedicado al estudio del objeto final asociado a los autómatas no deterministas. En general, las técnicas utilizadas en el capítulo 5 difieren de las presentadas en los capítulos 3 y 4. En consecuencia, al principio de este capítulo introducimos algunos conceptos preliminares sobre bisimulaciones y objetos finales. Nuestro resultado principal se presenta en el Teorema 5.17, que describe el autómata final no determinista con la ayuda de estructuras basadas en lenguajes. A continuación, relacionamos otras descripciones del autómata final no determinista con nuestra construcción. El capítulo 5 está basado en el siguiente artículo:

- [11] A. Ballester-Bolinches, E. Cosme-Llópez, and R. Esteban-Romero. **A description based on languages of the final non-deterministic automaton.** *Theor. Comput. Sci.*, 536(0):1 – 20, 2014.

Ciertamente, los diferentes puntos de vista utilizados en esta disertación ya han sido explorados en otros trabajos. Por eso, al final de cada capítulo, presentamos un estudio detallado de los trabajos relacionados y discutimos las aportaciones o mejoras realizadas en los resultados existentes. Finalmente, el capítulo 6 presenta las conclusiones e indica los trabajos que se han de realizar en el futuro. También presentamos algunos artículos de investigación que se derivan de la realización de este proyecto.

Perdut cada dia com sempre, patines
per un tobogan i passa sa vida a tota
hosti i a molta alta velocitat.
Tu m'entens, jo sé que tu m'entens.

Foto — Joan Miquel Oliver

# Contents

# CHAPTER 1

# Introduction

Fundamentally, Computer Science is a science of abstraction aiming at providing a right model for thinking about a process and devising the appropriate mechanisable techniques to solve it. Computer scientists must create abstractions of real-world problems that can be understood by computer users and, at the same time, that can be represented and manipulated inside a computer. Among several existing methods that can be used for this purpose, we underscore in this dissertation the expressiveness of automata.

Automata can be seen as graphs with outgoing arcs on each node (or *state*) labelled with symbols in a set called *alphabet*. States can be flagged as *starting* or *accepting* states. One of the most basic operations using an automaton is to take a sequence of symbols and follow from an starting state a path whose arcs are labelled by these symbols in the correct order. The automaton changes from state to state according to these *inputs*. Basically, automata render an accept/reject decision on any sequence of input characters by seeing whether there is a path from an starting state to some accepting state labeled by that sequence. The set of all words that are accepted, that is the *language accepted* (or *recognised*) by the automaton, represents the power of the automaton to discriminate words; in each context, some words are useful and some are not.

Imagine the designing process of a vending machine; for each product, we will be interested in collect the amount of money specified by the vendor. It should start in a void state, where no money has been entered by the consumer. The machine needs to differentiate all possible combinations of coins or bills inserted during the process and it should recognise when the sum of all these inputs is enough to pay the selected product. The system behind this process and its behaviour are described with an automaton. With such simple abstraction process, automata can be used to model many important kinds of software and hardware. Indeed, the behaviour of digital circuits, the lexical analyser of a typical compiler, the software for scanning large bodies of text with the purpose of finding occurrences of words, phrases and other patterns, or the software for verifying communication protocols or protocols for secure exchange of information are explained and designed using automata.

## Algebraic treatment of automata

During the development of this field, a desirable abstract description of the data handled by automata was seen to be necessary and algebraic techniques were used to describe these objects. Kleene's theorem (1956) [46] is considered the starting point of the algebraic study of automata theory. Kleene described finite deterministic automata (which he called nerve nets) and showed that the class of recognisable languages, that is, recognised by a finite automaton, is equal to the class of all rational languages, which can be written using rational expressions.

Kleene's rational expressions, which can be used for describing sequential circuits, were defined using three operators (union, concatenation and iterate) on languages. Word descriptions of problems can be more easily put in the regular expression language if the language is enriched by the inclusion of other logical operations. Thus, in 1964, Brzozowski [26] introduced the notion of derivative of a rational expression, which allowed him to prove Kleene's theorem without having to recur to non-deterministic automata.

These esentially combinatorical definitions can be interpreted in algebraic and logical terms. In this context, the definition of the *syntactic monoid*, a monoid constructed accordingly to each language, was a cornerstone in the algebraic study of these objects. This concept first appeared in a work of Rabin and Scott [57], although the notion is credited to Myhill. A fundamental question in the theory is to reveal what information about a language, or an automaton accepting this language, is encoded in its syntactic monoid. It was shown, in particular, that a language is recognisable if and only if its syntactic monoid is finite. The first classification results on recognisable languages were stated in terms of automata and the first non-trivial use of the syntactic monoid can be found in the work of Schützenberger [71]. Schützenberger theorem (1965) states that a rational language is star-free if and only if its syntactic monoid is aperiodic. This interesting result is considered, right after Kleene's theorem, as the most important result in the early years of the theory.

These results allowed Eilenberg (1974-76) [31, 32] to present an axiomatisation of the algebraic approach to automata theory. Following his work, a *variety of finite monoids* or *pseudovariety* is a class of finite monoids closed under taking submonoids, quotients and finite direct products. Eilenberg's theorem states that varieties of finite monoids are in one to one correspondence with varieties of regular languages, these being classes of regular languages closed under Boolean operations, derivatives and preimages of monoid homomorphisms. His correspondence is a direct one; larger varieties of languages correspond to larger varieties of monoids. For instance, the rational languages are associated to the variety of all finite monoids (Kleene [46]), the star-free languages with the variety of finite aperiodic monoids (Schützenberger [71]), and the piecewise testable languages with the variety of finite $\mathcal{J}$-trivial monoids (Simon [75]). Eilenberg's work put scattered results on diverse classes of languages into a general setting and initiated a research line in automata theory with particular emphasis in the study and description of classes of both monoids and languages.

Eilenberg's definition of a variety of finite monoids is similar to, but subtly different from, the definition of *variety of monoids* in the original sense of Birkhoff [19]. A variety of monoids is defined as being closed under taking substructures, taking quotients, and taking arbitrary (not necessarily finite) products of structures. Birkhoff proved two main results. First, he showed that varieties are characterised by the sets of identities they satisfy. For example, semigroups are characterised among structures with one binary operations by the associative identity $(xy)z = x(yz)$. Second, he gave a set of closure conditions that characterise those sets of identities.

Varieties play a central role in the theory of Universal Algebra, and it was thus natural to seek analogous connection between pseudovarieties of monoids and identities. It was Reiterman [59], who showed that varieties are characterised by the "implicit identities" that they satisfy. In a modern rendering, these "implicit identities" are better known as *profinite identities*. A profinite identity is a relation between two profinite words, these being limits of sequences of ordinary terms, or elements in the projective limit of a family of given monoids.

Almeida [7], among other researchers in the field, made further progress by characterising the sets of profinite identities that correspond to pseudovarieties of monoids. The profinite aproach is a very rich and powerful tool to understand several recent developments in the field. Pippenger exploited the duality established by Stone [77, 78] between Boolean algebras and certain topological spaces called *Stone spaces*. Following this path, Gehrke, Grigorieff, and Pin proved in [38] that any lattice of recognisable languages can be defined by a set of profinite equations, a result that subsumes Eilenberg's original result.

Several attempts were made to extend Eilenberg's variety theory to a larger scope. For instance, ordered syntactic semigroups were considered in [54]. The resulting extension of Eilenberg's variety theory permits to treat classes of languages that are not necessarily closed under complement. Other extensions were developed by Straubing [79] and Ésik and Ito [33] and more in [38].

However, in group theory, varieties are challenged by another notion. Although varieties are incontestably a central notion, many results are better formulated in the setting of *formations*. A formation of groups is a class of groups closed under taking subdirect products and quotients. The significance of formations in group theory is apparent since they are the first remarkable step in the development of a generalised Sylow theory. It was Gaschütz and Lubeseder who began their pioneering work on the subject in 1963 [35]. Since that time the subject has proliferated and has played a fundamental role in studying groups. This raised the question whether Eilenberg's variety theorem could be extended to a formation theorem. For the case of formation of finite monoids, we can find a positive answer in the paper [15].The weaker closure conditions for formations lead to more possibilities than for varieties as more general classes of languages can be described and understood.

## Coalgebraic treatment of automata

It turned out, however, that this algebraic treatment was unable to describe most of the inherent dynamical structures used in Computer Science. During the last years most of the state-based systems were not studied as algebras but as coalgebras, the formal dual of algebras. Coalgebras [63, 62] were used to represent infinite data or behavior defined by observations rather than constructors, and come equipped with corecursive definitions of functions and coinduction as a dual proof principle of induction. Coalgebra theory offers a unifying mathematical framework for state-based behavioural systems and (component-based and service-oriented) programming paradigms. Examples of coalgebras include streams and transition systems and several variations of automata. The theory of universal coalgebra provides a standard equivalence and a universal domain of behaviors, uniquely based on the type of the system, given by a functor $F$. Most of this new techniques rely on the description of the final coalgebra [73] (the categorical dual of the initial algebra).

For the case of deterministic automata, the first coalgebraic studies [62] of these structures focus on the notions of homomorphism and bisimulation that lead to a uniform presentation of automata theory. The main contribution for that matter is based in the coalgebraic study of languages. For deterministic automata, the set of all languages over a fixed alphabet plays a central role. It carries an automaton structure determined by the notion of Brzozowski's derivatives of languages [26, 28] and it is shown to be a final automaton. Taking this property as a basis, new proof methods based on bisimulations for language equality and language inclusion were presented [62]. Based on these pioneering works, Rutten used finality for both definitions and proofs by coinduction on power series [66] or stream calculus [64] based on *behavioural differential equations*. Other variants of automata have been studied from a coalgebraic perspective, specially non-deterministic automata [73] or linear weighted automata [20]. The case of automata is of special interest since they can be viewed both as an algebra and as a coalgebra. The grounds of this work are built on these previous constructions and results. Specially on [22], which is the first work that clearly present the dual nature of automata from the very beginning. Generally less information is known on the coalgebraic side, that is why we believe this coalgebraic aspect has not received the attention it deserves.

## Summary

In the present work we want to study automata both from an algebraic perspective and a coalgebraic one. We want to exploit the dual nature of these objects and present a unifying framework to explain and extend some recent results in automata theory.

Accordingly, Section 2 contains background material and definitions to keep the work as self-contained as possible. Thus, the notions of algebra and coalgebra for endofunctors are presented. We also introduce some basic concepts on monoids and languages. In this Chapter we also introduce the notions of deterministic and non-deterministic automata, homomorphisms and bisimulations of automata and the product and coproduct of these structures. Finally, we recall some basic notions of lattice theory.

From the algebraic perspective, automata are algebras with unary operations. In this context, an equation is just a pair of words, and it holds in an automaton if for every initial state, the states reached from that state by both words are the same. It can be shown that, for a given automaton, we can construct the largest set of equations it satifies, which turns out to be a congruence on the free monoid on the input alphabet. We use this construction to define the free automaton associated to a given automaton, denoted by $\mathsf{free}$. Coalgebraically, an automaton is a transition system with final states. A coequation is then a set of languages and it is satisfied by an automaton if, for every possible observation (colouring the states as either final or not) the language accepted by the automaton is within the specified coequation. Intuitively, coequations can be thought of as behaviours, or pattern specifications that a coalgebra is supposed to have. As we did before, for a given automaton, we can construct the smallest set of coequations it satifies, which turns out to be a special subset on the set of all languages over the input alphabet. We use this construction to define the cofree automaton associated to a given automaton, denoted by $\mathsf{cofree}$. These constructions based on equations and coequations are proved to be functorial.

In Chapter 3 we have established a new duality result between *congruence quotients* of the free monoid and its set of coequations, what we called *preformations of languages*, which are complete atomic boolean algebras closed under derivatives. This duality result does not impose any restriction on the size of the objects, therefore infinite objects are allowed. Chapter 3 is based on the following papers:

- [65] J.J.M.M. Rutten, A. Ballester-Bolinches, and E. Cosme-Llópez. **Varieties and covarieties of languages (preliminary version)**. In D. Kozen and M. Mislove, editors, Proceedings of MFPS XXIX, volume 298 of *Electron. Notes Theor. Comput. Sci.*, pages 7–28, 2013.

- [14] A. Ballester-Bolinches, E. Cosme-Llópez, and J. Rutten. **The dual equivalence of equations and coequations for automata**. *Information and Computation*, 244:49 – 75, 2015.

This duality result is used in Chapter 4 to present a renewed approach to Eilenberg's variety theorem. In the first place, we introduce an equivalent description based on equations and coequations of the original notion of variety of regular language, originally introduced by Eilenberg. This description is one of the best examples of the expressiveness power of the aforementioned functors $\mathsf{free}$ and $\mathsf{cofree}$. A suitable adaptation of this construction allows us to present an Eilenberg-like result for formations of (non-necessarily finite) monoids. In our case, we first prove that formations of monoids are in one-to-one correspondence with formations of congruences. A second step in our proof relates formations of congruences and formations of languages. All in all, these three concepts are shown to be equivalent

$$\begin{array}{c} \text{Formations} \\ \text{of monoids} \end{array} \Leftrightarrow \begin{array}{c} \text{Formations} \\ \text{of congruences} \end{array} \Leftrightarrow \begin{array}{c} \text{Formations} \\ \text{of languages} \end{array}$$

The first correspondence seems to be completely new and relates formations of monoids to filters of congruences on every possible free monoid. The last correspondence is one of the best possible examples of application of the duality theorem presented in Chapter 3. We also give an application of this equivalence to the case of relatively disjunctive languages. These theorems can be slightly adapted to cover the case of varieties of monoids in the sense of Birkhoff. We discuss this particular case at the end of the Chapter 4. The results of this Chapter have been submitted to a journal for its possible publication under the title

- [13] A. Ballester-Bolinches, E. Cosme-Llópez, R. Esteban-Romero, and J. Rutten. **Formations of monoids, congruences, and formal languages**. 2015. Submitted.

Chapter 5 is completely devoted to the study of the final object associated to non-deterministic automata. In general, the techniques applied in Chapter 5 differ from those presented in Chapters 3 and 4. Consequently, at the beginning of this chapter we introduce some basic background on bisimulations and final objects. Our main result is presented in Theorem 5.17 which describes the final non-deterministic automaton with the help of structures based on languages. Hereafter, we relate other descriptions of the final non-deterministic automaton with our construction. Chapter 5 is based on the following paper:

- [11] A. Ballester-Bolinches, E. Cosme-Llópez, and R. Esteban-Romero. **A description based on languages of the final non-deterministic automaton.** *Theor. Comput. Sci.*, 536(0):1 – 20, 2014.

Certainly, the point of view that we adopt throughout this work has been explored in some other references too. Therefore, at the end of each Chapter, we present a detailed study of the related work and how our work subsumes or improves the existing results. Finally, Chapter 6 sets out the conclusions and indicates future work. We also present some of the derived research papers we have made during the realisation of this project.

Digue'm les vives meravelles
del teu treball, del teu turment.
Sota el concert de les estrelles,
anem fumant tranquil·lament.

Cançó de suburbi — Toti Soler
Silvia Pérez Cruz, (Ovidi Montllor)

# Methodology

The working techniques and the metodology used in this work are the usual ones in basic mathematical research. External help of computer programs will assist us in the development of an appropriate methodology, specially in the fields of groups, monoids, and automata theory. We make use of the computer programs GAP (Groups Algorithms and Programming), MAGMA (The Magma Computational Algebra System), and SEMIGROUPE, for the study of automata and formal languages with which we can computationally analyse properties of groups and semigroups and find examples of structures with desired properties.

We begin this work with a searching and updating process of recent related bibliographical sources that can be of interest with the specified goals. The research will be later addressed having into account the proposed objectives and the consulted material. Nevertheless, the initial objectives can be reformulated depending on the results obtained.

The contact between our research group and other research groups is essential. It provide us new ideas and different perspectives in the challenges we face. We will ensure this aspect by attending research meetings, seminars and conferences. We also planned stays in different research centers with the idea of creating consolidated research networks.

> Defensem la pràctica pràctica de doctrines nul·les basada en els principis d'ultra-caos creatiu fonamentant els postulats en un granet de xufa.
>
> Filles d'un meló d'Alger — Orxata Sound System

# CHAPTER 2

# Preliminaries

## 2.1 Algebras and coalgebras for endofunctors

**Definition 2.1.** A *category* $\mathbf{X}$ is given by a tuple $\mathbf{X} = (\mathbf{X}_0, \mathbf{X}_1, \mathsf{Dom}, \mathsf{Cod}, \circ, id)$ of *objects* $\mathbf{X}_0$ and *arrows* $\mathbf{X}_1$, that has the following structure:

- Each arrow has a *domain* and a *codomain* which are objects. One writes $f\colon X \to Y$ if $X$ is the domain of the arrow $f$ and $Y$ its codomain. One also writes $X = \mathsf{Dom}(f)$ and $Y = \mathsf{Cod}(f)$;

- Given two arrows $f$ and $g$ such that $\mathsf{Cod}(f) = \mathsf{Dom}(g)$, the *composition* of $f$ and $g$ written $g \circ f$, or simply $gf$, is defined and has domain $\mathsf{Dom}(f)$ and codomain $\mathsf{Cod}(g)$, that is

$$\left.\begin{array}{c} f\colon X \to Y \\ g\colon Y \to Z \end{array}\right| \quad \longmapsto \quad gf\colon X \to Z;$$

- Composition is associative, that is, given $f\colon X \to Y$, $g\colon Y \to Z$ and $h\colon Z \to W$ it holds that $h(gf) = (hg)f$;

- For every object $X$ there is an identity arrow $id_X\colon X \to X$, satisfying $id_X g = g$ for every $g\colon Y \to X$ and $f id_X = f$ for every $f\colon X \to Y$.

For example, the tuple $(\mathbf{Set}_0, \mathbf{Set}_1)$, consisting in sets from a Grothendieck Universe $\mathcal{U}$ and usual mappings between sets respectively, is a category denoted by $\mathbf{Set}$. The *opposite category* or *dual* category $\mathbf{X}^{\mathrm{op}}$ of a given category $\mathbf{X}$ is formed by reversing the arrows, that is interchanging the domain and codomain of each arrow. Doing the reversal twice yields the original category, so the opposite of an opposite category is the original category itself. In symbols, $(\mathbf{X}^{\mathrm{op}})^{\mathrm{op}} = \mathbf{X}$.

**Definition 2.2.** Given two categories $\mathbf{X}$ and $\mathbf{X}'$, a *functor* $F\colon \mathbf{X} \to \mathbf{X}'$ consists on a pair of mappings $F_0\colon \mathbf{X}_0 \to \mathbf{X}'_0$ and $F_1\colon \mathbf{X}_1 \to \mathbf{X}'_1$ such that:

- For each $f\colon X \to Y$, $F_1(f)\colon F_0(X) \to F_0(Y)$;
- For each $f\colon X \to Y$, $g\colon Y \to Z$ it holds that $F_1(gf) = F_1(g)F_1(f)$;
- For each $X \in \mathbf{X}_0$, $F_1(id_X) = id_{F_0(X)}$.

When $\mathbf{X}$ and $\mathbf{X}'$ are the same category, we say that $F$ is an *endofunctor*. Given two categories $\mathbf{X}$ and $\mathbf{X}'$ an *equivalence of categories* consists of a functor $F\colon \mathbf{X} \to \mathbf{X}'$ and a functor $G\colon \mathbf{X}' \to \mathbf{X}$ with two natural isomorphisms $: FG \to id_{\mathbf{X}'}$ and $: GF \to id_{\mathbf{X}}$. If a category is the opposite or dual of another category then one speaks of a *duality of categories*, and says that the two categories are dually equivalent. We denote the existence of this duality by $\mathbf{X} \simeq \mathbf{X}'$. The reader is referred to [50] for more information about category theory.

We are ready to introduce the basic concepts of algebra and coalgebra for an endofunctor in a category. For an introduction to the theory of algebras and coalgebras, the reader is referred to [63] and [62].

**Definition 2.3.** Given a category $\mathbf{X}$, called the *base category*, and an endofunctor $H\colon \mathbf{X} \to \mathbf{X}$, an *$H$-algebra* consists of a pair $(X, \alpha)$, where $X$ is an object of $\mathbf{X}$ and $\alpha\colon H(X) \to X$ an arrow in $\mathbf{X}$. On the other hand, an *$H$-coalgebra* consists of a pair $(X, \alpha)$, where $X$ is an object of $\mathbf{X}$ and $\alpha\colon X \to H(X)$ an arrow in $\mathbf{X}$. We call $X$ the *base* and $\alpha$ the *structure map* of the (co)algebra. When the endofunctor is clear we will refer to the pair simply as a (co)algebra.

**Definition 2.4.** Given an endofunctor $H$ in the category $\mathbf{X}$ and two $H$-algebras $(X, \alpha)$ and $(Y, \beta)$, an *homomorphism of $H$-algebras* from $(X, \alpha)$ into $(Y, \beta)$ is an arrow $f\colon X \to Y$ in $\mathbf{X}$ satisfying $f\alpha = \beta H(f)$, i.e., an arrow such that the following diagram commutes.

$$\begin{tikzcd}
H(X) \arrow[r, "H(f)"] \arrow[d, "\alpha"'] & H(Y) \arrow[d, "\beta"] \\
X \arrow[r, "f"'] & Y
\end{tikzcd}$$

**Definition 2.5.** We say that an $H$-algebra $(X, \alpha)$ is *initial* if, for any other $H$-algebra $(Y, \beta)$, there exists a unique $H$-algebra homomorphism $f_Y\colon (X, \alpha) \to (Y, \beta)$ .

**Definition 2.6.** Given an endofunctor $H$ in the category $\mathbf{X}$ and two $H$-coalgebras $(X, \alpha)$ and $(Y, \beta)$, an *homomorphism of $H$-coalgebras* from $(X, \alpha)$ into $(Y, \beta)$ is an arrow $f\colon (X, \alpha) \to (Y, \beta)$ in $\mathbf{X}$ satisfying $H(f)\alpha = \beta f$, i.e., an arrow such that the following diagram commutes.

$$\begin{tikzcd}
X \arrow[r, "f"] \arrow[d, "\alpha"'] & Y \arrow[d, "\beta"] \\
H(X) \arrow[r, "H(f)"'] & H(Y)
\end{tikzcd}$$

**Definition 2.7.** We say that an $H$-coalgebra $(X, \alpha)$ is *final* if, for any other $H$-coalgebra $(Y, \beta)$, there exists a unique $H$-coalgebra homomorphism $f_Y\colon (X, \alpha) \to (Y, \beta)$ .

All underlying sets of the algebras and coalgebras presented in the following chapters will be objects of the category **Set** of sets and functions. For sets $X$ and $Z$ we define $X^Z = \{g \mid g\colon Z \to X\}$. For sets $X, Y, Z$ and functions $f\colon X \to Y$ we define $f^Z\colon X^Z \to Y^Z$ by $f^Z(g) = fg$. We define the *image* and the *kernel* of a function $f\colon X \to Y$ by

$$\begin{aligned} \mathsf{im}(f) &= \{y \in Y \mid \exists x \in X, \quad f(x) = y\} \\ \mathsf{ker}(f) &= \{(x_1, x_2) \in X \times X \mid f(x_1) = f(x_2)\} \end{aligned}$$

For an arbitrary set $A$, we will be considering algebras and coalgebras for the following endofunctors on **Set**

$$\begin{array}{rcl} F(S) &=& S^A \\ G(S) &=& S \times A \\ R(S) &=& \mathcal{P}_\omega(S)^A \\ (2 \times F)(S) &=& 2 \times S^A \\ (1 + G)(S) &=& 1 + (S \times A) \\ (2 \times R)(S) &=& 2 \times \mathcal{P}_\omega(S)^A \end{array}$$

Here $\mathcal{P}_\omega(S)$ denotes the set of all finite subsets of the set $S$. Moreover, 2 is a set with two elements and 1 is a set with one element, + stands for disjoint union of sets and × for its cartesian product.

## 2.2 Monoids

**Definition 2.8.** A *monoid* is a tuple $(M, \cdot, 1)$, where $M$ is a set and $\cdot$ an associative binary operation in $M$ that has $1 \in M$ as *identity element*. When the operation and the identity element in a monoid $(M, \cdot, 1)$ are clear, we will denote the monoid simply by $M$.

For monoids $M$ and $N$, an *homomorphism of monoids* is a function $f\colon M \to N$ such that

- for all $m, m' \in M$, it holds $f(mm') = f(m)f(m')$;
- the identity of $M$ is mapped to the identity of $N$.

The homomorphism $f$ is an *isomorphism of monoids* if it is both an injective and a surjective homomorphism of monoids. In this case, we will write $M \cong N$. Monoids with homomorphism of monoids form a category denoted by **Mon**.

**Definition 2.9.** For a monoid $M$, a subset $N \subseteq M$ is a *submonoid* if $1 \in N$ and $N$ is closed under the binary operation of $M$. In this case, the inclusion mapping $i\colon N \to M$ is an injective monoid homomorphism.

For a monoid $M$, a *right congruence* is an equivalence relation $\theta$ on $M$ such that, for all $(m, n) \in M \times M$ and $p \in M$,

- if $(m, n) \in \theta$, then $(mp, np) \in \theta$

A *left congruence* is an equivalence relation $\theta$ on $M$ such that, for all $(m, n) \in M \times M$ and $p \in M$,

- if $(m, n) \in \theta$, then $(pm, pn) \in \theta$

We call $\theta$ a *congruence* if it is both a right and a left congruence. Equivalently, an equivalence relation $\theta$ is a congruence if for all pairs $(m, n)$, $(m', n')$ in $\theta$, the pair $(mm', nn')$ also belongs to $\theta$. For a monoid $M$ and a congruence $\theta$ on $M$, the quotient $M/\theta$ is a monoid for the multiplication given by $([m], [n]) \mapsto [mn]$. In this case, the quotient mapping $\pi\colon M \to M/\theta$ is a surjective monoid homomorphism. We denote the set of all congruences over $M$ by $\mathsf{Con}(M)$.

Recall that, for a monoid homomorphism $f\colon M \to N$, the kernel of $f$, $\mathsf{ker}(f)$, is a congruence on $M$ and the image of $f$, $\mathsf{im}(f)$, is a submonoid of $N$. Moreover, the mapping $\overline{f}\colon M/\mathsf{ker}(f) \to \mathsf{im}(f)$, given by $[m] \to f(m)$, is an isomorphism of monoids.

## 2.3 Languages

**Definition 2.10.** An *alphabet* $A$ is a set, whose elements are called *letters*. A *word* over an alphabet $A$ is a finite sequence $a_1a_2 \cdots a_n$ of letters of $A$. We denote the empty word by $\varepsilon$. The set of all words over $A$ is denoted by $A^*$.

Note that $A^*$ can be regarded as the free monoid on the set $A$, where the multiplication in $A^*$ is defined as the *concatenation* of words. For words $v$ and $w$ in $A^*$ its concatenation is given by the word whose first letters bijectively correspond with the letters of $v$, respecting its order, and immediately followed by the letters of $w$ in the same way. We denote the concatenation of $v$ and $w$ by $vw$. One of the most important consequences of the universal property of the free monoid is presented in the following Proposition. It states that all free monoids are projective.

**Proposition 2.11** ([53, p. 10])**.** For a set $A$ and monoids $P$ and $Q$, if $\gamma\colon A^* \to Q$ is a monoid homomorphism and $\eta\colon P \to Q$ is a surjective monoid homomorphism, then there exists a monoid homomorphism $\varphi\colon A^* \to P$ with $\eta \circ \varphi = \gamma$.

$$\begin{tikzcd}
A^* \arrow[d, dashed, "\varphi"'] \arrow[dr, "\gamma"] & \\
P \arrow[r, two heads, "\eta"'] & Q
\end{tikzcd}$$

**Definition 2.12.** A *language* $L$ over $A$ is a subset $L \subseteq A^*$ and we denote the set of all languages over $A$ by

$$2^{A^*} \cong \{L \mid L \subseteq A^*\}$$

(we will ignore here and sometimes below the difference between subsets and characteristic functions assuming the above sets as equal). If $L$ and $L'$ are languages, we define

1. the *sum* of $L$ and $L'$, as $L + L' = L \cup L'$, which coincides with the set-theoretical union of $L$ and $L'$;

2. the *product* of $L$ and $L'$, as $LL' = \{ww' \mid w \in L,\ w' \in L'\}$, composed by the words which are the result of concatenating one word of $L$ and one word of $L'$;

3. the *complementation* of $L$, as $A^* \setminus L$, and

4. the *Kleene star* of $L$, as $L^* = \bigcup_{n\in\mathbb{N}} L^n$, where $L^0 = \{\varepsilon\}$, $L^1 = L$, and $L^{n+1} = L^nL$, for $n \in \mathbb{N}$.

**Definition 2.13.** The *set of all regular languages* $\mathcal{R}$ is the smallest set of languages containing all finite languages and closed under taking sums, products, and Kleene stars.

It is usual to identify a letter $a$ with the word $(a)$, of length one, and also with the language $\{a\}$. With this convention, we can identify the regular languages with the so-called *regular expressions.* For a language $L \subseteq A^*$ and $a \in A$ we define the *$a$-derivative* of $L$ by

$$L_a = \{v \in A^* \mid av \in L\},$$

and we define, more generally, for a word $w \in A$, the $w$-derivative of $L$ by

$$L_w = \{v \in A^* \mid wv \in L\}.$$

In fact, $L_a$ and $L_w$ are also called *right derivatives* of $L$, in contrast to the *left derivative* of $L$, which we define, respectively, by

$$_aL = \{v \in A^* \mid va \in L\}, \qquad _wL = \{v \in A^* \mid vw \in L\}.$$

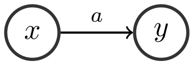
One readily verifies that the operations $(\ )_w$ and $_w(\ )$ of right and left derivatives commute with the Boolean operations of (possibly infinite) union, intersection and complementation, on languages.

## 2.4 Automata theory

**Definition 2.14.** Given an alphabet $A$, a *deterministic automaton* is a pair $(X, \alpha)$ consisting of a (possibly infinite) set $X$ of states and a transition function

$$\alpha \colon X \to X^A.$$

In pictures, we use the following notation:



$$\Leftrightarrow \quad \alpha(x)(a) = y$$

We will also write $x_a = \alpha(x)(a)$ and, more generally,

$$x_\varepsilon = x \qquad \text{and} x_{wa} = \alpha(x_w)(a),$$

with $w \in A^*$.

We observe that a deterministic automaton is an *$F$-coalgebra*[63]. Because there is, for any $A$ and $X$, a natural isomorphism

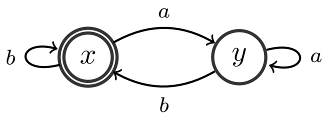
$$(\tilde{\ }) \colon \ (X \to X^A) \to ((X \times A) \to X), \qquad \tilde{\alpha}(x, a) = \alpha(x)(a).$$

deterministic automata are also *$G$-algebras* [51].

A deterministic automaton can be decorated by means of a *colouring* function

$$c \colon X \to 2,$$

using as set of colours $2 = \{0, 1\}$. We call a state $x$ *accepting* (or final) if $c(x) = 1$, and non-accepting if $c(x) = 0$. We call a triple $(X, c, \alpha)$ a deterministic *coloured* automaton. In pictures, we use a double circle to indicate that a state is accepting. For instance, in the following automaton



the state $x$ is accepting and the state $y$ is not. By pairing the functions $c$ and $\alpha$, we see that a deterministic coloured automaton is a $(2 \times F)$-coalgebra

$$\langle c, \alpha \rangle \colon X \to 2 \times X^A.$$

Given a deterministic coloured automaton $(X, c, \alpha)$ and a state $x \in X$, the set

$$o_c(x) = \{w \in A^* \mid c(x_w) = 1\}$$

is called the *language accepted* or *recognised* by the automaton $(X, c, \alpha)$ starting from the state $x$. A deterministic automaton can also have an *initial state* $x \in X$, here represented by a function

$$x \colon 1 \to X,$$

where $1 = \{0\}$. We call a triple $(X, x, \alpha)$ a *pointed* automaton. By pairing the functions $x$ and $\tilde{\alpha}$, we see that a deterministic pointed automaton is a $(1 + G)$-algebra:

$$[x, \tilde{\alpha}] \colon (1 + (X \times A)) \to X.$$

We call a 4-tuple $(X, x, c, \alpha)$ a *pointed and coloured deterministic automaton.* We could depict it by either of the following two diagrams

$$\begin{tikzcd}
1 \arrow[dr, bend left, "x"] & & 2 \\
& X \arrow[ur, bend left, "c"] \arrow[d, "\alpha"'] & \\
& X^A &
\end{tikzcd}
\qquad
\begin{tikzcd}
1 \arrow[dr, bend left, "x"] & & 2 \\
& X \arrow[ur, bend left, "c"] & \\
& X \times A \arrow[u, "\tilde{\alpha}"] &
\end{tikzcd}$$

We will be using the diagram on the left, which is just a matter of personal preference. We observe further that pointed and coloured deterministic automata are simply called *automata* in most of the literature on automata theory. A pointed and coloured automaton $(X, x, c, \alpha)$ is neither an algebra nor a coalgebra —because of $c$ and $x$, respectively— which can be a cause of fascination and confusion alike.

### 2.4.1 Homomorphisms, subautomata, and bisimulations

**Definition 2.15.** Let $(X, \alpha)$ and $(Y, \beta)$ be two deterministic automata. A function $h \colon X \to Y$ is a *homomorphism* between deterministic automata $(X, \alpha)$ and $(Y, \beta)$ if it makes the following equivalent diagrams commute:

$$\begin{tikzcd}
X \arrow[r, "h"] \arrow[d, "\alpha"'] & Y \arrow[d, "\beta"] \\
X^A \arrow[r, "h^A"'] & Y^A
\end{tikzcd}
\qquad
\begin{tikzcd}
X \arrow[r, "h"] & Y \\
X \times A \arrow[u, "\tilde{\alpha}"] \arrow[r, "(h, id_A)"'] & Y \times A \arrow[u, "\tilde{\beta}"']
\end{tikzcd}$$

Equivalently, $h(x_w) = h(x)_w$ for all $x \in X$ and $w \in A^*$. In this case, $h$ is both an $F$-coalgebra homomorphism and a $G$-algebra homomorphism. An *epimorphism* is a homomorphism that is surjective, and a *monomorphism* is a homomorphism that is injective. A homomorphism of pointed deterministic automata $(X, x, \alpha)$ and $(Y, y, \beta)$ and of coloured automata $(X, c, \alpha)$ and $(Y, d, \beta)$ moreover respects initial values and colours, respectively:

$$\begin{tikzcd}
1 \arrow[d, "x"'] \arrow[dr, bend left, "y"] & \\
X \arrow[r, "h"'] & Y
\end{tikzcd}
\qquad
\begin{tikzcd}
 & 2 \\
X \arrow[ur, bend left, "c"] \arrow[r, "h"'] & Y \arrow[u, "d"']
\end{tikzcd}$$

If in the diagrams above $X \subseteq Y$, and (i) $h$ is subset inclusion

$$h \colon X \subseteq Y$$

(and, moreover (ii) $x = y$ or (iii) $c = d$), then we call $X$ a (i) *subautomaton* of $Y$ (respectively (ii) *pointed* and (iii) *coloured subautomaton*). For a deterministic automaton $(X, \alpha)$ and $x \in X$, the *subautomaton generated by* $x$, denoted by

$$\langle x \rangle \subseteq X$$

consists of the smallest subset of $X$ that contains $x$ and is closed under transitions.

**Definition 2.16.** Let $(X, \alpha)$ and $(Y, \beta)$ be two deterministic automata. We call a relation $R \subseteq X \times Y$ a *bisimulation of deterministic automata* if for all $(x, y) \in X \times Y$,

$$\text{if } (x, y) \in R \quad \text{then } \ \forall a \in A, (x_a, y_a) \in R$$

(where $x_a = \alpha(x)(a)$ and $y_a = \beta(y)(a)$). The states $x \in X$ and $y \in Y$ are said to be *bisimilar* when there exists a bisimulation $R \subseteq X \times Y$ such that $(x, y) \in R$. For pointed deterministic automata $(X, x, \alpha)$ and $(Y, y, \beta)$, $R$ is a *pointed bisimulation* if, moreover, $(x, y) \in R$. And for coloured deterministic automata $(X, c, \alpha)$ and $(Y, d, \beta)$, $R$ is a *coloured bisimulation* if, moreover, for all $(x, y) \in X \times Y$,

$$\text{if } (x, y) \in R \quad \text{then } \ c(x) = d(y)$$

A bisimulation $E \subseteq X \times X$ is called a bisimulation *on* $X$. If $E$ is an equivalence relation then we call it a *bisimulation equivalence*. The quotient map of a bisimulation equivalence on $X$ is a homomorphism of deterministic automata:

$$\begin{tikzcd}
X \arrow[r, "q"] \arrow[d, "\alpha"'] & X/E \arrow[d, "{[\alpha]}"] \\
X^A \arrow[r, "q^A"'] & (X/E)^A
\end{tikzcd}$$

with the obvious definitions of $X/E$, $q$ and $[\alpha]$. If the equivalence $E$ is a pointed bisimulation on $(X, x, \alpha)$ or a coloured bisimulation on $(X, c, \alpha)$, then we moreover require, respectively,

$$\begin{tikzcd}
1 \arrow[d, "x"'] \arrow[dr, bend left, "{[x]}"] & \\
X \arrow[r, "h"'] & X/E
\end{tikzcd}
\qquad
\begin{tikzcd}
 & 2 \\
X \arrow[ur, bend left, "c"] \arrow[r, "h"'] & X/E \arrow[u, "{[c]}"']
\end{tikzcd}$$

with, again, the obvious definitions of $[x]$ and $[c]$. For a homomorphism $h\colon X \to Y$, $\mathsf{ker}(h)$ is a bisimulation equivalence on $X$ and $\mathsf{im}(h)$ is a subautomaton of $Y$. Any homomorphism $h$ is equal to the composition of an epimorphism followed by a monomorphism, as follows:

$$\begin{tikzcd}
X \arrow[r, "e"'] \arrow[d, "\alpha"'] \arrow[rr, bend left, "h"] & X/\mathsf{ker}(h) \arrow[r, "m"'] \arrow[d, "{[\alpha]}"] & Y \arrow[d, "\beta"] \\
X^A \arrow[r, "e^A"] \arrow[rr, bend right, "h^A"'] & (X/\mathsf{ker}(h))^A \arrow[r, "m^A"] & Y^A
\end{tikzcd}$$

with $e(x) = [x] = \{z \in X \mid h(z) = h(x)\}$, and $m([x]) = h(x)$. Note that $X/\mathsf{ker}(h) \cong \mathsf{im}(h)$. The pair $(e, m)$ is called an *epi-mono factorisation* of $h$.

### 2.4.2 Products and coproducts of automata

Deterministic automata (are both $G$-algebras and $F$-coalgebras and hence) have both products and coproducts, as follows.

- The *product* of deterministic automata $\{(X_i, \alpha_i) \mid i \in I\}$ is given by $(\prod_{i\in I} X_i, \gamma)$ where $\prod_{i\in I} X_i$ is the cartesian product and where

$$\gamma\colon (\prod_{i\in I} X_i) \to (\prod_{i\in I} X_i)^A \qquad \gamma((x_i)_{i\in I})(a) = (\alpha_i(x_i)(a))_{i\in I}.$$

- The *coproduct* of deterministic automata $\{(X_i, \alpha_i) \mid i \in I\}$ is given by $(\sum_{i\in I} X_i, \gamma)$ where $\sum_{i\in I} X_i$ is the disjoint union and where

$$\gamma\colon (\sum_{i\in I} X_i) \to (\sum_{i\in I} X_i)^A \qquad \gamma(z)(a) = \alpha_i(z)(a) \quad \text{if } z \in X_i.$$

Pointed deterministic automata (are $(1 + G)$-algebras and hence) have products, as follows. The product of pointed deterministic automata $\{(X_i, x_i, \alpha_i) \mid i \in I\}$ is given by $(\prod_{i\in I} X_i, (x_i)_{i\in I}, \gamma)$ with $(\prod_{i\in I} X_i, (x_i)_{i\in I}, \gamma)$ as above and with initial state

$$(x_i)_{i\in I}\colon 1 \to \prod_{i\in I} X_i.$$

Coloured deterministic automata (are $(2 \times F)$-coalgebras and hence) have coproducts, as follows. The coproduct of coloured automata $\{(X_i, c_i, \alpha_i) \mid i \in I\}$ is given by $(\sum_{i\in I} X_i, [(c_i)_{i\in I}], \gamma)$ with $(\sum_{i\in I} X_i, \gamma)$ as above and with colouring function

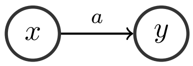
$$[(c_i)_{i\in I}]\colon \sum_{i\in I} X_i \to 2 \qquad [(c_i)_{i\in I}](z) = c_i(z) \quad \text{if } z \in X_i.$$

### 2.4.3 Non-deterministic automata

**Definition 2.17.** A non-deterministic *automaton* is a pair $(X, \alpha)$ consisting of a (possibly infinite) set $X$ of states and a transition function

$$\alpha \colon X \to \mathcal{P}_\omega(X)^A,$$

that assigns to each letter and to each state a finite set of states. When we assign to each state a single new state, we will recover the previous definition of deterministic automaton. In pictures, we use the following notation:



$$\Leftrightarrow \quad y \in \alpha(x)(a).$$

We will also write $x_a = \alpha(x)(a)$ and, more generally,

$$x_\varepsilon = \{x\}, \qquad x_{wa} = \bigcup\{y_a \mid y \in x_w\}.$$

We observe that non-deterministic automata are *R-coalgebras*. An automaton can be decorated by means of a *colouring* function

$$c \colon X \to 2,$$

using a basic set of colours $2 = \{0, 1\}$. We call a state $x$ *accepting* (or final) if $c(x) = 1$, and non-accepting if $c(x) = 0$. We call a triple $(X, c, \alpha)$ a *coloured* non-deterministic automaton. In pictures, we use a double circle to indicate that a state is accepting. By pairing the functions $c$ and $\alpha$, we see that non-deterministic coloured automata are $(2 \times R)$-coalgebras:

$$\langle c, \alpha \rangle \colon X \to 2 \times \mathcal{P}_\omega(X)^A.$$

Given a non-deterministic coloured automaton $(X, c, \alpha)$ and a state $x \in X$, the set

$$o_c(x) = \{w \in A^* \mid \exists y \in x_w \ (c(y) = 1)\}$$

is called the *language accepted* or *recognised* by the automaton $(X, c, \alpha)$ starting from the state $x$. It is also common to consider an initial state or a set of inital states in the study of non-deterministic automata, but we will not need it in our development since we will only be interested in the coalgebraic structure of non-deterministic automata. In fact, it is easy to find generalisations of the concepts above introduced for the case of non-deterministic automata. To simplify the reading, we present just the notions that will appear later.

**Definition 2.18.** A function $h \colon X \to Y$ is a *homomorphism* between non-deterministic coloured automata $(X, c, \alpha)$ and $(Y, d, \beta)$ if it makes the following diagram commute:

$$\begin{tikzcd}
 & 2 \\
X \arrow[ur, "c"] \arrow[r, "h"] \arrow[d, "\alpha"'] & Y \arrow[u, "d"'] \arrow[d, "\beta"] \\
\mathcal{P}_\omega(X)^A \arrow[r, "\mathcal{P}_\omega(h)"'] & \mathcal{P}_\omega(Y)^A
\end{tikzcd}$$

Equivalently, $\mathcal{P}_\omega(h)(x_w) = \{h(y) \mid y \in x_w\} = h(x)_w$ for all $x \in X$ and $w \in A^*$ and $c(x) = 1$ if and only if $d(h(x)) = 1$ In this case, $h$ is a $(2 \times R)$-coalgebra homomorphism.

**Definition 2.19.** We call a relation $R \subseteq X \times Y$ a *bisimulation of non-deterministic coloured automata*, or simply *bisimulation*, if for all $(x, y) \in R$ and all $a \in A$, the following conditions are satisfied

- $c(x) = 1$ if and only if $d(y) = 1$;
- If $x' \in x_a$, then there exists $y' \in Y$ with $y' \in y_a$ and $(x', y') \in R$;
- If $y' \in y_a$, then there exists $x' \in X$ with $x' \in x_a$ and $(x', y') \in R$.

Two states $x \in X$ and $y \in Y$ are said to be *bisimilar* when there exists a bisimulation $R$ between $X$ and $Y$ such that $(x, y) \in R$. A bisimulation $R \subseteq X \times X$ is called a bisimulation *on* $X$.

## 2.5 Lattice theory

**Definition 2.20.** A *partially* ordered set $\mathbb{P}$ is a pair $(P, \leq)$ where $P$ is a set and $\leq$ is a binary relation on $P$ which is reflexive, antisymmetric and transitive. Partially ordered sets are also called *posets*. We read "$x \leq y$" as "$x$ is less than or equal to $y$". We will write that $x \nleq y$ to indicate that it is not the case that $x \leq y$. To indicate that $x \leq y$ but $x \neq y$ we write $x < y$ and we say that "$x$ is (strictly) less than $y$".

**Definition 2.21.** Let $\mathbb{P} = (P, \leq)$ be a partially ordered set, let $x \in P$, and let $X \subseteq P$. Then,

1. $x$ is a *maximal element of* $X$ *in* $\mathbb{P}$ if $x \in X$ and there is no $y \in X$ such that $x < y$;
2. $x$ is a *maximum element of* $X$ *in* $\mathbb{P}$ if $x \in X$ and for every $y \in X$, $y \leq x$;
3. $x$ is an *upper bound of* $X$ if, for every $y \in X$, $y \leq x$;
4. $x$ is a *supremum of* $X$ if it is an upper bound of $X$ with $x \leq z$ for any upper bound $z$ of $X$;
5. $x$ is a *minimal element of* $X$ *in* $\mathbb{P}$ if $x \in X$ and there is no $y \in X$ such that $y < x$;
6. $x$ is a *minimum element of* $X$ *in* $\mathbb{P}$ if $x \in X$ and for every $y \in X$, $x \leq y$;
7. $x$ is a *lower bound of* $X$ if, for every $y \in X$, $x \leq y$;
8. $x$ is an *infimum of* $X$ if it is a lower bound of $X$ with $z \leq x$ for any lower bound $z$ of $X$.

If $X$ has a supremum, it is unique and we denote it by $\sup X$. Dually, if $X$ has a infimum, it is unique and we denote it by $\inf X$. If $\mathbb{P}$ has a maximum element, it is called the *top element* of $\mathbb{P}$. Dually, if $\mathbb{P}$ has a minimum element, it is called the *bottom element* of $\mathbb{P}$. If these elements exist, they are unique. A partially ordered set $\mathbb{P}$ is *bounded* if it has a top and a bottom element, usually denoted by 1 and 0 respectively.

**Definition 2.22.** Recall that a non-empty subset $F$ of a partially ordered set $P$ is called a *filter* if it satisfies:

- if $x, y \in F$, then there exists $z \in F$ with $z \leq x$ and $z \leq y$;
- if $x \in F$ and $x \leq y$, then $y \in F$.

If $x$ is an element in $P$, the subset $[x) = \{y \in P \mid x \leq y\}$ is always a filter. A filter $F$ is *principal* if it has the form $F = [x)$ for some element $x \in P$.

**Definition 2.23.** A partially ordered set $\mathbb{L} = (L, \leq)$ is a *lattice* if, for every $x, y \in L$, $\inf\{x, y\}$ and $\sup\{x, y\}$ exist. In this case we will use the notations

$$x \wedge y = \inf\{x, y\}, \qquad x \vee y = \sup\{x, y\}.$$

The binary operation $\wedge$ is called the *meet* operation of $\mathbb{L}$ and $\vee$ is called the *join* operation of $\mathbb{L}$. It follows from the definition that the operations of join and meet satisfy the following laws for all $x, y, z \in L$.

$$\begin{array}{lll}
x \wedge x = x, & x \vee x = x & \text{(Idempotent);} \\
x \wedge y = y \wedge x, & x \vee y = y \vee x & \text{(Commutative);} \\
x \wedge (y \wedge z) = (x \wedge y) \wedge z, & x \vee (y \vee z) = (x \vee y) \vee z & \text{(Associative);} \\
x \wedge (x \vee y) = z, & x \vee (x \wedge y) = x & \text{(Absoption).}
\end{array}$$

**Definition 2.24.** A *lattice as an algebra* is an algebra $\mathbb{L} = (L, \wedge, \vee)$ where $\wedge$ and $\vee$ are two binary operations, called respectively *meet* and *join*, that satisfy all above laws.

The relation between the two notions of lattice is stated in the following theorem

**Theorem 2.25.** *If $\mathbb{L} = (L, \leq)$ is a lattice, then defining the binary operations $\wedge$ and $\vee$ in $L$ by $x \wedge y = \inf\{x, y\}$ and $x \vee y = \sup\{x, y\}$, then the algebra $\mathbb{L}^a = (L, \wedge, \vee)$ is a lattice as an algebra. Further, if $\mathbb{L}^a = (L, \wedge, \vee)$ is a lattice as an algebra, then defining the relation $\leq$ in $L$ by $x \leq y$ if and only if $x \wedge y = x$, the structure $\mathbb{L}^p = (L, \leq)$ is a poset that is a lattice. Moreover, $(\mathbb{L}^a)^p = \mathbb{L}$ and $(\mathbb{L}^p)^a = \mathbb{L}$.*

In the sequel we will move back and forth between the poset notion and the algebraic notion of lattice without explicitly mentioning it, that is, we will identify lattices $\mathbb{L} = (L, \wedge, \vee)$ and $\mathbb{L} = (L, \leq)$ when they correspond one to the other.

**Definition 2.26.** For lattices $\mathbb{L}_1 = (L_1, \wedge_1, \vee_1)$ and $\mathbb{L}_2 = (L_2, \wedge_2, \vee_2)$, a function $h\colon L_1 \to L_2$ is a *lattice homomorphism* from $\mathbb{L}_1$ to $\mathbb{L}_2$ if for every $x, y \in L_1$,

$$h(x \wedge_1 y) = h(x) \wedge_2 h(y), \qquad h(x \vee_1 y) = h(x) \vee_2 h(y).$$

**Definition 2.27.** A lattice $\mathbb{L}$ is called *distributive* if, for all $x, y, z \in L$ one of the following dual conditions holds

$$x \vee (y \wedge z) = (x \vee y) \wedge (x \vee z);$$

$$x \wedge (y \vee z) = (x \wedge y) \vee (x \wedge z).$$

**Definition 2.28.** A bounded lattice $\mathbb{L} = (L, \wedge, \vee, 0, 1)$ is *complemented* if for each $x \in L$, there exists a unique element $x' \in L$ complementing $x$, that is, there exists $x' \in L$ with

$$x \wedge x' = 0 \qquad \text{and} \qquad x \vee x' = 1.$$

**Definition 2.29.** A lattice $\mathbb{L}$ is a *Boolean lattice* if it is complemented, bounded and distributive.

**Definition 2.30.** A Boolean algebra $\mathbb{B}$ is an algebra $\mathbb{B} = (B, \wedge, \vee, \,', 0, 1)$ such that

- $(B, \wedge, \vee)$ is a Boolean lattice;
- $'$ is the complement operation of that lattice;
- 1 and 0 are respectively the top and bottom elements.

The difference between Boolean algebras and Boolean lattices is that in the Boolean algebras the operation of complementation and the bounds are part of the algebra structure while in a Boolean lattice not. This has some effects in several notions concerning Boolean algebras. For example for homomorphisms. An homomorphism between Boolean algebras has to respect the complementation operation as well as the bounds, whereas an homomorphism between the corresponding Boolean lattices does not need to.

**Definition 2.31.** A Boolean algebra $\mathbb{B}$ is called *complete* if every subset has both a supremum and an infimum. An element $a \in B$ is called *atomic* whenever, for all $b \in B$: if $b \leq a$ then either $b = 0$ or $b = a$. A Boolean algebra $\mathbb{B}$ is called *atomic* if every element $b \in B$ can be expressed as the supremum of a (possibly infinite) set of atoms in $B$.

The class of all complete atomic Boolean algebras together with Boolean algebra homomorphisms preserving arbitrary infima and suprema forms a category, denoted by $\mathsf{CABA}$. Every complete atomic Boolean algebra $B$ is isomorphic to $\mathcal{P}(S)$, for some set $S$. (As a consequence, the cardinality of a *finite* Boolean algebra, which is always complete and atomic, is a power of 2). More precisely, there exists the following dual equivalence between the category $\mathsf{Set}$ of sets and functions, and the category $\mathsf{CABA}$:

$$\begin{tikzcd}
\mathsf{Set} \arrow[r, bend left, "\mathcal{P}"] \arrow[r, phantom, "\simeq"] & \mathsf{CABA}^{\mathrm{op}} \arrow[l, bend left, "\mathsf{At}"]
\end{tikzcd}$$

where the functor $\mathsf{At}$ maps a complete atomic Boolean algebra to its set of atoms.

A la vall blanca em vaig deixar
fermes arrels i un trist record,
sembrant la terra amb les cançons
vaig caminar perdut
perseguint el meu rumb.

Camins — Obrint Pas
Pep Gimeno "Botifarra"

# CHAPTER 3

# Equations and coequations: A dual equivalence

## 3.1 Introduction

Because of the natural isomorphism

$$(X \times A) \to X \quad \cong \quad X \to X^A,$$

a deterministic automaton can be viewed both as an algebra [31, 32] and as a coalgebra [62, 63]. This algebra-coalgebra duality in the modelling of automata leads us to the following setting for our investigations:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[dr, bend left, "x"] & & 2 \\
A^* \arrow[r, dashed, "r_x"'] & (X,\alpha) \arrow[ur, bend left, "c"] \arrow[r, dashed, "o_c"'] & 2^{A^*} \arrow[u, "\varepsilon?"']
\end{tikzcd} \tag{3.1}$$

In the middle, we have the automaton $(X, \alpha)$. A function $x\colon 1 \to X$ represents the choice of a designated point, that is, an initial state, $x \in X$. Dually, a function $c\colon X \to 2$ gives us a (binary) colouring of the states in $X$ or, equivalently, a set $\{x \mid c(x) = 1\}$ of final or accepting states. On the left side of our diagram, $A^*$ is the automaton of all words over $A$, with transitions

$$v \xrightarrow{a} va$$

and with the empty word $\varepsilon$ as initial state. Furthermore, every point $x\colon 1 \to X$ determines a unique homomorphism

$$r_x\colon A^* \to X, \qquad w \mapsto x_w,$$

that sends any word $w$ to the state $x_w$ reached from the initial state $x$ on input $w$. Dually, on the right hand side of our diagram, $2^{A^*}$ is the automaton of all languages over $A$, with transitions

$$L \xrightarrow{a} L_a = \{v \in A^* \mid av \in L\},$$

and colouring function $\varepsilon?$, asking whether the empty word belongs to a language or not

$$\varepsilon?(L) = \begin{cases} 1, & \text{if } \varepsilon \in L; \\ 0, & \text{if } \varepsilon \notin L. \end{cases}$$

Every colouring $c\colon X \to 2$ determines a unique homomorphism

$$o_c\colon X \to 2^{A^*} \qquad x \mapsto \{w \in A^* \mid c(x_w) = 1\}$$

that sends a state $x$ to the language that it accepts.

As it turns out, a pointed automaton $(X, x, \alpha)$ is an algebra (and not a coalgebra); a coloured automaton $(X, c, \alpha)$ is a coalgebra (and not an algebra). And a pointed and coloured automaton $(X, x, c, \alpha)$, which is what in the literature is usually taken as the definition of 'deterministic automaton', is neither an algebra nor a coalgebra.

Now sets of *equations* will live in the left – algebraic – part of our diagram and correspond to the *kernels* of the homomorphisms $r_x$; that is, sets of pairs of words $(v, w)$ with $x_v = x_w$. Dually, sets of *coequations* live in the right – coalgebraic – part of our diagram and correspond to the *image* of the homomorphisms $o_c$; that is, sets of languages containing $o_c(x)$, for every $x \in X$. Satisfaction of sets of equations and coequations by the automaton $(X, \alpha)$ will then be defined by quantifying over all points $x\colon 1 \to X$ and all colourings $c\colon X \to 2$, respectively.

The main result of the present chapter will be the proof that equations and coequations of automata are related by a dual equivalence. To this end, we will further refine diagram (3.1) as follows:

$$\begin{tikzcd}
1 \arrow[d] \arrow[dr, bend left] \arrow[drr, bend left, "x"] & & & & 2 \\
A^* \arrow[r] & \mathsf{free}(X,\alpha) \arrow[r] & (X,\alpha) \arrow[r] \arrow[urr, bend left, "c"] & \mathsf{cofree}(X,\alpha) \arrow[r] \arrow[ur, bend left] & 2^{A^*} \arrow[u]
\end{tikzcd}$$

The new diagram includes, for every automaton $(X, \alpha)$ a new automaton $\mathsf{free}(X, \alpha)$, which will be shown to represent the *largest set of equations* satisfied by $(X, \alpha)$. And, dually, we will construct an automaton $\mathsf{cofree}(X, \alpha)$, which will represent the *smallest set of coequations* satisfied by $(X, \alpha)$. The automaton $\mathsf{free}(X, \alpha)$ will turn out to be isomorphic to the so-called *transition monoid* from algebraic language theory [67, 55] and as a consequence, $\mathsf{cofree}(X, \alpha)$ can be viewed as its dual.

Next, we will show that the constructions of $\mathsf{free}(X, \alpha)$ and $\mathsf{cofree}(X, \alpha)$ are in fact functorial, that is, they act also on (certain) homomorphisms of automata. If we then restrict the functor $\mathsf{cofree}$ to the image of the category of automata under $\mathsf{free}$, we obtain our main result: a dual equivalence. This dual equivalence relates, more precisely, two special classes of automata: on the one hand, the class of quotients $A^*/C$ of the automaton $A^*$ with respect to a congruence relation $C \subseteq A^* \times A^*$; and on the other hand, the class of preformations of languages, which in the present paper are defined as subautomata of the automaton $2^{A^*}$ that are complete atomic Boolean algebras closed under left and right language derivatives. As it turns out, this duality is a lifting of the well-known dual equivalence between sets and complete atomic Boolean algebras: on congruence quotients, $\mathsf{cofree}$ acts as the powerset construction, and on preformations, applying $\mathsf{free}$ amounts to taking the set of atoms.

We then illustrate the dual equivalence between equations and coequations by applications to both regular languages and non-regular ones, such as context-free languages. Finally, we introduce the notion of *equational bisimulation* and a corresponding coinduction proof principle. For a given congruence relation $C$, we can show that a language satisfies $C$ and hence belongs to the correponding preformation of languages, by constructing a suitable equational bisimulation.

## 3.2 Setting the scene

The set $A^*$ determines a pointed automaton $(A^*, \varepsilon, \sigma)$ with initial state $\varepsilon$ and transition function $\sigma$ defined by

$$\sigma\colon A^* \to (A^*)^A, \qquad \sigma(w)(a) = wa. \tag{3.2}$$

**Proposition 3.1.** The pointed automaton $(A^*, \varepsilon, \sigma)$ is *initial* in the following sense: for any given automaton $(X, \alpha)$, every choice of initial state $x\colon 1 \to X$ induces a unique function $r_x\colon A^* \to X$, given by $r_x(w) = x_w$, that makes the following diagram commute:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[dr, bend left, "x"] & \\
A^* \arrow[r, dashed, "r_x"] & X \\
(A^*) \times A \arrow[u, "\tilde{\sigma}"] \arrow[r, dashed, "(r_x{,}id_A)"] & X \times A \arrow[u, "\tilde{\alpha}"']
\end{tikzcd}$$

This property makes $(A^*, \varepsilon, \sigma)$ an *initial* $(1+G)$*-algebra*. Equivalently, the automaton $(A^*, \sigma)$ is a *free* $G$*-algebra on the set* $1$. The function $r_x$ maps a word $w$ to the state $x_w$ reached from the initial state $x$ on input $w$ and is therefore called the *reachability* map for $(X, x, \alpha)$.

The set $2^{A^*}$ of languages determines a coloured automaton $(2^{A^*}, \varepsilon?, \tau)$ with colouring function $\varepsilon?$ defined by

$$\varepsilon?\colon 2^{A^*} \to 2, \qquad \qquad \varepsilon?(L) = \begin{cases} 1, & \text{if } \varepsilon \in L; \\ 0, & \text{if } \varepsilon \notin L. \end{cases}$$

and transition function $\tau$ defined by

$$\tau\colon 2^{A^*} \to (2^{A^*})^A, \qquad \tau(L)(a) = L_a. \tag{3.3}$$

**Proposition 3.2.** The coloured automaton $(2^{A^*}, \varepsilon?, \tau)$ is *final* in the following sense: for any given automaton $(X, \alpha)$, every choice of colouring function $c\colon X \to 2$ induces a unique function $o_c\colon X \to 2^{A^*}$, given by $o_c(x) = \{w \mid c(x_w) = 1\}$, that makes the following diagram commute:

$$\begin{tikzcd}
 & 2 \\
X \arrow[ur, bend left, "c"] \arrow[r, dashed, "o_c"] \arrow[d, "\alpha"'] & 2^{A^*} \arrow[u, "\varepsilon?"'] \arrow[d, "\tau"] \\
X^A \arrow[r, dashed, "(o_c)^A"] & (2^{A^*})^A
\end{tikzcd}$$

This property makes $(2^{A^*}, \varepsilon?, \tau)$ a *final* $(2{\times}F)$*-coalgebra*. Equivalently, the automaton $(2^{A^*}, \tau)$ is an $F$*-coalgebra that is cofree on the set* 2. The function $o_c$ maps a state $x$ to the language $o_c(x)$ accepted by $x$. Since the language $o_c(x)$ can be viewed as the observable behaviour of $x$, the function $o_c$ is called the *observability* map.

Summarising, we have set the following scene for our investigations:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[dr, bend left, "x"] & & 2 \\
A^* \arrow[r, dashed, "r_x"] \arrow[d, "\sigma"'] & X \arrow[r, dashed, "o_c"] \arrow[d, "\alpha"] \arrow[ur, bend left, "c"] & 2^{A^*} \arrow[u, "\varepsilon?"'] \arrow[d, "\tau"] \\
(A^*)^A \arrow[r, dashed, "(r_x)^A"] & X^A \arrow[r, dashed, "(o_c)^A"] & (2^{A^*})^A
\end{tikzcd} \tag{3.4}$$

If the reachability map $r_x$ is *surjective* then we call $(X, x, \alpha)$ *reachable*. If the observability map $o_c$ is *injective* then we call $(X, c, \alpha)$ *observable*. And if $r_x$ is surjective *and* $o_c$ is injective then we call $(X, x, c, \alpha)$ (reachable and observable, or:) *minimal*.

Fixing the language $L \in 2^{A^*}$, we obtain the following variation of the picture above:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[dr, bend left, "L"] & \\
A^* \arrow[r, "h"] \arrow[dr, bend right, "L"'] & 2^{A^*} \arrow[d, "\varepsilon?"] \\
 & 2
\end{tikzcd}$$

where the lower $L$ is in fact the characteristic function of $L \subseteq A^*$, and where the homomorphism $h$ satisfies $h(w) = L_w$. As a consequence, we have

$$h(v) = h(w) \;\Leftrightarrow\; v \equiv_{\mathsf{MN}} w$$

where on the right, we have the celebrated *Myhill-Nerode* equivalence, defined by

$$v \equiv_{\mathsf{MN}} w \;\Leftrightarrow\; \forall u \in A^*,\ vu \in L \Leftrightarrow wu \in L$$

A *minimal automaton accepting* $L$ is now obtained by the epi-mono factorisation of $h$:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[dr, bend left, "x"] \arrow[drr, bend left, "L"] & & \\
A^* \arrow[r, "q"] \arrow[drr, bend right, "L"'] & A^*/\mathsf{ker}(h) \arrow[r, "i"] \arrow[dr, bend right, "c"] & 2^{A^*} \arrow[d, "\varepsilon?"] \\
 & & 2
\end{tikzcd}$$

where $x = q \circ \varepsilon$ and $c = \varepsilon? \circ i$. This minimal automaton is unique up-to isomorphism because epi-mono factorisations are. And because $A^*/\mathsf{ker}(h) \cong \mathsf{im}(h)$, it is equal to

$$\langle L \rangle \subseteq 2^{A^*}$$

that is, the subautomaton of $(2^{A^*}, \tau)$ generated by $L$. All in all we have obtained the following picture:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[dr, bend left] \arrow[drr, bend left, "L"] & & \\
A^* \arrow[r, "r"] \arrow[drr, bend right, "L"'] & \langle L \rangle \arrow[r, "i"] \arrow[dr, bend right] & 2^{A^*} \arrow[d, "\varepsilon?"] \\
& & 2
\end{tikzcd} \tag{3.5}$$

with $r(w) = L_w$ and $i(K) = K$, for all $w \in A^*$ and $K \in \langle L \rangle$. In this case, $\mathsf{ker}(r) = \equiv_{\mathsf{MN}}$.

In conclusion of this section, we observe that $\langle L \rangle$ is finite iff the language $L$ is *rational*. This fact is a version [26, 28] of Kleene's correspondence between finite automata and rational languages [46].

### 3.2.1 Equations and coequations

Let $A$ be an alphabet. In this section we will be referring to the situation of (3.4).

**Definition 3.3** (equations)**.** A *set of equations* is a bisimulation equivalence relation $E \subseteq A^* \times A^*$ on the automaton $(A^*, \sigma)$. We define $(X, x, \alpha) \models E$ —and say: *the pointed automaton* $(X, x, \alpha)$ *satisfies* $E$— by

$$(X, x, \alpha) \models E \;\Leftrightarrow\; \forall (v, w) \in E,\ x_v = x_w.$$

Because

$$\forall (v, w) \in E,\ x_v = x_w \;\Leftrightarrow\; E \subseteq \mathsf{ker}(r_x),$$

we have, equivalently, that $(X, x, \alpha) \models E$ iff the reachability map $r_x$ factors through $A^*/E$:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[dr, bend left, "{[\varepsilon]}"] \arrow[drr, bend left, "x"] & & \\
A^* \arrow[r, "q"] \arrow[rr, bend right, "r_x"'] & A^*/E \arrow[r, "h"] & X
\end{tikzcd}$$

where the homomorphisms (of pointed automata) $q$ and $h$ are given by

$$q(w) = [w] \quad \text{and} \quad h([w]) = r_x(w),$$

respectively. We define $(X, \alpha) \models E$ —and say: *the automaton* $(X, \alpha)$ *satisfies* $E$— by

$$\begin{aligned}
(X, \alpha) \models E \quad &\Leftrightarrow \quad \forall x : 1 \to X, \quad (X, x, \alpha) \models E \\
&\Leftrightarrow \quad \forall x \in X,\ \forall (v, w) \in E,\ x_v = x_w.
\end{aligned}$$

□

Note that we consider *sets* of equations $E$ and that $(v, w) \in E$ implies $(vu, wu) \in E$, for all $v, w, u \in A^*$, because $E$ is —by definition— a bisimulation relation on $(A^*, \sigma)$. Still we will sometimes consider also a *single* equation $(v, w) \in A^* \times A^*$ and write

$$(X, \alpha) \models\ v = w,$$

to denote

$$(X, \alpha) \models \mathbf{v{=}w},$$

where $\mathbf{v{=}w}$ is defined as the smallest bisimulation equivalence on $A^*$ containing $(v, w)$. Furthermore, we will use $(X, \alpha) \models \{v = w,\, t = u\}$ to denote

$$(X, \alpha) \models \; v = w \;\wedge\; (X, \alpha) \models \; t = u$$

**Definition 3.4** (coequations)**.** A *set of coequations* is a subautomaton $D \subseteq 2^{A^*}$ of the automaton $(2^{A^*}, \tau)$. We define $(X, c, \alpha) \models D$ —and say: *the coloured automaton* $(X, c, \alpha)$ *satisfies* $D$— by

$$(X, c, \alpha) \models D \;\Leftrightarrow\; \forall x \in X,\; o_c(x) \in D.$$

Because

$$\forall x \in X,\; o_c(x) \in D \;\Leftrightarrow\; \mathsf{im}(o_c) \subseteq D,$$

we have, equivalently, that $(X, c, \alpha) \models D$ iff the observability map $o_c$ factors through $D$:

$$\begin{tikzcd}
 & & 2 \\
X \arrow[rru, bend left, "c"] \arrow[r, "h"] \arrow[rr, bend right, "o_c"'] & D \arrow[ru, bend left, "\varepsilon?"] \arrow[r, "i"] & 2^{A^*} \arrow[u, "\varepsilon?"']
\end{tikzcd}$$

where the homomorphisms (of coloured automata) $h$ and $i$ are given by

$$h(x) = o_c(x) \quad \text{and} \quad i(L) = L,$$

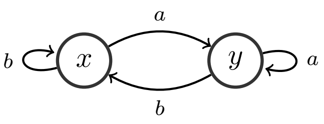
respectively. We define $(X, \alpha) \models D$ —and say: *the automaton* $(X, \alpha)$ *satisfies* $D$— by

$$\begin{aligned}(X, \alpha) \models D \;&\Leftrightarrow\; \forall c\colon X \to 2, \quad (X, c, \alpha) \models D \\ &\Leftrightarrow\; \forall c\colon X \to 2,\, \forall x \in X,\; o_c(x) \in D.\end{aligned}$$

□

**Example 3.5.** We consider the automaton $(Z, \gamma)$ defined by the following diagram:

$(Z, \gamma) \;=$



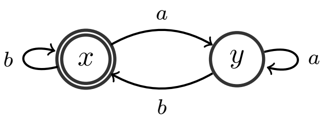
Here are some examples of equations:

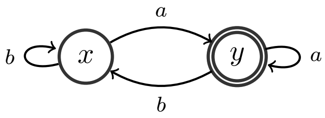
$$\begin{aligned}(Z, x, \gamma) \;&\models\; \{b = \varepsilon,\, ab = \varepsilon,\, aa = a\} \\ (Z, y, \gamma) \;&\models\; \{a = \varepsilon,\, ba = \varepsilon,\, bb = b\}\end{aligned}$$

Taking the intersection of the (bisimulation equivalences generated by) these sets, we obtain that

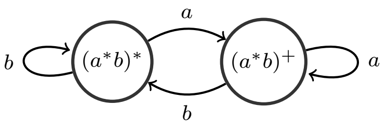
$$(Z, \gamma) \models \{aa = a,\, bb = b,\, ab = b,\, ba = a\}$$

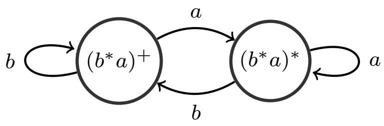
The above set of equations or, again more precisely, the bisimulation equivalence relation on $(A^*, \sigma)$ generated by it, is the largest set of equations satisfied by $(Z, \gamma)$. For examples of coequations, we consider the following 2 (out of all 4 possible) coloured versions of $(Z, \gamma)$:

$(Z, c, \gamma) \ =$ 

$(Z, d, \gamma) \ =$ 

(Thus $c(x) = 1$, $c(y) = 0$, $d(x) = 0$ and $d(y) = 1$.) The observability mappings $o_c$ and $o_d$ map these automata to

$\mathsf{im}(o_c) \ =$ 

$\mathsf{im}(o_d) \ =$ 

It follows that

$$(Z, c, \gamma) \models \ \{(a^*b)^*, \ (a^*b)^+\} \qquad (Z, d, \gamma) \models \ \{(b^*a)^*, \ (b^*a)^+\}$$

□

### 3.2.2 Free and cofree automata

Let $(X, \alpha)$ be an automaton, then, on one hand, we show how to construct an automaton that corresponds to the *largest set of equations* satisfied by $(X, \alpha)$. And, on the other hand, we show how to construct an automaton that corresponds to the *smallest set of coequations* satisfied by $(X, \alpha)$.

**Definition 3.6.** Let $X = \{x_i \mid i \in I\}$ be the set of states of an automaton $(X, \alpha)$. We define a pointed automaton $\mathsf{free}(X, \alpha)$ in two steps, as follows:

(i) First, we take the product of the pointed automata $(X, x_i, \alpha)$ that we obtain by letting the initial element $x_i$ range over $X$. This yields a pointed automaton $(\Pi X, \bar{x}, \bar{\alpha})$ with

$$\Pi X = \prod_{x\colon 1 \to X} X_x \ \cong \ X^{|X|},$$

(where $X_x = X$), with $\bar{x} = (x_i)_{i \in I}$, and with $\bar{\alpha}\colon \Pi X \to (\Pi X)^A$ defined component-wise

$$\bar{\alpha}((y_i)_{i \in I})(a) = ((y_i)_a)_{i \in I}.$$

(ii) Next we consider the reachability map $r_{\bar{x}}\colon A^* \to \Pi X$ and define:

$$\mathsf{Eq}(X, \alpha) = \ \ker(r_{\bar{x}}), \qquad \mathsf{free}(X, \alpha) = \ A^*/\mathsf{Eq}(X, \alpha).$$

This yields the pointed automaton $(\mathsf{free}(X, \alpha), [\varepsilon], [\sigma])$:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[dr, bend left, "{[\varepsilon]}"] \arrow[drr, bend left, "\bar{x}"] & & \\
A^* \arrow[r] \arrow[rr, bend right, "r_{\bar{x}}"'] & \mathsf{free}(X,\alpha) \arrow[r] & \Pi X
\end{tikzcd}$$

Note that $\mathsf{free}(X,\alpha) \cong \mathsf{im}(r_{\bar{x}})$. □

**Definition 3.7.** Let $X = \{x_i \mid i \in I\}$ be the set of states of an automaton $(X,\alpha)$. We define a coloured automaton $\mathsf{cofree}(X,\alpha)$ in two steps, as follows:

(i) First, we take the coproduct of the $2^{|X|}$ coloured automata $(X,c,\alpha)$ that we obtain by letting $c$ range over the set $X \to 2$ of all colouring functions. This yields a coloured automaton $(\Sigma X, \hat{c}, \hat{\alpha})$ with
$$\Sigma X = \sum_{c\colon X \to 2} X_c,$$
(where $X_c = X$), and with $\hat{c}$ and $\hat{\alpha}$ defined component-wise.

(ii) Next we consider the observability map $o_{\hat{c}}\colon \Sigma X \to 2^{A^*}$ and define:
$$\mathsf{coEq}(X,\alpha) = \mathsf{im}(o_{\hat{c}}), \qquad \mathsf{cofree}(X,\alpha) = \mathsf{coEq}(X,\alpha).$$
This yields the coloured automaton $(\mathsf{cofree}(X,\alpha), \varepsilon?, \tau)$:

$$\begin{tikzcd}
 & & 2 \\
\Sigma X \arrow[r] \arrow[urr, bend left, "\hat{c}"] \arrow[rr, bend right, "o_{\hat{c}}"'] & \mathsf{cofree}(X,\alpha) \arrow[r] \arrow[ur, bend left, "\varepsilon?"] & 2^{A^*} \arrow[u, "\varepsilon?"']
\end{tikzcd}$$

Note that $\mathsf{cofree}(X,\alpha) \cong \Sigma X / \ker(o_{\hat{c}})$. □

The automata $\mathsf{free}(X,\alpha)$ and $\mathsf{cofree}(X,\alpha)$ are *free* and *cofree* on $(X,\alpha)$, respectively, because of the following universal properties:

$$\begin{tikzcd}
1 \arrow[d] \arrow[dr, bend left, "\forall x"] & & & & 2 \\
\mathsf{free}(X,\alpha) \arrow[r, "\exists!"'] & (X,\alpha) & & (X,\alpha) \arrow[ur, bend left, "\forall c"] \arrow[r, "\exists!"'] & \mathsf{cofree}(X,\alpha) \arrow[u]
\end{tikzcd}$$

For every point $x\colon 1 \to X$ there exists a unique homomorphism from $\mathsf{free}(X,\alpha)$ to $(X,x,\alpha)$, given by the "$x$-th" projection from the product $\Pi X$ to $X$. Dually, for every colouring $c\colon X \to 2$, there exists a unique homomorphism from $(X,c,\alpha)$ to $\mathsf{cofree}(X,\alpha)$, given by the "$c$-th" embedding of $X$ into the coproduct $\Sigma X$.

The main raison d'être for the constructions of $\mathsf{free}$ and $\mathsf{cofree}$ is that they represent the sets $\mathsf{Eq}(X,\alpha)$ and $\mathsf{coEq}(X,\alpha)$, which are, by construction, the largest set of equations and the smallest set of coequations satisfied by $(X,\alpha)$.

**Proposition 3.8.** The set $\mathsf{Eq}(X, \alpha)$ is the *largest* set of equations satisfied by $(X, \alpha)$:

$$\mathsf{Eq}(X,\alpha) \quad = \quad \bigcup \{ E \subseteq A^* \times A^* \mid \ E \text{ is a set of equations and } \ (X,\alpha) \models E \ \}.$$

The set $\mathsf{coEq}(X, \alpha)$ is the *smallest* set of coequations satisfied by $(X, \alpha)$:

$$\mathsf{coEq}(X,\alpha) \quad = \quad \bigcap \{ D \subseteq 2^{A^*} \mid \ D \text{ is a set of coequations and } \ (X,\alpha) \models D \ \}.$$

□

**Proposition 3.9.** The set of equations $\mathsf{Eq}(X, \alpha)$ is a congruence on $A^*$.

*Proof.* We already know that $\mathsf{Eq}(X, \alpha)$ is a right-congruence. Let $(v, w) \in \mathsf{Eq}(X, \alpha)$ and $u \in A^*$. For a state $x \in X$, we have

$$x_{uv} = (x_u)_v = (x_u)_w = x_{uw}$$

(since equations $(v, w) \in \mathsf{Eq}(X, \alpha)$ hold in *all* states of $X$). It follows that $(uv, uw) \in \mathsf{Eq}(X, \alpha)$ and we conclude that $\mathsf{Eq}(X, \alpha)$ is a congruence. □

Applying the picture above to the minimal automaton $\langle L \rangle$ of a given language $L \in 2^{A^*}$ we obtain the following refinement of (3.5):

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[drr, bend left] \arrow[drrrr, bend left, "L"] & & & & \\
A^* \arrow[r, "r_2"'] \arrow[rr, bend left, "r_1"] \arrow[drrrr, bend right, "L"'] & \mathsf{free}\langle L \rangle \arrow[r] & \langle L \rangle \arrow[r] \arrow[drr, bend right] & \mathsf{cofree}\langle L \rangle \arrow[r] & 2^{A^*} \arrow[d, "\varepsilon?"] \\
& & & & 2
\end{tikzcd} \tag{3.6}$$

We already saw in (3.5) that $\ker(r_1) = \equiv_{\mathsf{MN}}$, the Myhill-Nerode equivalence for $L$. Furthermore, it follows from Proposition 3.8 and Proposition 3.9 that

$$\mathsf{Eq}\langle L \rangle \ = \ \ker(r_2) \ = \ \equiv_L \tag{3.7}$$

where $\equiv_L$ is the so-called *syntactic congruence* of $L$, which is defined, for all $v, w \in A^*$, by

$$v \equiv_L w \quad \text{ if and only if } \quad \forall u_1, u_2 \in A^*, \ (u_1 v u_2 \in L \Leftrightarrow u_1 w u_2 \in L)$$

**Corollary 3.10.** *For a language $L \in 2^{A^*}$, the congruences $\mathsf{Eq}\langle L \rangle$ and $\equiv_L$ coincide.*

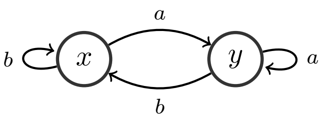
*Proof.* Let $(v, w) \in \mathsf{Eq}\langle L \rangle$ and let $u_1$ be an arbitrary word in $A^*$. The language $L_{u_1}$ is in $\langle L \rangle$ and satisfies the equation $L_{u_1 v} = L_{u_1 w}$, that is, for any word $u_2 \in A^*$,

$$(u_2 \in L_{u_1 v} \Leftrightarrow u_2 \in L_{u_1 w}) \qquad \text{equivalently,} \qquad (u_1 v u_2 \in L \Leftrightarrow u_1 w u_2 \in L)$$

that is, $(v, w) \in \equiv_L$. The other inclusion is proved similarly. □

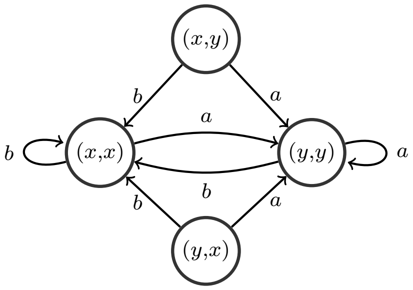
**Example 3.11** (Example 3.5 continued)**.** We consider our previous example

$(Z, \gamma) \;=$



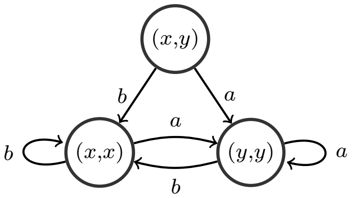
The product of $(Z, x, \gamma)$ and $(Z, y, \gamma)$ is:

$(\Pi Z, (x, y), \gamma) \;=$



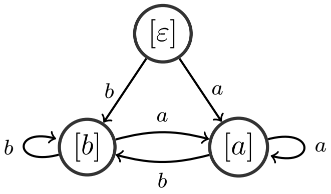
Taking $\mathsf{im}(r_{(x,y)})$ yields the part that is reachable from $(x, y)$:

$\mathsf{im}(r_{(x,y)}) \;=$



We know that $\mathsf{free}(Z, \gamma) \cong \mathsf{im}(r_{(x,y)})$, which leads to the following isomorphic automaton:

$\mathsf{free}(Z, \gamma) \;=$



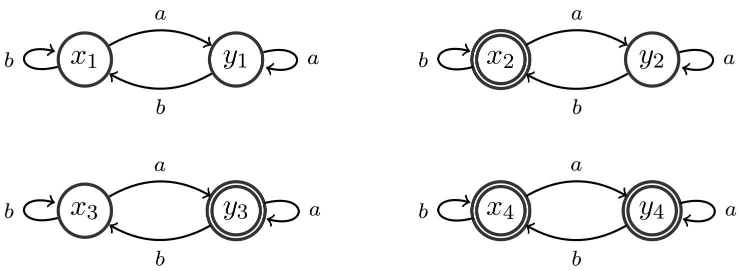
Since $\mathsf{free}(Z, \gamma) = A^*/\mathsf{Eq}(Z, \gamma)$, we can deduce from the above automaton that $\mathsf{Eq}(Z, \gamma)$ consists of

$$\mathsf{Eq}(Z, \gamma) = \{aa = a,\ bb = b,\ ab = b,\ ba = a\}$$

where the set on the right represents the smallest bisimulation equivalence – in fact, a congruence – on $(A^*, \sigma)$. The set $\mathsf{Eq}(Z, \gamma)$ is the largest set of equations satisfied by $(Z, \gamma)$.

Next we turn to coequations. The coproduct of all 4 coloured versions of $(Z, \gamma)$ is

$(\Sigma Z, \hat{c}, \hat{\gamma}) \;=$

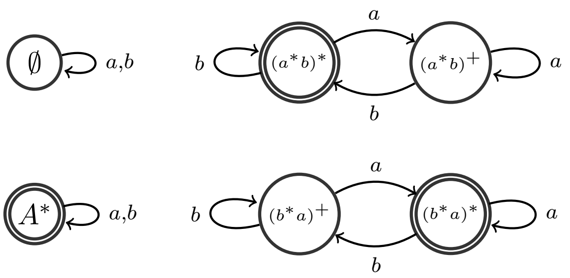


The observability map $o_{\hat{c}} \colon \Sigma Z \to 2^{A^*}$ is given by

| $o_{\hat{c}}(x_1)$ | $o_{\hat{c}}(y_1)$ | $o_{\hat{c}}(x_2)$ | $o_{\hat{c}}(y_2)$ | $o_{\hat{c}}(x_3)$ | $o_{\hat{c}}(y_3)$ | $o_{\hat{c}}(x_4)$ | $o_{\hat{c}}(y_4)$ |
|---|---|---|---|---|---|---|---|
| $\emptyset$ | $\emptyset$ | $(a^*b)^*$ | $(a^*b)^+$ | $(b^*a)^+$ | $(b^*a)^*$ | $A^*$ | $A^*$ |

Since $\mathsf{cofree}(Z,\gamma) = \mathsf{im}(o_{\hat{c}})$, this yields

$\mathsf{cofree}(Z,\gamma) \;=$



(3.8)

The set of states of this automaton is $\mathsf{cofree}(Z,\gamma)$, which is the smallest set of coequations satisfied by $(Z,\gamma)$. □

Summarising the present section, we have obtained, for every automaton $(X,\alpha)$, the following refinement of our previous scene (3.4):

$$\begin{tikzcd}
1 \arrow[d] \arrow[dr, bend left] \arrow[drr, bend left, "\forall x"] & & & & 2 \\
A^* \arrow[r] & \mathsf{free}(X,\alpha) \arrow[r, dashed] & (X,\alpha) \arrow[r, dashed] \arrow[urr, bend left, "\forall c"] & \mathsf{cofree}(X,\alpha) \arrow[r] \arrow[ur, bend left] & 2^{A^*} \arrow[u]
\end{tikzcd} \tag{3.9}$$

The automata $\mathsf{free}(X,\alpha)$ and $\mathsf{cofree}(X,\alpha)$ represent the largest set of equations and the smallest set of coequations satisfied by $(X,\alpha)$. As we mentioned earlier, all of this applies to *infinite* $X$ as well.

## 3.3 A dual equivalence

In this section, we shall first show that —when suitably restricted— both constructions of $\mathsf{free}$ and $\mathsf{cofree}$ are in fact functorial, that is, they act not only on automata but also on homomorphisms. Next we shall see that by restricting the functors $\mathsf{free}$ and $\mathsf{cofree}$ further still, they turn out to form a dual equivalence.

We will be using the following categories:

$\mathcal{A}$: the category of automata $(X,\alpha)$ and automata homomorphisms;

$\mathcal{A}_m$: the category of automata $(X,\alpha)$ and automata monomorphisms;

$\mathcal{A}_e$: the category of automata $(X,\alpha)$ and automata epimorphisms.

As it turns out, we can extend the definitions of $\mathsf{free}$ and $\mathsf{cofree}$ to monomorphisms and epimorphisms, respectively, such that we obtain functors of the following type:

$$\mathsf{free} \colon \mathcal{A}_m \to (\mathcal{A}_e)^{\mathrm{op}}, \qquad \mathsf{cofree} \colon \mathcal{A}_e \to (\mathcal{A}_m)^{\mathrm{op}}.$$

Here the superscript op indicates a reversal of arrows: for monomorphisms,

$$\begin{tikzcd}(X,\alpha) \arrow[r, "m"] & (Y,\beta) & {} \arrow[r, mapsto, "\mathsf{free}"] & {} & \mathsf{free}(Y,\beta) \arrow[r, "\mathsf{free}(m)"] & \mathsf{free}(X,\alpha),\end{tikzcd}$$

where $\mathsf{free}(m)$ is defined by

$$\mathsf{free}(m)(\,[w]_{\mathsf{Eq}(Y,\beta)}\,) \;=\; [w]_{\mathsf{Eq}(X,\alpha)}.$$

Because $m$ is a monomorphism, we have $\mathsf{Eq}(Y,\beta) \subseteq \mathsf{Eq}(X,\alpha)$, which implies that $\mathsf{free}(m)$ is a well-defined epimorphism. Similarly, for epimorphisms,

$$\begin{tikzcd}(X,\alpha) \arrow[r, "e"] & (Y,\beta) & {} \arrow[r, mapsto, "\mathsf{cofree}"] & {} & \mathsf{cofree}(Y,\beta) \arrow[r, "\mathsf{cofree}(e)"] & \mathsf{cofree}(X,\alpha),\end{tikzcd}$$

where $\mathsf{cofree}(e)$ is just set inclusion. Because $e$ is an epimorphism, we have $\mathsf{coEq}(Y,\beta) \subseteq \mathsf{coEq}(X,\alpha)$, which implies that $\mathsf{cofree}(e)$ is a well-defined monomorphism.

### 3.3.1 Main theorems

**Congruence quotients**

Next we introduce the category $\mathcal{C}$ of *congruence quotients*, which is defined as follows:

$$\begin{aligned}\mathsf{objects}(\mathcal{C}) &= \{(A^*/C,[\sigma]) \mid C \subseteq A^* \times A^* \text{ is a congruence relation}\},\\ \mathsf{arrows}(\mathcal{C}) &= \{e\colon A^*/C \to A^*/D \mid e \text{ is an epimorphism of automata}\},\end{aligned}$$

where $[\sigma]$ corresponds to the quotient transition derived from the transition introduced in 3.2. We observe that $\mathcal{C}$ is a subcategory of $\mathcal{A}_e$ and that it is in fact a set: $\mathcal{C}$ is isomorphic to the set of all congruence relations on $A^*$, together with set inclusion. That is, there exists a (unique) epimorphism $e\colon A^*/C \to A^*/D$ if and only if $C \subseteq D$.

Since congruence quotients come equipped with a canonical choice of transition function, that is, $[\sigma]$, we shall often simply write $A^*/C$ for $(A^*/C,[\sigma])$.

**Theorem 3.12.** $\mathsf{free}(\mathcal{A}_m) = \mathcal{C}^{\mathrm{op}}$

*Proof.* For every automaton $(X,\alpha)$, $\mathsf{free}(X,\alpha) = A^*/\mathsf{Eq}(X,\alpha)$ is a congruence, by Proposition 3.9. For the reverse inclusion, consider a congruence $C \subseteq A^* \times A^*$. One readily shows that

$$\mathsf{Eq}(A^*/C) = C,$$

which implies $\mathsf{free}(A^*/C) \;=\; A^*/\mathsf{Eq}(A^*/C) \;=\; A^*/C$. This proves the theorem for objects. For arrows, we already saw that $\mathsf{free}$ maps a monomorphism to an epimorphism of congruence quotients. Conversely, let $e\colon A^*/C \to A^*/D$ be an epimorphism. We define

$$m\colon A^*/D \to \;(A^*/C + A^*/D),$$

where $+$ denotes the disjoint union of automata. Because

$$\mathsf{Eq}(A^*/C + A^*/D) \;=\; C \cap D,$$

and because $C \subseteq D$, it follows that $\mathsf{free}(A^*/C + A^*/D) \;=\; A^*/(C \cap D) \;=\; A^*/C$, which implies that $\mathsf{free}(m) = e$. □

### Preformations of languages

We will be using the following notion of a *preformation of languages.*

**Definition 3.13.** A preformation of languages is a set $V \subseteq 2^{A^*}$ such that:

(i) $V$ is a complete atomic Boolean subalgebra of $2^{A^*}$;

(ii) for all $L \in 2^{A^*}$: if $L \in V$ then for all $a \in A$, both $L_a \in V$ and ${}_aL \in V$.

We note that, being a subalgebra of $2^{A^*}$, a preformation $V$ always contains both $\emptyset$ and $A^*$. □

Next we define the *category* $\mathcal{PL}$ of preformations of languages, as follows:

$$\begin{aligned} \mathsf{objects}(\mathcal{PL}) &= \{(V,\tau) \mid V \subseteq 2^{A^*} \text{ is a preformation of languages}\}, \\ \mathsf{arrows}(\mathcal{PL}) &= \{m\colon V \to W \mid m \text{ is an monomorphism of automata}\}. \end{aligned}$$

where $\tau$ corresponds to the transition introduced in 3.3 of language derivatives. The category $\mathcal{PL}$ is a subcategory of $\mathcal{A}_m$; furthermore, $\mathcal{PL}$ is in fact a set and the arrows in $\mathcal{PL}$ are just set inclusion. Since preformations of languages come equipped with a canonical choice of transition function, that is, $\tau$ (right-derivatives of languages), we shall often simply write $V$ for $(V,\tau)$.

The main result of this subsection will be that

$$\mathsf{cofree}(\mathcal{C}) = (\mathcal{PL})^{\mathrm{op}},$$

which we shall prove in several steps.

We begin with an elementary but useful property of colourings, which uses the following definition. For an automaton $(X,\alpha)$ and state $x \in X$, we define the following ("one-point") colouring:

$$\delta_x\colon X \to 2, \qquad \delta_x(y) = 1 \iff x = y.$$

**Lemma 3.14.** *For every automaton $(X,\alpha)$, state $y \in X$ and colouring $c\colon X \to 2$,*

$$o_c(y) = \bigcup \{\, o_{\delta_x}(y) \mid \ x \in X \ \textit{and}\ c(x) = 1 \ \}.$$

The states of congruence quotients are equivalence classes of words $w \in A^*$, that is, languages $[w] \subseteq A^*$. The following lemma shows that each of them occurs as the observable behaviour of the inital state $[\varepsilon]$, under the corresponding one-point colouring.

**Lemma 3.15.** *For every congruence quotient $A^*/C \in \mathcal{C}$ and every $[w] \in A^*/C$,*

$$o_{\delta_{[w]}}([\varepsilon]) = [w].$$

*Proof.* For all $v \in A^*$,

$$v \in o_{\delta_{[w]}}([\varepsilon]) \iff \delta_{[w]}([\varepsilon]_v) = 1 \iff [\varepsilon]_v = [w] \iff [v] = [w] \iff v \in [w].$$

□

The following lemma shows that all the observable behaviour of a congruence quotient stems from its initial state.

**Lemma 3.16.** *For every congruence quotient $A^*/C \in \mathcal{C}$ and every $L \in \mathsf{coEq}(A^*/C)$, there exists a colouring $c\colon A^*/C \to 2$ such that*

$$o_c([\varepsilon]) = L.$$

*Proof.* If $L \in \mathsf{coEq}(A^*/C)$ then there exist a state $[w] \in A^*/C$ and a colouring $d\colon A^*/C \to 2$ with $o_d([w]) = L$. We define a new colouring $c\colon A^*/C \to 2$, for all $[v] \in A^*/C$, by

$$c([v]) = d([w]_v).$$

Note that $c$ is well-defined because $C$ is a (left) congruence on $A^*$. It now follows that

$$v \in o_c([\varepsilon]) \;\Leftrightarrow\; c([\varepsilon]_v) = 1 \;\Leftrightarrow\; c([v]) = 1 \;\Leftrightarrow\; d([w]_v) = 1 \;\Leftrightarrow\; v \in o_d([w]) \;\Leftrightarrow\; v \in L,$$

which concludes the proof. □

Combining the above, we obtain the following characterisation.

**Proposition 3.17.** For every congruence quotient $A^*/C \in \mathcal{C}$,

$$\mathsf{coEq}(A^*/C) = \{\, L \in 2^{A^*} \mid L = \bigcup V \text{ for some } V \subseteq A^*/C \,\}.$$

Therefore,

$$\mathsf{coEq}(A^*/C) \cong \mathcal{P}(A^*/C).$$

*Proof.* There is a trivial one-to-one correspondence between colourings

$$c\colon A^*/C \to 2,$$

and subsets $V \subseteq A^*/C$ given by $V_c = c^{-1}(1)$. Using Lemma 3.14 and Lemma 3.15, we obtain, as a consequence, that

$$\begin{aligned} o_c([\varepsilon]) &= \bigcup\{\, o_{\delta_K}([\varepsilon]) \mid K \in A^*/C \text{ and } c(K) = 1 \,\} \\ &= \bigcup\{\, K \mid K \in A^*/C \text{ and } c(K) = 1 \,\} \\ &= \bigcup V_c. \end{aligned}$$

The first equality of the proposition now follows from Lemma 3.16. Since the languages $L \in A^*/C$ form a partitioning of $A^*$, the second identity (isomorphism) follows. □

We are ready to prove the following.

**Proposition 3.18.** For every congruence quotient $A^*/C \in \mathcal{C}$, $\mathsf{coEq}(A^*/C)$ is a preformation of languages with $A^*/C$ as the set of atoms.

*Proof.* It follows from Proposition 3.17 that $\mathsf{coEq}(A^*/C)$ is a complete atomic Boolean algebra, with $A^*/C$ as the set of atoms, and containing $A^*$ and $\emptyset$.

Because $\mathsf{coEq}(A^*/C)$ is a subautomaton of $(2^{A^*}, \tau)$, it is closed under right derivatives.

In order to prove that it is also closed under left derivatives, consider $L \in \mathsf{coEq}(A^*/C)$ and $w \in A^*$. By Lemma 3.16, there exists a colouring $c\colon A^*/C \to 2$ with $L = o_c([\varepsilon])$. We define a new colouring $c_w\colon A^*/C \to 2$, for $[v] \in A^*/C$, by

$$c_w([v]) = c([vw]).$$

(Note that $c_w$ is well-defined because $C$ is a (left) congruence on $A^*$.) Because

$$v \in o_{c_w}([\varepsilon]) \Leftrightarrow c_w([v]) = 1 \Leftrightarrow c([vw]) = 1 \Leftrightarrow vw \in L \Leftrightarrow v \in {}_wL,$$

it follows that $o_{c_w}([\varepsilon]) = {}_wL$. And because $o_{c_w}([\varepsilon])$ is in $\mathsf{coEq}(A^*/C)$, so is ${}_wL$. □

Still on our way towards a proof of $\mathsf{cofree}(\mathcal{C}) = (\mathcal{PL})^{\mathrm{op}}$, let us next fix a preformation of languages $V \in \mathcal{PL}$ and show that it is the image under $\mathsf{cofree}$ of a congruence quotient on $A^*$. To this end, we define the following mapping:

$$\eta \colon A^* \to \mathsf{At}(V) \qquad \eta(w) = \text{ the unique atom } L \in V \text{ with } w \in L.$$

Since $V$ is a complete atomic Boolean algebra containing $A^*$, $\eta$ is well-defined and surjective. We shall show next that it is a congruence quotient of $A^*$.

**Lemma 3.19.** *The set* $\mathsf{ker}(\eta)$ *is a congruence on* $A^*$ *and hence* $\eta$ *is a congruence quotient*

$$\eta \colon (A^*, \sigma) \to (\mathsf{At}(V), [\sigma]).$$

*Proof.* It suffices to show that, for all $v, w \in A^*$, if $\eta(v) = \eta(w)$ then, for all $u \in A^*$,

$$\eta(uv) = \eta(uw) \quad \text{and} \quad \eta(vu) = \eta(wu)$$

In order to prove the first equality, we assume $\eta(v) = \eta(w)$ and consider $\eta(uv)$. Because $uv \in \eta(uv)$ we have $v \in \eta(uv)_u$. Because $V$ is closed under right derivatives, $\eta(uv)_u \in V$ and because $V$ is atomic, we have $\eta(v) \subseteq \eta(uv)_u$. We have the following sequence of implications:

$$\begin{aligned} \eta(v) \subseteq \eta(uv)_u \quad &\Rightarrow \quad \eta(w) \subseteq \eta(uv)_u \\ &\Rightarrow \quad w \in \eta(uv)_u \\ &\Rightarrow \quad uw \in \eta(uv) \\ &\Rightarrow \quad \eta(uw) \subseteq \eta(uv). \end{aligned}$$

The same argument will prove $\eta(uv) \subseteq \eta(uw)$, which proves the first equality. The second equality follows by the same argument, using left instead of right derivatives. □

There is also the following.

**Lemma 3.20.** $\mathsf{Eq}(V) = \mathsf{ker}(\eta)$.

*Proof.* We have to show, for all $v, w \in A^*$, that

$$(\text{ for all } L \in V : L_v = L_w\ ) \Leftrightarrow \eta(v) = \eta(w).$$

From $\varepsilon \in \eta(v)_v = \eta(v)_w$ it follows that $w \in \eta(v)$ and hence $\eta(v) = \eta(w)$, which proves the above implication from left to right.

For the implication from right to left, assume $\eta(v) = \eta(w)$. Since $V$ is a complete atomic Boolean algebra, it suffices to prove that $L_v = L_w$ for $L \in \mathsf{At}(V)$, since (right) derivatives commute with unions. So consider $u \in A^*$ and $\eta(u) \in \mathsf{At}(V)$. For all $x \in A^*$,

$$x \in \eta(u)_v \Rightarrow vx \in \eta(u) \Rightarrow \eta(u) = \eta(vx) \Rightarrow \eta(u) = \eta(wx) \Rightarrow x \in \eta(u)_w,$$

where the last but one implication follows from Lemma 3.19. This proves $\eta(u)_v \subseteq \eta(u)_w$. The same argument proves the reverse inclusion, which concludes the proof. □

Combining the two lemma's above now gives the following.

**Proposition 3.21.** $\mathsf{free}(V) = (\mathsf{At}(V), [\sigma])$.

*Proof.*

$$\mathsf{free}(V) \;=\; (A^*/\mathsf{Eq}(V), [\sigma]) \;=\; (A^*/\mathsf{ker}(\eta), [\sigma]) \;=\; (\mathsf{At}(V), [\sigma])$$

□

**Corollary 3.22.** $\mathsf{cofree} \circ \mathsf{free}(V) = V$.

*Proof.* By Proposition 3.21, $\mathsf{cofree} \circ \mathsf{free}(V) = \mathsf{cofree}(\mathsf{At}(V), [\sigma])$. And by Proposition 3.17, $\mathsf{cofree}(\mathsf{At}(V), [\sigma]) = V$. □

Finally, we obtain the main result of this subsection.

**Theorem 3.23.** $\mathsf{cofree}(\mathcal{C}) = (\mathcal{PL})^{\mathrm{op}}$.

*Proof.* The identity holds for objects, by Proposition 3.18 and Corollary 3.22. Furthermore, every epimorphism of congruence quotients is mapped by $\mathsf{cofree}$ to the reversed inclusion of the corresponding preformations, and conversely, every inclusion of preformations is easily seen to stem from an epimorphism of congruence quotients. □

### free and cofree form a dual equivalence

We have obtained the following dual equivalence.

**Theorem 3.24.** *The category $\mathcal{C}$ of congruence quotients is dually equivalent to the category $\mathcal{PL}$ of preformations of languages via the functors* $\mathsf{free}$ *and* $\mathsf{cofree}$. *That is,*

$$\mathsf{cofree}\colon\ \mathcal{C} \;\simeq\; (\mathcal{PL})^{\mathrm{op}}\ :\mathsf{free}$$

*Proof.* For a preformation of languages $V$,

$$\mathsf{cofree} \circ \mathsf{free}(V) \;=\; V,$$

by Corollary 3.22. For a congruence quotient $A^*/C$, we have

$$\mathsf{free} \circ \mathsf{cofree}(A^*/C) \;=\; \mathsf{At}(\mathsf{cofree}(A^*/C)) \;=\; A^*/C,$$

by Proposition 3.21 and Proposition 3.18, respectively. This proves the theorem for objects. One readily shows that this correspondence extends to arrows as well. □

As a consequence of our Theorem 3.24 we deduce the following corollary.

**Corollary 3.25.** *For every congruence $C$ in $A^*$ and every language $L$ in $2^{A^*}$,*

$$L \in \mathsf{coEq}(A^*/C) \qquad \Leftrightarrow \qquad C \subseteq \mathsf{Eq}\langle L\rangle.$$

*Proof.* If $L \in \mathsf{coEq}(A^*/C)$, then $\langle L\rangle$ is completely included in $\mathsf{coEq}(A^*/C)$. By Theorems 3.12 and 3.24 there exists an epimorphism from $A^*/C$ to $\mathsf{free}\langle L\rangle$, that is, $C \subseteq \mathsf{Eq}\langle L\rangle$. On the contrary, if $C \subseteq \mathsf{Eq}\langle L\rangle$, there exists an epimorphism from $A^*/C$ to $\mathsf{free}\langle L\rangle$. By Theorem 3.24, $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is completely included in $\mathsf{coEq}(A^*/C)$. Recall that the colouring $\delta_L\colon A^*/\mathsf{Eq}\langle L\rangle \to 2$, given by $\delta_L([w]) = 1$ iff $w \in L$, is a well-defined function and the equation $o_{\delta_L}([\varepsilon]) = L$ holds, therefore $L \in \mathsf{coEq}(A^*/C)$. □

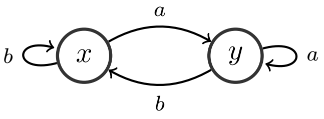
**Corollary 3.26.** *Let $L$ be a language in $2^{A^*}$, then $L \in \mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$.*

### 3.3.2 Illustrating the duality

We illustrate the duality Theorem 3.24 with some examples.

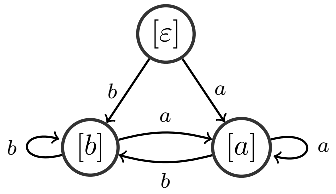
**Example 3.27** (Example 3.11, continued)**.** We consider our previous example

$(Z, \gamma) =$



for which we had computed

$\mathsf{free}(Z, \gamma) =$

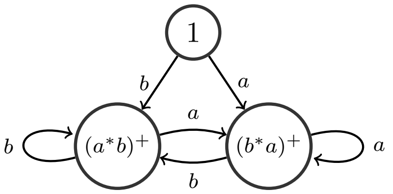


We recall that the transition structure of the automaton $\mathsf{free}(Z, \gamma)$ is inherited from the automaton $(A^*, \sigma)$ and hence satisfies

$$[w] \xrightarrow{a} [wa].$$

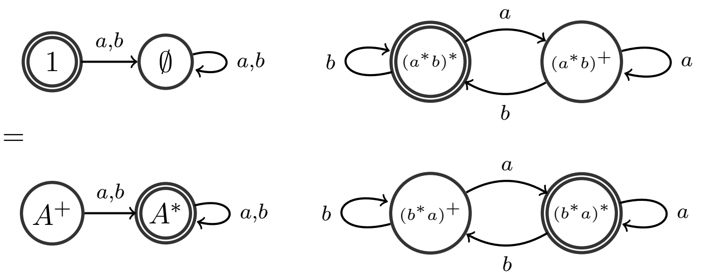
(In particular, transitions between these states are *not* given, as in $(2^{A^*}, \tau)$, by right derivatives.) By Lemma 3.15, each of the languages $[\varepsilon]$, $[a]$ and $[b]$ can be explicitly computed as the behaviour of the initial state $[\varepsilon]$, under the corresponding one-point colouring. This gives:

$\mathsf{free}(Z, \gamma) =$



By a computation similar to the one in Example 3.11, we obtain

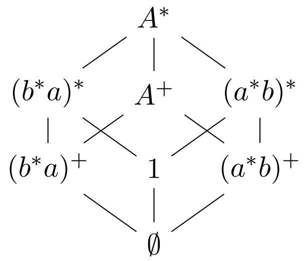
$\mathsf{cofree} \circ \mathsf{free}(Z, \gamma) =$



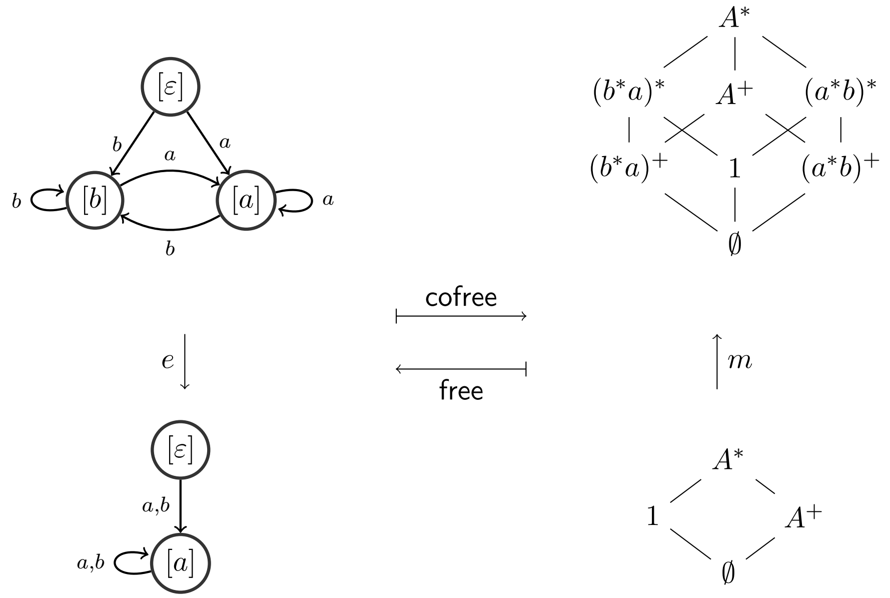
(3.10)

By Proposition 3.18, the automaton $\mathsf{cofree} \circ \mathsf{free}(X, \alpha)$ is a preformation of languages. In particular, it is a Boolean subalgebra of $2^{A^*}$, which we can represent as follows (indicating language inclusion by edges):

$\mathsf{cofree} \circ \mathsf{free}(Z, \gamma) \;=$ (3.11)



(Note that $\mathsf{cofree} \circ \mathsf{free}(Z, \gamma) \cong \mathcal{P}(\mathsf{free}(Z, \gamma))$. Since $\mathsf{free} \circ \mathsf{cofree} \circ \mathsf{free} = \mathsf{free}$, we obtain the following picture, in which we have included an example of an epimorphism $e$ and its image, to illustrate the action of free and cofree on arrows:



(Although it is made superfluous by the duality theorem, it is an interesting little exercise to apply free to the automaton $\mathsf{cofree} \circ \mathsf{free}(Z, \gamma)$ 'by hand', that is, by using the definition of free.) □

**Example 3.28.** Here is an example of an application of the duality Theorem 3.24 to a language that is not regular. Let $A = \{a, b\}$ and let, for $w \in A^*$,

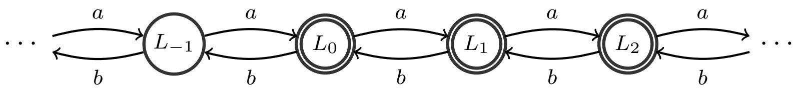
$$\begin{array}{lcl} |w|_a & = & \text{number of } a\text{'s occurring in } w; \\ |w|_b & = & \text{number of } b\text{'s occurring in } w. \end{array}$$

We consider the context-free language $L$ defined by

$$L = \{\, w \in A^* \mid |w|_a \geq |w|_b \,\}.$$

Its minimal automaton $\langle L \rangle$, which is the smallest subset of $2^{A^*}$ that contains $L$ and is closed under right derivatives, looks as follows:

$\langle L \rangle \;=$



where $L_n = \{ w \in A^* \mid |w|_a + n \geq |w|_b \}$, for all $n \in \mathbb{Z}$. If we define a transition function $\alpha\colon \mathbb{Z} \to \mathbb{Z}^A$ by $n_a = n+1$ and $n_b = n-1$, then we obtain an isomorphism $\langle L \rangle \cong (\mathbb{Z}, \alpha)$. It is easy to see that $\mathsf{free}\langle L \rangle \cong \langle L \rangle$. If we next define a transition function $\beta\colon \mathcal{P}(\mathbb{Z}) \to \mathcal{P}(\mathbb{Z})^A$, for all $K \subseteq \mathbb{Z}$, by

$$K_a = K + 1 = \{n+1 \mid n \in K\}, \qquad K_b = K - 1 = \{n-1 \mid n \in K\}.$$

then it follows that $\mathsf{cofree}\langle L \rangle \cong (\mathcal{P}(\mathbb{Z}), \beta)$. □

**Example 3.29.** In this example, which is taken from [29], we shall illustrate how the duality Theorem 3.24 can be used for the equational definition of interesting classes of languages. Let $A = \{a, b\}$ and let $\mathbf{ab{=}ba}$ denote the smallest congruence on $A^*$ containing the equation $(ab, ba)$. It is easy to prove that, for all $v, w \in A^*$,

$$(v, w) \in \mathbf{ab{=}ba} \;\Leftrightarrow\; |v|_a = |w|_a \text{ and } |v|_b = |w|_b.$$

As a consequence, languages $[w]$ in the congruence quotient $A^*/\mathbf{ab{=}ba}$ satisfy

$$[w] = \{ v \in A^* \mid v \text{ is a permutation of } w \},$$

(with the usual definition of permutation of words). By the duality Theorem 3.24, we have that $V = \mathsf{cofree}(A^*/\mathbf{ab{=}ba})$ is a preformation of languages. We now call a language $L$ *commutative* whenever $L \in V$. This terminology is justified by the following equivalences:

$$\begin{aligned} L \in V \;&\Leftrightarrow\; L \text{ is the union of permutation equivalence classes } [w] \\ &\Leftrightarrow\; \langle L \rangle \models ab = ba. \end{aligned}$$

The first equivalence follows from the fact that $V$ is a preformation with atoms $[w]$; the second from the fact that $\mathsf{free}(V) = A^*/\mathbf{ab{=}ba}$, whence $\mathsf{Eq}(V) = \mathbf{ab{=}ba}$. □

### 3.3.3 Equational bisimulations

This section introduces the notion of equational bisimulation and we show how it can be used to prove that a language satisfies a given set of equations. First of all, recall the following property on coloured bisimulations (see Chapter 2 and [62]), which follows from the fact that $2^{A^*}$ is a final $(2 \times F)$-coalgebra.

**Proposition 3.30** ([62])**.** Let $R \subseteq 2^{A^*} \times 2^{A^*}$ be a coloured bisimulation on $(2^{A^*}, \varepsilon?, \tau)$. If $(K, L) \in R$ then $K = L$.

It follows that no non-trivial coloured bisimulation can be defined on $2^{A^*}$. The above property is often called the coinduction proof method: in order to show that $K = L$, it suffices to define a coloured bisimulation $R$ with $(K, L) \in R$. We refer to [62] for examples that illustrate the usefulness of this proof method. In [23], it is shown how variations on the above proof method lead to surprisingly efficient algorithms for proving the equivalence of non-deterministic finite automata.

Here we generalise the notion of bisimulation for languages as follows. Let $C \subseteq A^* \times A^*$ be a congruence. We call a relation $R \subseteq 2^{A^*} \times 2^{A^*}$ an *equational bisimulation* with respect to $C$, or $C$*-bisimulation* for short, if, for all $(K, L) \in R$,

(i) $\varepsilon \in K \;\Leftrightarrow\; \varepsilon \in L$;

(ii) $\forall (v,w) \in C,\ (K_v, L_w) \in R$.

We have the following corresponding proof principle.

**Proposition 3.31.** Let $C \subseteq A^* \times A^*$ be a congruence and let $R \subseteq 2^{A^*} \times 2^{A^*}$ be a $C$-bisimulation. For all $(K,L) \in R$,

(i) $K = L$;

(ii) $\langle K \rangle \models C$.

*Proof.* Since $(a,a) \in C$, for all $a \in A$, any $C$-bisimulation is trivially also an ordinary bisimulation. Thus (1) follows from Proposition 3.30. For (2), let $(K,L) \in R$ and consider any state $K_u \in \langle K \rangle$ and any pair $(v,w) \in C$. Since $(K,K) = (K,L) \in R$ and $R$ is a $C$-bisimulation, and since $(uv, uw) \in C$, it follows that $(K_{uv}, K_{uw}) \in R$. By (1), we have $K_{uv} = K_{uw}$ and thus $(K_u)_v = (K_u)_w$, which proves (2). □

**Example 3.32.** Let $K = aA^* + b(a^*b)^* + b(b^*a)^+$. We shall use Proposition 3.31 to show that $K$ is commutative. Referring to Example 3.29, we need to prove that $\langle K \rangle \models ab = ba$. Let

$$M = A^*, \qquad N = (a^*b)^* + (b^*a)^+, \qquad O = (a^*b)^+ + (b^*a)^*,$$

and let

$$R = \{\langle K, K \rangle\} \cup \{M, N, O\}^2$$

Then $R$ is an (**ab=ba**)-bisimulation. Thus $\langle K \rangle \models ab = ba$, by Proposition 3.31. □

**Example 3.33.** For a next example, we return to the context-free language of Example 3.28:

$$L = \{\, w \in A^* \mid |w|_a \geq |w|_b \,\},$$

and show that also $L$ is commutative. Let $L_n = \{\, w \in A^* \mid |w|_a + n \geq |w|_b \,\}$ and let

$$S = \{\langle L_n, L_n \rangle \mid n \in \mathbb{Z}\,\}.$$

Then $S$ is an (**ab=ba**)-bisimulation and thus $\langle L \rangle \models ab = ba$, by Proposition 3.31. □

## 3.4 Discussion and future work

The algebra-coalgebra duality of diagram (3.1) is a modern rendering of the duality between *reachability* and *observability* of automata [10, 9], which ultimately goes back to Kalman's duality between controllability and observability in system theory [44, 45]. Our work builds on [22] and [21], using the combined algebra-coalgebra perspective on automata that was used there to give a new proof and various generalisations of Brzozowski's [26] minimisation algorithm. Our work is remotely related to [5], where the same perspective plays a role, albeit in a rather different manner. None of these papers, however, —nor for that matter any other paper we know of— discusses the relation between equations and coequations for automata.

In algebraic language theory (cf. [31, 32, 55, 58]), regular languages are typically studied in terms of so-called *syntactic* monoids and congruences. For every set, there is the monoid $(X^X, \cdot, 1_X)$ defined by

$$X^X = \{\phi \mid \phi\colon X \to X\,\}, \qquad 1_X(x) = x, \qquad \phi \cdot \psi = \psi \circ \phi.$$

It can be used to define for every automaton $(X, \alpha)$ a pointed automaton

$$(X^X, 1_X, \tilde{\alpha}), \qquad \tilde{\alpha}(\phi)(a)(x) = \phi(x)_a,$$

where $\phi(x)_a = \alpha(\phi(x))(a)$, as usual. Now the *transition monoid* [58] for $(X, \alpha)$:

$$(\mathsf{trans}(X, \alpha), 1_X, \tilde{\alpha}),$$

is defined by $\mathsf{trans}(X, \alpha) = \mathsf{im}(r_{1_X})$, where $r_{1_X}$ is the reachability map of $(X^X, 1_X, \tilde{\alpha})$:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[r, bend left, "1_X"] \arrow[rr, bend left, "1_X"] & & \\
A^* \arrow[r] \arrow[rr, bend right, "r_{\bar{x}}"'] & \mathsf{trans}(X, \alpha) \arrow[r] & X^X
\end{tikzcd}$$

**Theorem 3.34.** *For an automaton $(X, \alpha)$,*

$$(\mathsf{free}(X, \alpha), \bar{x}, \bar{\alpha}) \cong (\mathsf{trans}(X, \alpha), 1_X, \tilde{\alpha}).$$

*Proof.* Let $X = \{x_1, \ldots, x_n\}$. For every $\bar{y} \in \mathsf{free}(X, \alpha)$ we define

$$\phi_{\bar{y}} \colon X \to X \qquad \phi_{\bar{y}}(x_i) = y_i$$

Then $\phi(\bar{y}) = \phi_{\bar{y}}$ defines an isomorphism of pointed automata. □

We have defined $\mathsf{free}(X, \alpha)$ by using the product space $\Pi X$ rather than the function space $X^X$, because it allows us to define the automaton $\mathsf{cofree}(X, \alpha)$ using the coproduct $\Sigma X$. As a consequence, $\mathsf{cofree}(X, \alpha)$ can be seen as the dual of $\mathsf{free}(X, \alpha)$ or, equivalently, of the transition monoid. If $(X, \alpha) = \langle L \rangle$, the minimal automaton for a fixed language $L \in 2^{A^*}$:

$$\begin{tikzcd}
1 \arrow[d, "\varepsilon"'] \arrow[r, bend left, "1_X"] & \\
A^* \arrow[r, "r_{1_X}"'] & \mathsf{free}\langle L \rangle \cong \mathsf{trans}\langle L \rangle
\end{tikzcd}$$

then the kernel of the reachability map $r_{1_X}$ is the syntactic congruence $\equiv_L$ of $L$, as we already observed in (3.6) and (3.7). Interestingly, the fact that $\mathsf{free}(X, \alpha)$ carries a monoid structure (which it inherits from the concatenation of words in $A^*$) does not play any role in our proof of the duality between $\mathsf{free}$ and $\mathsf{cofree}$.

The way we have obtained the dual equivalence, namely, as a restriction of the (more generally defined) constructions of $\mathsf{free}$ and $\mathsf{cofree}$ – or, in other words, the constructions of the syntactic monoid and its dual – seems to be new. For the case of *finite automata*, our duality as such coincides with the use of Stone duality in [37, Theorem 1]. There all automata $\mathcal{A} = (Q, A, \delta, I, F)$ are finite. Consequently, the *language recognised by* $\mathcal{A}$, denoted $L(\mathcal{A})$, is regular. For a finite alphabet $A$, the concatenation operation on $A^*$ gives rise to a *residuated family* of operations on the set of all languages of $A^*$ as follows. *Complex concatenation* on $\mathcal{P}(A^*)$ is given by

- $KL = \{uv \mid u \in K \text{ and } v \in L\}$

The *residuals* of this operation are uniquely determined by the *residuation laws*:

- $\forall K, L, M \in \mathcal{P}(A^*) \qquad KM \subseteq L \;\Leftrightarrow\; M \subseteq K\backslash L \;\Leftrightarrow\; K \subseteq L/M$

In particular, for any word $w \in A^*$, the following operations coincide

$$\{w\}\backslash L \,=\, L_w \qquad \text{and} \qquad L/\{w\} \,=\, {}_wL$$

We will make use of the following Lemmas about regular languages.

**Lemma 3.35** ([37, Proposition 1])**.** *If $L$ is a regular language over $A$, then the set of right and left derivatives $\{{}_yL_x \mid x, y \in A^*\}$ is finite.*

**Lemma 3.36** ([15, Proposition 2.14])**.** *If $L$ is a regular language over $A$, then every class $[w]$ in $A^*/\mathsf{Eq}\langle L\rangle$ can be expressed as follows*

$$[w] = \bigcap\{{}_yL_x \mid w \in {}_yL_x\} \setminus \bigcup\{{}_yL_x \mid w \notin {}_yL_x\}. \tag{3.12}$$

**Definition 3.37** ([37, Definition 4])**.** Let $A$ be a finite alphabet and $L \subseteq A^*$ a language over $A$. Let $\mathcal{B}(L)$ be the Boolean subalgebra of $\mathcal{P}(A^*)$ generated by the set $\{{}_yL_x \mid x, y \in A^*\}$. We will call $\mathcal{B}(L)$ the *quotienting ideal* generated by $L$. More generally a quotienting ideal of $\mathcal{P}(A^*)$ is a Boolean subalgebra which is closed under the quotienting operations $(\;)_x$ and ${}_y(\;)$ for all $x, y \in A^*$.

The following theorem is one of the most important results of [37].

**Theorem 3.38** ([37, Theorem 1])**.** *Let $L$ be a language recognised by an automaton. The extended dual of the Boolean algebra with additional operations $(\mathcal{B}(L), \backslash, /)$ is the syntactic monoid of $L$. In particular, it follows that the syntactic monoid of $L$ is finite and is effectively computable.*

The next proposition states that the dual object to the syntactic monoid of a regular language $L$ coincides with the preformation of languages $\mathsf{cofree} \circ \mathsf{free}\langle L\rangle$ we described in the present paper. To prove it, we will use a previous lemma stating that every class $[w]$ in $A^*/\mathsf{Eq}\langle L\rangle$ belongs to the Boolean algebra generated by the set $\{{}_yL_x \mid x, y \in A^*\}$.

**Proposition 3.39.** For a regular language $L$ over an alphabet $A^*$, the Boolean algebra with additional operations $(\mathcal{B}(L), \backslash, /)$ and $\mathsf{cofree} \circ \mathsf{free}\langle L\rangle$ coincide.

*Proof.* We will use the following abbreviation $V = \mathsf{cofree} \circ \mathsf{free}\langle L\rangle$. Let $(v, w) \in \mathsf{Eq}\langle L\rangle$, then for all $x \in A^*$, we have that $L_{xv} = L_{xw}$. Therefore, for $y \in A^*$ we deduce the equations ${}_yL_{xv} = {}_yL_{xw}$. It follows that $\mathsf{Eq}\langle L\rangle \subseteq \mathsf{Eq}\langle {}_yL_x\rangle$. By Corollary 3.25 ${}_yL_x$ is included in $V$. Since $V$ is a variety of languages, we conclude that $\mathcal{B}(L)$ is included in $V$. Recall that for any pair of languages $K, M$ in $2^{A^*}$, the equations $K\backslash M = \bigcap_{w\in K} M_w$ and $M/K = \bigcap_{w\in K} {}_wM$ hold. Hence, $V$ is closed under residuals. Now, let $[u]$ be an element in $A^*/\mathsf{Eq}\langle L\rangle$. Since $L$ is regular, every atom in $V$ can be defined according to Lemmas 3.35 and 3.36 using finitely many Boolean operations. Thus, it belongs to the Boolean algebra generated by the derivatives of $L$. It follows that $V$ is included in $\mathcal{B}(L)$. □

Recall that, for regular languages, the set $\mathcal{B}(L)$ is a finite lattice and it is, therefore, complete and atomic. We can say that, for finite automata, our duality coincides with that obtained by Stone duality in Theorem 3.38. It is interesting to note that our result emerges from an structural study of automata and, so far, no direct appeal to Stone duality is required.

The present chapter already contains some contributions that encourage us to continue working along these lines. The first relevant insight is that we are able to deal with *infinite* automata and non-regular languages. It lies in the fact that the duality we have found is the (conceptually simpler) discrete duality between sets and complete atomic Boolean algebras. The latter duality is also used in [61], where it was lifted to a dual equivalence between deterministic automata and the so-called Boolean automata. We hope to retrieve some of the results presented in the papers [6, 7, 56, 38, 36, 37], specially Reiterman's characterisation in terms of profinite equations. Further limit constructions of non-necessarily finite monoids need to be investigated.

A second useful approach we have presented here is the categorical description of the duality presented in Theorem 3.24 and its more manageable Corollary 3.25. Because we are working within the algebra-coalgebra duality, we can use both algebraic notions, such as congruence, and coalgebraic notions, such as bisimulations. Our notion of equational bisimulation, which is a generalisation of the standard notion, seems to be new and so does the corresponding coinduction proof principle. Within this context of the algebra-coalgebra duality, we also want to study the notions of varieties and covarieties of automata. In [65], some initial results are mentioned but with the present duality in place, we expect that more can be said. The notion of equational bisimulation and its corresponding coinduction proof principle deserve further study, both the present instance for automata and its coalgebraic generalisations.

Finally, it would be interesting to investigate to what extent our duality can be further generalised to other dynamical systems, such as Moore automata and probabilistic automata. The algebra-coalgebra duality as such has already been extended to such automata in [22, 21], leading to generalisations of Brzozowszki's algorithm. In [69], for example, the authors developed an extension based on our Theorem 3.24 for the case of weighted automata. Weighted automata are a generalisation of non-deterministic automata introduced by Schützenberger [70]. Every transition carries a cost of its execution, this being an element in a semiring. Thus, paths labelled with words also carry a total cost representing resources, probabilities or time of execution. In addition, we plan to study the connections with [18] and [4], where dualities for generalised rational structures have been studied.

I reconeixes una força antiga
i sense discussió t'hi entregaràs
i furgaràs els seus racons
per revelar el poder que s'hi amaga.

Teresa Rampell — Manel

# CHAPTER 4

# Eilenberg's theorem revisited

## 4.1 Introduction

One of the most important results in the algebraic study of formal languages and automata is Eilenberg's variety theorem [32] establishing a one-to-one correspondence between varieties of regular languages, which are classes of regular languages closed under Boolean operations, derivatives, and preimages under monoid morphisms, and varieties of finite monoids, which are classes of finite monoids closed under finite products, submonoids, and homomorphic images. At the heart of Eilenberg's variety theorem lie the characterisation of varieties of regular languages by their syntactic monoids and the closure properties of the corresponding classes of finite monoids.

Several extensions of Eilenberg's theorem, obtained by replacing monoids by other algebraic structures [17] or by modifying the closure properties on the definition of variety of languages, are known in the literature. In this context, we mention a local version of Eilenberg's theorem proved by Gehrke, Grigorieff, and Pin [38] working with a fixed finite alphabet and considering only regular languages on it, and the extension of this result up to the level of an abstract duality of categories by Adámek, Milius, Myers, and Urbat [4].

Another further step in this research programme is to replace varieties of finite monoids by the more general notion of formation, that is, a class of finite monoids closed under taking epimorphic images and finite subdirect products. Formations of finite groups are important for a better understanding of the structure of finite groups, and the more general notion of formation of algebraic structures, introduced and studied by Shemetkov and Skiba in [72], plays a central role in universal algebra. Therefore it seems quite natural to seek an Eilenberg type theorem establishing a connection between formations of finite monoids and formations of regular languages, which are classes of regular languages closed under Boolean operations and derivatives with a weaker property on the closure under inverse monoid morphism. This was established in [15]. The weaker closure conditions for formations lead to more possibilities than for varieties as more general classes of languages can be described and understood.

Our principal aim here is to extend the main theorem of [15] to the level of general monoids. Our results are motivated by the significant role played by formations of non-necessarily finite groups in the structural study of the groups and some interesting families of non-regular languages that have recently appeared in the literature.

The main result of this Chapter is an Eilenberg type theorem which states a bijection between formations of non-necessarily finite monoids and formations of non-necessarily regular languages. This result is the most general correspondence known to us. The approach we use benefits from the dual equivalence presented in the previous Chapter. In fact, this Chapter provides one of the best possible examples of the expressiveness of the functors free and cofree. The coalgebraic approach used in this result highlights the fundamental role of duality in algebraic automata theory. Furthermore, this dual equivalence generalises a recent line of work that shows the correspondence between local varieties of regular languages and local pseudovarieties of monoids [38]. This result is called the *local* Eilenberg's theorem in [4].

Our approach depends heavily on the notion of a formation of congruences, which is a function that assings every alphabet $A$ to a filter on the set of all congruences on $A^*$ closed under taking kernels of surjective monoid homomorphisms. We prove that there is a bijective correspondence between formations of monoids and formations of congruences (Theorem 4.16), and that formations of languages are in a one-to-one correspondence with formations of congruences (Theorem 4.19).

As an interesting case, we extend the original result by Eilenberg to varieties of monoids. Although similar, the notion of pseudovariety differs from the notion of variety introduced by Birkhoff [19]. A class of monoids is a variety of monoids if it is closed under taking substructures, quotients and (not necessarily finite) products. Thus, infinite objects are allowed in a variety.

We end the Chapter by showing an example of an application to relatively disjunctive languages. The generalised disjunctive languages have been considered by some authors in the literature, e.g., Guo, Reis, and Thierrin ([41, 81, 58]). A language $L$ is relatively disjunctive if there exists a dense language intersecting finitely many times on each class of the syntactic congruence associated to $L$. It has been shown in [49] that this condition is equivalent to $L$ having a non relatively regular syntactic monoid, that is, a monoid not containing a finite ideal. In this paper, we prove that the set of all non-r-disjunctive languages is a formation of languages and consequently, we see that it is Boolean algebra closed under derivatives.

Accordingly, Section 4.2 recovers the various definitions on the original Eilenberg's variety theorem and we present an alternative definition for variety of regular languages founded on equations and coequations. Section 4.3 contains our main results, Theorem 4.16 and Theorem 4.19. It also contains an example of an application of our main theorems to relatively disjunctive languages and a discussion on the particular case of varieties of monoids. Finally, in Section 4.4 we relate our work to other existing results.

## 4.2 Eilenberg's variety theorem

Eilenberg's variety theorem [32] is a fundamental result in computer science. It underscores the importance of *varieties of finite monoids* or *pseudovarieties* in the study of regular languages. We recall in this sections the definitions used in the classical Eilenberg's variety theorem. We do this to better understand the results we will prove below.

**Definition 4.1.** A *variety of finite monoids*, or *pseudovariety*, is a set of finite monoids $\mathbf{H}$ satisfying the following conditions:

(i) every homomorphic image of a monoid of $\mathbf{H}$ belongs to $\mathbf{H}$;

(ii) every submonoid of a monoid of $\mathbf{H}$ belongs to $\mathbf{H}$;

(iii) the direct product of a finite family of monoids of $\mathbf{H}$ also belongs to $\mathbf{H}$.

**Definition 4.2.** A *variety of regular languages* is a function $\mathcal{H}$ that assigns to every alphabet $A$ a set of regular languages satisfying

(i) for each alphabet $A$, $\mathcal{H}(A)$ is closed under Boolean operations and derivatives;

(ii) if $L$ is a language of $\mathcal{H}(B)$, then for each monoid homomorphism $\varphi\colon A^* \to B^*$ the language $\varphi^{-1}(L)$ belongs to $\mathcal{H}(A)$.

Recall that the syntactic morphism is just the quotient homomorphism $\eta\colon B^* \to \mathsf{free}\langle L\rangle$ (See Corollary 3.10). However, at first sight, no other relation with monoids seems to appear in the definition of variety of regular languages. Despite this, it was Eilenberg [32] who proved the following striking theorem.

**Theorem 4.3** ([32])**.** *There is a one-to-one correspondence between varieties of finite monoids and varieties of regular languages.*

In order to prove the above result, Eilenberg associates to each variety of finite monoids $\mathbf{H}$, the set $\mathcal{H}(A)$ of all recognisable languages of $A^*$ whose syntactic monoid belongs to $\mathbf{H}$. Conversely, to each variety of regular languages $\mathcal{H}$, he associates the variety of finite monoids $\mathbf{H}$ generated by the syntactic monoids of every regular language $L$ in $\mathcal{H}(A)$, for certain alphabet $A$. These constructions define mutually inverse bijective correspondences between varieties of finite monoids and varieties of regular languages.

With this in mind, we will next give an alternative characterisation of varieties of regular languages based on equations and coequations. Thus, we introduce the following definition.

**Definition 4.4.** An *EC-variety of regular languages* is a function $\mathcal{H}$ that assigns to every alphabet $A$ a set of regular languages satisfying the following conditions:

(i)' for each alphabet $A$, if $L$ is a language in $\mathcal{H}(A)$, then $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is included in $\mathcal{H}(A)$;

(ii)' for each alphabet $A$, if $\mathsf{coEq}(A^*/C_1)$, $\mathsf{coEq}(A^*/C_2)$ are included in $\mathcal{H}(A)$, then so is $\mathsf{coEq}(A^*/C_1 \cap C_2)$;

(iii)' for every two alphabets $A$ and $B$, if $L$ is a language in $\mathcal{H}(B)$ and $\eta\colon B^* \to \mathsf{free}\langle L\rangle$ denotes the quotient homomorphism, then for each monoid homomorphism $\varphi\colon A^* \to B^*$, the set $\mathsf{coEq}(A^*/\mathsf{ker}(\eta \circ \varphi))$ belongs to $\mathcal{H}(A)$.

Here, EC stands for equations-coequations. The above definition underscores the importance of congruences in the study of these kind of correspondences; moreover, its connection with varieties of finite monoids seems much more natural. Our main result in this subsection states that varieties of regular languages and EC-varieties of regular languages are equivalent notions. This result can be regarded as an equational-coequational version of Eilenberg's theorem. The proof of our result depends on the Lemmas 3.35 and 3.36.

**Theorem 4.5.** *Let $\mathcal{H}$ be a function that assigns to every alphabet $A$ a set of regular languages. Then, $\mathcal{H}$ is a variety of regular languages if and only if it is an EC-variety of regular languages.*

*Proof.* We start showing that every variety of regular languages $\mathcal{H}$ is an EC-variety of regular languages.

(i)' Let $L$ be a regular language in $\mathcal{H}(A)$ and consider a class $[w]$ in $A^*/\mathsf{Eq}\langle L\rangle$ then, by Lemmas 3.35 and 3.36, this class is a finite Boolean combination of languages in $\mathcal{H}(A)$. It follows that $[w]$ is an element in $\mathcal{H}(A)$. Recall that every language $K \in \mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is expressed as $K = \bigcup_{w\in K}[w]$ by Proposition 3.17. Recall that this union is finite as $L$ is regular and there are finitely many such classes $[w]$. Hence, as every atom $[w]$ is included in $\mathcal{H}(A)$, we conclude that $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is completely included in $\mathcal{H}(A)$.

(ii)' Now, assume that $\mathsf{coEq}(A^*/C_1)$ and $\mathsf{coEq}(A^*/C_2)$ are included in $\mathcal{H}(A)$. Then, for any word $w \in A^*$, the languages $[w]_{C_1}$ and $[w]_{C_2}$ are regular languages in $\mathcal{H}(A)$. As $\mathcal{H}(A)$ is a Boolean algebra, the language $[w]_{C_1\cap C_2} = [w]_{C_1} \cap [w]_{C_2}$ is in $\mathcal{H}(A)$. As every atom $[w]_{C_1\cap C_2}$ is included in $\mathcal{H}(A)$, so is $\mathsf{coEq}(A^*/C_1 \cap C_2)$.

(iii)' Now, let $L$ be a language in $\mathcal{H}(B)$, let $\eta\colon B^* \to \mathsf{free}\langle L\rangle$ denote the quotient homomorphism, and let $\varphi\colon A^* \to B^*$ be a monoid homomorphism. As $L$ is a language in $\mathcal{H}(B)$ then, for all pair of words $x, y \in B^*$, the language ${}_yL_x$ is in $\mathcal{V}(B)$. It follows that $\varphi^{-1}({}_yL_x)$ is a language in $\mathcal{H}(A)$, for all $x, y \in B^*$. By item EC(i) in this proof, the preformation of languages $\mathsf{coEq}(A^*/\mathsf{Eq}\langle {}_yL_x\rangle)$ is completely included in $\mathcal{H}(A)$. As $L$ is regular, there are finitely many such derivatives ${}_yL_x$. Therefore, by several applications of item EC(ii), the set $\mathsf{coEq}(A^*/\bigcap_{x,y\in B^*}\mathsf{Eq}\langle {}_yL_x\rangle)$ is included in $\mathcal{H}(A)$. We claim that $\bigcap_{x,y\in B^*}\mathsf{Eq}\langle {}_yL_x\rangle$ is included in $\mathsf{ker}(\eta\circ\varphi)$. Assume towards a contradiction that a pair $(v, w)$ of $\bigcap_{x,y\in B^*}\mathsf{Eq}\langle {}_yL_x\rangle$ is not included in $\mathsf{ker}(\eta\circ\varphi)$, therefore there exists some $x \in B^*$ with $L_{x\varphi(v)} \neq L_{x\varphi(v)}$. Therefore, we can find some $y \in B^*$ with $y \in L_{x\varphi(v)}$ such that $y \notin L_{x\varphi(w)}$. We have the following chain of implications

$$\begin{array}{ccccc} y \in L_{x\varphi(v)} & \Rightarrow & x\varphi(v)y \in L & \Rightarrow & \varphi(v) \in {}_yL_x \\ & \Rightarrow & v \in \varphi^{-1}({}_yL_x) & \Rightarrow & \varepsilon \in [\varphi^{-1}({}_yL_x)]_v \\ & \Rightarrow & \varepsilon \in [\varphi^{-1}({}_yL_x)]_w & \Rightarrow & w \in \varphi^{-1}({}_yL_x) \\ & \Rightarrow & \varphi(w) \in {}_yL_x & \Rightarrow & x\varphi(w)y \in L \\ & \Rightarrow & y \in L_{x\varphi(w)}, & & \end{array}$$

that is, we have obtained a contradiction. It follows that $\bigcap_{x,y\in B^*}\mathsf{Eq}\langle {}_yL_x\rangle$ is included in $\mathsf{ker}(\eta\circ\varphi)$. Using the duality Theorem 3.24, we conclude that $\mathsf{coEq}(A^*/\mathsf{ker}(\eta\circ\varphi))$ is included in $\mathsf{coEq}(A^*/\bigcap_{x,y\in B^*}\mathsf{Eq}\langle {}_yL_x\rangle)$ and therefore it is included in $\mathcal{H}(A)$.

Now, assume that $\mathcal{H}$ is an EC-variety of regular languages and let us show that it is a variety of regular languages.

(i) Let $L$ be a language in $\mathcal{H}(A)$, then $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is in included in $\mathcal{H}(A)$. Note that the complement of $L$ is in $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ and so is every derivative. Now, let $L$ and $K$ be two languages in $\mathcal{H}(A)$. Then $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ and $\mathsf{coEq}(A^*/\mathsf{Eq}\langle K\rangle)$ are included in $\mathcal{H}(A)$ and, therefore, so is $\mathsf{coEq}(A^*/D)$ for $D = \mathsf{Eq}\langle L\rangle \cap \mathsf{Eq}\langle K\rangle$. As $D \subseteq \mathsf{Eq}\langle L\rangle$, by Corollary 3.25, we conclude that $L$ is in $\mathsf{coEq}(A^*/D)$. With a similar prove we conclude that $K$ also belongs to $\mathsf{coEq}(A^*/D)$. Therefore $K \cap L$ and $K \cup L$ are both included in $\mathsf{coEq}(A^*/D)$ as it is a preformation of languages.

(ii) Let $L$ be a language in $\mathcal{H}(B)$ and consider a monoid homomorphism $\varphi\colon A^* \to B^*$. If $\eta\colon B^* \to \mathsf{free}\langle L\rangle$ denotes the quotient homomorphism, we have that $\mathsf{coEq}(A^*/\mathsf{ker}(\eta\circ\varphi))$ is included in $\mathcal{H}(A)$. Consider the following coloration on $A^*/\mathsf{ker}(\eta\circ\varphi)$

$$\begin{array}{rccl} \xi\colon & A^*/\mathsf{ker}(\eta\circ\varphi) & \longrightarrow & 2 \\ & [w] & \longmapsto & \begin{cases} 1, & \text{if } \varphi(w) \in L; \\ 0, & \text{if } \varphi(w) \notin L. \end{cases} \end{array}$$

In fact, if $(v, w)$ is a pair in $\mathsf{ker}(\eta\circ\varphi)$, then $(\varphi(v), \varphi(w))$ belongs to $\mathsf{Eq}\langle L\rangle$. In particular,

$$\varphi(w) \in L \quad \Leftrightarrow \quad \varepsilon \in L_{\varphi(w)} \quad \Leftrightarrow \quad \varepsilon L_{\varphi(v)} \quad \Leftrightarrow \quad \varphi(v) \in L.$$

Thus, the colouring $\xi$ is well-defined. Now, the language $o_\xi([\varepsilon])$ is a language in $\mathsf{coEq}(A^*/\mathsf{ker}(\eta\circ\varphi))$ and, consequently, it belongs to $\mathcal{H}(A)$. Note that

$$v \in o_\xi([\varepsilon]) \quad \Leftrightarrow \quad \xi([\varepsilon]_v) = 1 \quad \Leftrightarrow \quad \varphi(v) \in L \quad \Leftrightarrow \quad v \in \varphi^{-1}(L).$$

□

From the last theorem and the Eilenberg's theorem we obtain, as direct consequence, the following corollary.

**Corollary 4.6.** *There is a one-to-one correspondence between varieties of finite monoids and EC-varieties of regular languages.*

## 4.3 Eilenberg's formation theorem

### 4.3.1 Formations

In this section we define the notions of formations that we will use in what follows. For the sake of simplicity, we write $(A^*/C)$ instead of $(A^*/C, [\sigma])$ and $\langle L\rangle$ instead of $(\langle L\rangle, \tau)$.

**Formations of monoids**

**Definition 4.7.** Following [40, p. 78], we say that a monoid $M$ is a *subdirect product* of a product of a family of monoids $(M_i)_{i\in I}$ if $M$ is a submonoid of the direct product $\prod_{i\in I} M_i$ and each induced projection $\pi_i$ from $M$ onto $M_i$ is surjective. A monoid $P$ which is isomorphic to such a submonoid $M$ is also called a subdirect product of the family of monoids $(M_i)_{i\in I}$. In this case, the projections *separate* the elements of $M$, that is, if $\pi_i(x) = \pi_i(y)$ for all $i \in I$, then $x = y$.

In fact, we have the following proposition.

**Proposition 4.8** ([40, Proposition 3.1])**.** A monoid $M$ is a subdirect product of a family of monoids $(f_i\colon M_i)_{i\in I}$ if and only if there is a family of surjective morphisms $(M \to M_i)_{i\in I}$ which separates the elements of $M$.

Subdirect products allow us to introduce the notion of formation of monoids, which is a particular case of the most general notion of formation of algebraic structures, introduced and studied by Shemetkov and Skiba in [72].

**Definition 4.9.** A *formation of monoids* is a set of monoids $\mathbf{F}$ satisfying:

(i) if $A \in \mathbf{F}$ and $B \cong A$, then $B \in \mathbf{F}$;

(ii) every quotient of a monoid of $\mathbf{F}$ also belongs to $\mathbf{F}$;

(iii) the subdirect product of a finite family of monoids of $\mathbf{F}$ also belongs to $\mathbf{F}$.

We present some examples of interesting formations of monoids.

**Example 4.10.**

1. Any pseudovariety of monoids $\mathbf{H}$ is a formation of monoids. In particular, the pseudovariety of all finite monoids, denoted by $\mathbf{Fin}$, is a formation of monoids.

2. If $\mathbf{F}$ is a formation of monoids, then $\mathbf{F}_\omega$ defined as the class of all monoids in $\mathbf{F}$ that are finite is again a formation.

3. We say that a monoid $M$ has a *zero* if there exists an element $0 \in M$, such that for every element $m \in M$, the equation $m0 = 0m = 0$ holds. Such an element is unique and thus, one speaks of *the* zero element. The class $\mathbf{Z}$ of all monoids with zero is a formation of monoids.

4. A monoid $M$ is called *relatively regular* (*r-regular* for short) (see [49]) if it contains a finite ideal. The class $\mathbf{R}$ of all r-regular monoids is a formation of monoids. Usual integers with multiplication $(\mathbb{Z}, \cdot, 1)$ is $r$-regular as it is a monoid with zero. The set $\mathbb{Z}^*$ of nonzero integers is a submonoid of $\mathbb{Z}$ without finite ideals. Therefore, $\mathbf{R}$ is not closed under substructures.

5. A monoid $M$ is called *cyclic* if it is generated by one element $m \in M$. That is $M$ consists of all powers $m^k$ of $m$ (here we use the notation $m^0 = 1$). If all these powers are distinct, then $M$ is isomorphic to the additive monoid of all natural numbers $(\mathbb{N}, +, 0)$. For a finite cyclic monoid $M = \langle m \rangle$ there is a smallest number $n$ with the property $m^n = m^k$, for some $k > n$; $n$ is called the *index* of the element $m$ (of $M$). In this connection, if $r$ is the smallest nonzero number with the property $m^n = m^{n+r}$, then $r$ is called the *period* of $m$ (of $M$). The pair $(n, r)$ is called the *type* of $m$ (of $M$). For any type $(n, r)$ with $n, r \in \mathbb{N}$ and $r \geq 1$, the relation:

$$\theta_{n,r} = \Delta_{\mathbb{N}} \cup \{(p, q) \in \mathbb{N} \times \mathbb{N} \mid p, q \geq n \text{ and } p \equiv q \mod r\}$$

is a congruence on $\mathbb{N}$. The resulting quotient $\mathbb{N}/\theta_{n,r}$ is a finite cyclic monoid with type $(n, r)$. Every finite cyclic monoid $M$ with type $(n, r)$ is isomorphic to the quotient $\mathbb{N}/\theta_{n,r}$. A monoid is called *periodic* if all its cyclic submonoids are finite. The class $\mathbf{P}$ of all periodic monoids is a formation of monoids. A monoid $M$ is called *aperiodic* if there exists a natural number $k \in \mathbb{N}$ satisfying $m^k = m^{k+1}$ for all $m \in M$. Obviously, aperiodic monoids are periodic. The class $\mathbf{A}$ of all aperiodic monoids is also a formation of monoids.

6. A *locally finite* monoid is a monoid in which every finitely generated submonoid is finite. Obviuosly, locally-finite monoids are periodic. The converse is false: there are even torsion groups that are not locally finite (see Burnside problem). The class of all

locally-finite monoids, denoted by $\mathbf{L_{Fin}}$ is a formation of monoids. In general, if $\mathbf{F}$ is a formation of finite monoids. A *locally* $\mathbf{F}$ monoid is a monoid in which every finitely generated submonoid belongs to $\mathbf{F}$. The class $\mathbf{L_F}$ of all locally $\mathbf{F}$ monoids is a formation of monoids.

**Definition 4.11.** For a monoid $M$, its *residual* with respect to a formation of monoids $\mathbf{F}$, written $C^M_{\mathbf{F}}$, is defined as

$$C^M_{\mathbf{F}} = \bigcap\{C \in \mathsf{Con}(M) \mid M/C \in \mathbf{F}\}.$$

The above family is not empty as the total relation $\nabla_M$ is always included.

*Remark.* In general, the quotient $M/C^M_{\mathbf{F}}$ does not necessarily belongs to the formation $\mathbf{F}$. In fact, for the formation $\mathbf{Z}_\omega$ of all finite monoids with zero, the set $\mathbb{N}$ of all natural numbers including zero with the usual multiplication is a monoid whose residual with respect to $\mathbf{Z}_\omega$ is the diagonal relation. However, $\mathbb{N}$ is not finite.

### Formation of congruences

**Definition 4.12.** A *formation of congruences* is a function $\mathbb{F}$ that assigns to a set $A$, a set of congruences on $A^*$ satisfying the following conditions.

(i) for each set $A$, the set $\mathbb{F}(A)$ is a filter in $\mathsf{Con}(A^*)$;

(ii) for every two sets $A$ and $B$, and for every congruence $E \in \mathbb{F}(B)$ with quotient morphism $\eta\colon B^* \to B^*/E$, if there exists a monoid homomorphism $\varphi\colon A^* \to B^*$ such that the composition $\eta \circ \varphi\colon A^* \to B^*/E$ is a surjective monoid homomorphism, then $\mathsf{ker}(\eta \circ \varphi)$ is a congruence in $\mathbb{F}(A)$. It can be depicted as follows:

$$\begin{tikzcd}
A^* \arrow[r, two heads] \arrow[d, "\varphi"'] \arrow[dr, two heads] & A^*/\mathsf{ker}(\eta \circ \varphi) \arrow[d, dashed, two heads] \\
B^* \arrow[r, two heads, "\eta"'] & B^*/E
\end{tikzcd}$$

### Formations of languages

**Definition 4.13.** A *formation of languages* is a function $\mathcal{F}$ that assigns to every alphabet $A$ a set of formal languages satisfying the following conditions.

(i) for each alphabet $A$, if $L$ is a language in $\mathcal{F}(A)$, then $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is included in $\mathcal{F}(A)$;

(ii) for each alphabet $A$, if both $\mathsf{coEq}(A^*/C_1)$, $\mathsf{coEq}(A^*/C_2)$ are included in $\mathcal{F}(A)$, then so is $\mathsf{coEq}(A^*/C_1 \cap C_2)$;

(iii) for every two alphabets $A$ and $B$, if $L$ is a language in $\mathcal{F}(B)$ and $\eta\colon B^* \to \mathsf{free}(\langle L\rangle)$ denotes the quotient morphism, then for each monoid morphism $\varphi\colon A^* \to B^*$ such that $\eta \circ \varphi$ is surjective, the set $\mathsf{coEq}(A^*/\mathsf{ker}(\eta \circ \varphi))$ belongs to $\mathcal{F}(A)$.

The above definition was completely given in terms of equations and coequations and from the very beginning it clearly underscores the relation between languages and congruences. This will have an impact on the later appearing results as it simplifies most of the steps in the proofs. The unique difference between this notion and Definition 4.4 is that in third item, the composition $\eta \circ \varphi$ needs to be surjective. Moreover, in this new notion we do not require the languages to be regular.

### 4.3.2 Eilenberg's theorem for formations of monoids

We are now in position to show three different Eilenberg relations for formations. We first show that formations of monoids are in one-to-one correspondence with formations of congruences. After this result, we show that formations of congruences are in one-to-one correspondece with formations of languages. Consequently, formations for monoids and languages have also this correspondence.

#### Monoids vs congruences

**Proposition 4.14.** Every formation of monoids $\mathbf{F}$ determines, in a canonical way, a formation of congruences $\mathbb{F}$.

*Proof.* Consider the assignment:

$$\mathbb{F}\colon A \longmapsto \{C \in \mathsf{Con}(A^*) \mid A^*/C \in \mathbf{F}\}.$$

Let $C_1$ and $C_2$ be congruences in $\mathbb{F}(A)$, then $A^*/C_1$ and $A^*/C_2$ are monoids in $\mathbf{F}$. Note that $C_1 \cap C_2$ is included in $C_i$ for $i = 1, 2$. If we consider the corresponding quotient homomorphisms $\pi_i\colon A^*/C_1 \cap C_2 \to A^*/C_i$ for $i = 1, 2$, we have that $\{\pi_1, \pi_2\}$ is a family of surjective morphisms separating the elements of $A^*/C_1 \cap C_2$. It follows from Proposition 4.8 that $A^*/C_1 \cap C_2$ is a subdirect product of two monoids in $\mathbf{F}$. Therefore $C_1 \cap C_2$ is a congruence in $\mathbb{F}(A)$. Now, if $C$ is a congruence in $\mathbb{F}(A)$ and $D$ is a congruence on $A^*$ with $C \subseteq D$, we have that $A^*/D$ is a quotient of $A^*/C$. It follows that $A^*/D$ is a monoid in $\mathbf{F}$, and we conclude that $D$ is a congruence in $\mathbb{F}(A)$. Therefore, $\mathbb{F}(A)$ is a filter in $\mathsf{Con}(A^*)$.

Let $A$ and $B$ be two sets, and let $E$ be a congruence in $\mathbb{F}(B)$ with quotient morphism $\eta\colon B^* \to B^*/E$. Let $\varphi\colon A^* \to B^*$ be a monoid homomorphism such that the composition $\eta \circ \varphi\colon A^* \to B^*/E$ is a surjective monoid homomorphism. Hence, $A^*/\mathsf{ker}(\eta \circ \varphi)$ is isomorphic to $B^*/E$, which is a monoid in $\mathbf{F}$. It follows that $A^*/\mathsf{ker}(\eta \circ \varphi)$ is in $\mathbf{F}$ and $\mathsf{ker}(\eta \circ \varphi)$ is a congruence in $\mathbb{F}(A)$. □

**Proposition 4.15.** Every formation of congruences $\mathbb{F}$ determines, in a canonical way, a formation of monoids $\mathbf{F}$.

*Proof.* We take $\mathbf{F}$ to be the class of all monoids $M$ for which there exists a set $A$ and a congruence $C \in \mathbb{F}(A)$ satisfying $M \cong A^*/C$. We claim that this set is a formation of monoids.

Let $f\colon M \to N$ be the surjective monoid homomorphism defined on a monoid $M$ in $\mathbf{F}$. Then there exists a set $A$ and a congruence $C \in \mathbb{F}(A)$ satisfying $M \cong A^*/C$. Let $\gamma\colon A^* \to M$ be a monoid homomorphism with kernel $C$. Then $f \circ \gamma\colon A^*/C \to N$ is a surjective monoid homomorphism. Moreover, $C \subseteq \mathsf{ker}(f \circ \gamma)$, which implies that $\mathsf{ker}(f \circ \gamma)$ is a congruence in $\mathbb{F}(A)$. Finally, $A^*/\mathsf{ker}(f \circ \gamma)$ is isomorphic to $N$, and so $N$ belongs to $\mathbf{F}$.

Now, let $M$ be a monoid that can be expressed as the subdirect product of a finite family $(M_i)_{i\in n}$ of monoids in $\mathbf{F}$. Therefore, for each index $i \in n$ there exists a set $A_i$ and a congruence $C_i \in \mathsf{Con}(A_i)$ satisfying $M_i \cong A_i^*/C_i$. Let us denote the corresponding quotient homomorphisms as $\eta_i \colon A_i^* \to A_i^*/C_i$. Consider the set $B = \bigcup_{i\in n} A_i$. By the universal property of the free monoid, we can construct a monoid epimorphism $\varphi_i \colon B^* \to A_i^*$ for all $i \in n$. Thus, $\eta_i \circ \varphi_i \colon B^* \to A_i^*/C_i$ is a surjective monoid homomorphism for all $i \in n$. As $\mathbb{F}$ is a formation of congruences, the congruence $D_i = \mathsf{ker}(\eta_i \circ \varphi_i)$ belongs to $\mathbb{F}(B)$ for all $i \in n$. Note that $M$ can be expressed as the subdirect product of the finite family of monoids $\{B^*/D_i \mid i \in n\}$. Since $B$ generates each monoid in the family, $M$ is generated by $B$. It follows that $M \cong B^*/F$ for some congruence $F$ on $B^*$. Since $M$ is a subdirect product of the monoids $B^*/D_i$, we have that $\bigcap_{i\in n} D_i \subseteq F$. Note that $\bigcap_{i\in n} D_i$ is a congruence in $\mathbb{F}(B)$ as it is a finite intersection of congruences in $\mathbb{F}(B)$. Finally, $F$ is a congruence in $\mathbb{F}(B)$ and $M$ belongs to $\mathbf{F}$. □

**Theorem 4.16.** *The mappings $\mathbb{F} \mapsto \mathbf{F}$ and $\mathbf{F} \mapsto \mathbb{F}$ define mutually inverse correspondences between formations of congruences and formations of monoids.*

*Proof.* Consider a formation of monoids $\mathbf{F}$. The first correspondence gives us the formation of congruences $\mathbb{F}$ that assigns to each set $A$ the set $\{C \in \mathsf{Con}(A^*) \mid A^*/C \in \mathbf{F}\}$ of all congruences whose quotient belongs to $\mathbf{F}$. Let $\mathbf{H}$ be the class of all monoids $M$ for which there exists a set $A$ and a congruence $C \in \mathbb{F}(A)$ satisfying $M \cong A^*/C$. It immediately follows that $\mathbf{H}$ is included in $\mathbf{F}$. The other inclusion follows easily since every monoid can be written as a quotient of a free monoid.

Now, let $\mathbb{F}$ be a formation of congruences. The first correspondence gives us $\mathbf{F}$, which is equal to the class of all monoids $M$ for which there exists a set $A$ and a congruence $C \in \mathbb{F}(A)$ satisfying $M \cong A^*/C$. Let $\mathbb{H}$ denote the formation of congruence quotients that assigns to each set $A$ the set $\{C \in \mathsf{Con}(A^*) \mid A^*/C \in \mathbf{F}\}$. For a fixed set $A$, if $C$ is a congruence in $\mathbb{F}(A)$, then $A^*/C$ is a monoid in $\mathbf{F}$ and $C$ belongs to $\mathbb{H}(A)$. Let $C$ be a congruence in $\mathbb{H}(A)$, then $A^*/C$ is a monoid in $\mathbf{F}$. Therefore, there exists a set $B$ and a congruence $D \in \mathbb{F}(B)$ satisfying $A^*/C \cong B^*/D$. Let $\eta \colon B^* \to B^*/D$ and $\delta \colon A^* \to A^*/C$ be the corresponding quotient homomorphisms. Let $\rho \colon A^*/C \to B^*/D$ be the corresponding monoid isomorphism. It follows that $\gamma = \rho \circ \delta$ is a monoid epimorphism from $A^*$ onto $B^*/D$. By Proposition 2.11, there exists a monoid homomorphism $\varphi \colon A^* \to B^*$ with $\eta \circ \varphi = \gamma$. Summarising,

$$\begin{tikzcd}
A^* \arrow[r, two heads, "\delta"] \arrow[d, dashed, "\varphi"'] \arrow[dr, two heads, "\gamma"] & A^*/C \arrow[d, two heads, "\rho"] \\
B^* \arrow[r, two heads, "\eta"'] & B^*/D
\end{tikzcd}$$

As $\mathbb{F}$ is a formation of congruences, $\mathsf{ker}(\eta \circ \varphi)$ belongs to $\mathbb{F}(A)$. Finally, $C$ is in $\mathbb{F}(A)$ as $\mathsf{ker}(\eta \circ \varphi) = \mathsf{ker}(\gamma) = \mathsf{ker}(\rho \circ \delta) = C$. □

### Congruences vs languages

**Proposition 4.17.** Every formation of congruences $\mathbb{F}$ determines, in a canonical way, a formation of languages $\mathcal{F}$.

*Proof.* Consider the assignment:

$$\mathcal{F}\colon A \longmapsto \bigcup\{\mathsf{coEq}(A^*/C) \mid C \in \mathbb{F}(A)\}.$$

Let $L$ be a language in $\mathcal{F}(A)$, then there exists a congruence $C$ in $\mathbb{F}(A)$ for which $L$ is a language in $\mathsf{coEq}(A^*/C)$. By Proposition 3.25, we have that $C \subseteq \mathsf{Eq}\langle L\rangle$. Thus, $\mathsf{Eq}\langle L\rangle$ is a congruence in $\mathbb{F}(A)$. Hence, $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is included in $\mathcal{F}(A)$. Now, if $\mathsf{coEq}(A^*/C_1)$ and $\mathsf{coEq}(A^*/C_2)$ are included in $\mathcal{F}(A)$, then the congruences $C_1$, $C_2$ are in $\mathbb{F}(A)$. By assumption, the congruence $C_1 \cap C_2$ also belongs to $\mathbb{F}(A)$. Thus, $\mathsf{coEq}(A^*/C_1 \cap C_2)$ is in $\mathcal{F}(A)$. Let $L$ be a language of $\mathcal{F}(B)$ with quotient morphism $\eta\colon B^* \to \mathsf{free}(\langle L\rangle)$. Let $\varphi\colon A^* \to B^*$ such that $\eta\circ\varphi$ is surjective, then $\mathsf{ker}(\eta\circ\varphi)$ is a congruence in $\mathbb{F}(A)$. Thus, $\mathsf{coEq}(A^*/\mathsf{ker}(\eta\circ\varphi))$ belongs to $\mathcal{F}(A)$. Hence, $\mathcal{F}$ is a formation of languages. □

**Proposition 4.18.** Every formation of languages $\mathcal{F}$ determines, in a canonical way, a formation of congruences $\mathbb{F}$.

*Proof.* Consider the assignment:

$$\mathbb{F}\colon A \longmapsto \{C \in \mathsf{Con}(A^*) \mid \mathsf{coEq}(A^*/C) \subseteq \mathcal{F}(A)\}.$$

Let $C$ be a congruence in $\mathbb{F}(A)$. If $D$ is a congruence on $A^*$ with $C \subseteq D$, then, by Theorem 3.24, $\mathsf{coEq}(A^*/D)$ is included in $\mathsf{coEq}(A^*/C)$, which is included in $\mathcal{F}(A)$ by assumption. Now, let $C_1$ and $C_2$ be two congruences in $\mathbb{F}(A)$, then $\mathsf{coEq}(A^*/C_1)$ and $\mathsf{coEq}(A^*/C_2)$ belong to $\mathcal{F}(A)$. As $\mathcal{F}$ is a formation of languages, then $\mathsf{coEq}(A^*/C_1 \cap C_2)$ is included in $\mathcal{F}(A)$. Hence, $C_1 \cap C_2$ is a congruence in $\mathbb{F}(A)$. Hence, $\mathbb{F}$ maps each alphabet $A$ to a filter in $\mathsf{Con}(A^*)$.

Let $A$ and $B$ be two sets and let $C$ be a congruence in $\mathbb{F}(B)$. Consider the corresponding quotient homomorphism $\eta\colon B^* \to B^*/C$. Let $\varphi\colon A^* \to B^*$ be a monoid homomorphism such that the composition $\eta\circ\varphi\colon A^* \to B^*/C$ is a surjective monoid homomorphism. Since $\mathcal{F}$ is a formation of languages, $\mathsf{coEq}(A^*/\mathsf{ker}(\eta\circ\varphi))$ is included in $\mathcal{F}(A)$. It follows that $\mathsf{ker}(\eta\circ\varphi)$ is a congruence in $\mathbb{F}(A)$. □

**Theorem 4.19.** *The mappings $\mathbb{F} \mapsto \mathcal{F}$ and $\mathcal{F} \mapsto \mathbb{F}$ define mutually inverse correspondences between formations of congruences and formations of languages.*

*Proof.* It immediately follows from the assignments we have chosen. □

### Languages vs monoids

**Proposition 4.20.** Every formation of languages $\mathcal{F}$ determines, in a canonical way, a formation of monoids $\mathbb{F}$.

*Proof.* Just consider the composition of the correspondences given by Propositions 4.18 and 4.15. Hence, we take $\mathbf{F}$ to be the class of all monoids $M$ that are isomorphic to $A^*/C$ for some congruence $C$ on $A^*$ satisfying that $\mathsf{coEq}(A^*/C) \subseteq \mathcal{F}(A)$. □

**Proposition 4.21.** Every formation of monoids $\mathbf{F}$ determines, in a canonical way, a formation of languages $\mathcal{F}$.

*Proof.* Just consider the composition of the correspondences given by Propositions 4.14 and 4.17. Hence, for a set $A$ we take the set of languages

$$\mathcal{F}(A) = \bigcup\{\mathsf{coEq}(A^*/C) \mid A^*/C \in \mathbf{F}\}.$$

□

**Theorem 4.22.** *The assignments* $\mathbf{F} \mapsto \mathcal{F}$ *and* $\mathcal{F} \mapsto \mathbf{F}$ *define mutually inverse correspondences between formations of monoids and formations of languages.*

*Proof.* It immediately follows from Theorems 4.16 and 4.19. □

### 4.3.3 An application to relatively disjunctive languages

In this subsection we present a direct application of our last results. The following results can be found in [81]. All the notation appearing there has been translated to our notation. We denote the *cardinality* of a language $L$ over $A$ by $|L|$. We call a language $L$ over $A$ *disjunctive* if $\mathsf{Eq}\langle L\rangle$ is the diagonal relation on $A^*$. We call a language $L$ over $A$ *dense* if $A^*wA^* \cap L \neq \emptyset$ for every $w \in A^*$; otherwise, the language $L$ is said to be *thin.* According to Reis and Shyr, a language $L$ is dense if and only if $L$ contains a disjunctive language [58]. We shall call a language over $A$ *relatively f-disjunctive* [*relatively disjunctive*] (*rf-disjunctive* [r-disjunctive] for short) if there exists a dense language $D$ over $A$ such that for all $w \in A^*$

$$\left|[w]_{\mathsf{Eq}\langle L\rangle} \cap D\right| < \aleph_0, \qquad \left[\left|[w]_{\mathsf{Eq}\langle L\rangle} \cap D\right| \leq 1\right].$$

It has been shown in [41] that $L$ is rf-disjunctive if and only if $L$ is r-disjunctive, if and only if either $A^*$ has no dense $\mathsf{Eq}\langle L\rangle$-classes or has infinitely many dense $\mathsf{Eq}\langle L\rangle$-classes. The next result can be found in [49]. It relates r-disjunctive languages and r-regular monoids.

**Theorem 4.23.** *A language $L$ over $A$ is r-disjunctive if and only if* $\mathsf{free}(\langle L\rangle)$ *is not r-regular.*

We shall denote by $\mathcal{F}(A)$ the set of all non-r-disjunctive languages over $A$. As a consequence of our previous section, we have a result on disjunctive languages that does not follow immediately from the definition.

**Corollary 4.24.** *The assignment* $\mathcal{F}\colon A \to \mathcal{F}(A)$ *is a formation of languages.*

*Proof.* It follows from Theorem 4.22 and the contrapositive version of Theorem 4.23. □

### 4.3.4 Varieties of monoids

We present in this subsection some algebraic definitions in order to show a variant of Eilenberg's variety theorem [32]. Here, varieties of finite monoids are replaced by varieties of monoids (as stated by Birkhoff [19]) and varieties of regular languages are replaced by varieties of languages. The definition of variety of languages is given in terms of equations and coequations as we did in Definitions 4.4 and 4.13.

**Definition 4.25.** A *variety of monoids* is a set of monoids $\mathbf{V}$ satisfying the following conditions.

(i) every homomorphic image of a monoid of $\mathbf{V}$ belongs to $\mathbf{V}$;

(ii) every submonoid of a monoid of $\mathbf{V}$ belongs to $\mathbf{V}$;

(iii) the direct product of every family of monoids of $\mathbf{V}$ also belongs to $\mathbf{V}$.

There are two points in which this definition differs from that of pseudovariety (Definition 4.1). One is that all monoids in $\mathbf{V}$ are not assumed to be finite. The second one is that $\mathbf{V}$ is closed under arbitrary direct products. Birkhoff proved two main results; the characterisation of varieties by sets of identities and the closure conditions a class of algebras must satisfy in order to be a variety. As a consequence, varieties of monoids are equationally defined classes of monoids [52, 19]. The following theorem of Kogalovskiĭ [47] (see also [52, 39]) characterises varieties of monoids in terms of quotients and subdirect products.

**Theorem 4.26.** *A class of monoids* $\mathbf{V}$ *is a variety if and only if it is closed under taking arbitrary subdirect products and quotients.*

Consequently, the main difference between varieties of monoids and formations of monoids is that in a variety, arbitrary subdirect products are allowed. In fact, Kogalovskiĭ proves that from the closure under under quotients and arbitrary subdirect products we retrieve closure under submonoids. To mimic this property and bearing in mind an Eilenberg result for varieties, we present the following definition.

**Definition 4.27.** A *variety of languages* is a function $\mathcal{V}$ that assigns to every alphabet $A$ a set of formal languages satisfying the following conditions.

(i) for each alphabet $A$, if $L$ is a language in $\mathcal{V}(A)$, then $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is included in $\mathcal{V}(A)$;

(ii) for each alphabet $A$, if the set $\{\mathsf{coEq}(A^*/C_i) \mid i \in I\}$ is included in $\mathcal{V}(A)$, then so is $\mathsf{coEq}(A^*/\bigcap_{i\in I} C_i)$;

(iii) for every two alphabets $A$ and $B$, if $L$ is a language in $\mathcal{F}(B)$ and $\eta\colon B^* \to \mathsf{free}(\langle L\rangle)$ denotes the quotient morphism, then for each monoid morphism $\varphi\colon A^* \to B^*$ such that $\eta \circ \varphi$ is surjective, the set $\mathsf{coEq}(A^*/\mathsf{ker}(\eta\circ\varphi))$ belongs to $\mathcal{F}(A)$.

Here, we require closure under arbitrary intersection of congruences to mirror the respective closure under arbitrary products in the definition of variety of monoid. We can show that varieties of languages are in one-to-one correspondence with varieties of monoids. This proof was included in the paper [14]. We decided not to include it here because it is just a slight modification of the correspondences above. Consequently, we adopted the name "variety of languages" to emphasize this property. As expected, different classes of monoids are related with different assignments of languages. This definition differs form Definition 4.4 as we do not require the languages to be regular. The second item requires also conditions on arbitrary families of congruences. This is also a shared difference with Definition 4.13.

These differences has also a counterpart in the congruence side. Among other particularities for varieties, the residual of a monoid is always a congruence whose quotient is a monoid in the variety.

**Proposition 4.28.** If $\mathbf{V}$ is a variety of monoids, then for every monoid $M$, the quotient $M/C^M_{\mathbf{V}}$ is a monoid in $\mathbf{V}$.

*Proof.* Note that $M/C^M_{\mathbf{V}}$ is the subdirect product of the family of all quotients of $M$ in $\mathbf{V}$. Kogalovskiĭ's Theorem 4.26 guarantees us that this subdirect product is in $\mathbf{V}$. □

As a consequence, if $\mathbf{V}$ is a variety of monoids, the assignment of Proposition 4.14 maps each alphabet $A$ to a principal filter. In this case, the principal filter generated by the residual of $A^*$ over $\mathbf{V}$

$$\mathbb{V}\colon A \longmapsto \{C \in \mathsf{Con}(A^*) \mid A^*/C \in \mathbf{V}\} = [C_{\mathbf{V}}^{A^*}).$$

In the case of varieties, Theorem 4.16 gives a correspondence between varieties $\mathbf{V}$ and formations of congruences $\mathbb{V}$ satisfying that for all $A$, the set $\mathbb{V}(A)$ is a principal filter in $\mathsf{Con}(A^*)$.

## 4.4 Discussion and future work

The theorems presented in this chapter follow a long standing tradition of results relating classes of monoids with the languages they can recognize. Scattered results in this direction appeared in the mid-sixties. Schützenberger [71], for example, proved that star-free languages are in one-to-one correspondence with aperiodic monoids. The success of Eilenberg's theorem relies on the generality of the result; he understood that finite aperiodic monoids are just an example of a pseudovariety. We can find further instances of this result: the rational languages, for example, are associated with the variety of all finite monoids [32, 46] and the piecewise testable languages with the variety of finite $\mathscr{J}$-trivial monoids [75]. A great deals of analogous results have been established over the past years. These results strengthen the relation between monoids, automata, and languages.

Several attempts to generalize this result appear in the literature; see for instance [54, 79, 33]. These papers aimed at extending Eilenberg's result by relaxing some conditions on the class of monoids or on the class of languages. A strong attempt to embrace all these results in a common categorical framework was made in [5], where the authors introduced varieties of languages in a category $\mathscr{C}$, and proved their correspondence with pseudovarieties of monoids in a closed monoidal category $\mathscr{D}$, provided that $\mathscr{C}$ and $\mathscr{D}$ are dual on finite objects. In any case, all the results involve classes of finite monoids.

A strong attempt to present a general result aiming at generalising varieties to the more general notion of formation was made in [15], altought it was still made for formations of finite monoids. Since this result clearly relates to the theorems presented here, we will specify the existing connections. Possibly, the biggest difference with this work is the definition of "formation of language" adopted by them, much more in the original spirit of Eilenberg's concept. We reproduce their definition to see that, for regular languages, the following definition and definition 4.13 coincide. In order to avoid confusion, we will rename that concept.

**Definition 4.29.** A *formation of regular languages* is a function $\mathcal{R}$ that assigns to every alphabet $A$ a set of regular languages over $A$ satisfying:

(i)' for each alphabet $A$, $\mathcal{R}(A^*)$ is closed under Boolean operations and derivatives;

(ii)' for every two alphabets $A$ and $B$, if $L$ is a language in $\mathcal{R}(B)$ and $\eta\colon B^* \to \mathsf{free}\langle L\rangle$ denotes the quotient morphism, then for each monoid morphism $\varphi\colon A^* \to B^*$ such that $\eta \circ \varphi$ is surjective, the language $\varphi^{-1}(L)$ belongs to $\mathcal{R}(A)$.

Clearly, this definition relates with Definition 4.2, the main difference being that here the composition $\eta \circ \varphi$ needs to be surjective. We will prove that if $\mathcal{F}$ is a formation of languages (in the sense of Definition 4.13) assigning to each alphabet $A$ a set of regular languages, then this is equivalent to being a formation of regular languages (in the sense of Definition 4.29). To do so, we will need the following Lemma

**Lemma 4.30.** *Let $\mathcal{R}$ be an formation of regular languages. For an alphabet $A$, if $C$ is a congruence on $A^*$ with finite quotient $A^*/C$, then*

$$\mathsf{coEq}(A^*/C) \subseteq \mathcal{R}(A) \quad \textit{if and only if} \quad [w]_C \in \mathcal{R}(A) \textit{ for all } w \in A^*.$$

*Proof.* Let $w$ be a word in $A^*$, for the coloration $c_w \colon A^*/C \to 2$, given by $c_w([u]_C) = 1$ if and only if $[u]_C = [w]_C$, we have that $[w]_C = o_{c_w}([\varepsilon])$ is a language in $\mathsf{coEq}(A^*/C)$ which is included in $\mathcal{R}(A)$. On the converse, let $L$ be language in $\mathsf{coEq}(A^*/C)$, then $L = \bigcup\{[w]_C \mid w \in L\}$. As $A^*/C$ is finite, $L$ is a finite union of languages in $\mathcal{R}(A)$. Hence, $L$ belongs to $\mathcal{R}(A)$. □

**Theorem 4.31.** *Let $\mathcal{F}$ be a formation of languages assigning a set of regular languages to each alphabet, then $\mathcal{F}$ is a formation of regular languages. Conversely, if $\mathcal{R}$ is a formation of regular languages, then $\mathcal{R}$ is a formation of languages assigning a set of regular languages to each alphabet.*

*Proof.* Assume $\mathcal{F}$ is a formation of languages (see Definition 4.13) assigning a set of regular languages to each alphabet.

(i)' Let $L$ be a language in $\mathcal{F}(A)$, then $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is included in $\mathcal{F}(A)$. As $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is a preformation of languages containing $L$, then the complement and every derivative of $L$ belong to it. Let $L_1$ and $L_2$ be two languages in in $\mathcal{F}(A)$, then $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L_1\rangle)$ and $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L_2\rangle)$ are included in $\mathcal{F}(A)$. It follows that

$$D = \mathsf{coEq}(A^*/[\mathsf{Eq}\langle L_1\rangle \cap \mathsf{Eq}\langle L_2\rangle])$$

is also included in $\mathcal{F}(A)$. By Proposition 3.25, we have that $L_1$ and $L_2$ are languages in $D$, which is a preformation of languages, then $L_1 \cap L_2$ and $L_1 \cup L_2$ are languages in $D$.

(ii)' Now, consider two alphabets $A$ and $B$, and let $L$ be a language in $\mathcal{F}(B)$. Let $\eta$ denote the quotient morphism $\eta\colon B^* \to \mathsf{free}\langle L\rangle$ and let $\varphi\colon A^* \to B^*$ be a monoid morphism such that $\eta \circ \varphi$ is surjective. Then $\mathsf{coEq}(A^*/\mathsf{ker}(\eta\circ\varphi))$ is included in $\mathcal{F}(A)$. Let $L'$ be a language in $\langle\varphi^{-1}(L)\rangle$, then there exists some word $u \in A^*$ with $L' = [\varphi^{-1}(L)]_u$. Let $(v, w)$ be a pair in $\mathsf{ker}(\eta\circ\varphi)$, then

$$\begin{aligned} L'_v &= [\varphi^{-1}(L)]_{uv} = \{x \in A^* \mid uvx \in \varphi^{-1}(L)\} = \{x \in A^* \mid \varphi(uvx) \in L\} \\ &= \{x \in A^* \mid \varphi(u)\varphi(v)\varphi(x) \in L\} = \{x \in A^* \mid \varphi(x) \in L_{\varphi(u)\varphi(v)}\}. \end{aligned}$$

Recall that $L_{\varphi(u)}$ is a language in $\langle L\rangle$, and $(\varphi(v), \varphi(w))$ is a pair in $\mathsf{Eq}\langle L\rangle$, therefore:

$$L'_v = \{x \in A^* \mid \varphi(x) \in L_{\varphi(u)\varphi(v)}\} = \{x \in A^* \mid \varphi(x) \in L_{\varphi(u)\varphi(w)}\} = L'_w.$$

It follows that $\mathsf{ker}(\eta\circ\varphi) \subseteq \mathsf{Eq}\langle\varphi^{-1}(L)\rangle$. Again by Proposition 3.25, we have $\varphi^{-1}(L)$ is a language in $\mathsf{coEq}(A^*/\mathsf{ker}(\eta\circ\varphi))$.

Consequently, $\mathcal{F}$ is a formation of regular languages.

Now, let $\mathcal{R}$ be a formation of regular languages (see Definition 4.29).

(i) Let $L$ be a language in $\mathcal{R}(A)$. It is well known that a language $L$ over an alphabet $A$ is regular if and only if the set $\{{}_vL_w \mid v, w \in A^*\}$ is finite. Let $[u]$ be an element in $A^*/\mathsf{Eq}\langle L\rangle$, then it holds by 3.36 that

$$[u] = \bigcap\{{}_vL_w \mid u \in {}_vL_w\} \setminus \bigcup\{{}_vL_w \mid u \notin {}_vL_w\}.$$

That is, every atom in $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ belongs to the Boolean algebra generated by the derivatives of $L$. It follows from Lemma 4.30 that $\mathsf{coEq}(A^*/\mathsf{Eq}\langle L\rangle)$ is included in $\mathcal{R}(A)$.

(ii) Now, assume that $\mathsf{coEq}(A^*/C_1)$ and $\mathsf{coEq}(A^*/C_2)$ are both included in $\mathcal{R}(A)$. Let $[w]_{C_1\cap C_2}$ be an atom in $\mathsf{coEq}(A^*/C_1 \cap C_2)$. The equation $[w]_{C_1\cap C_2} = [w]_{C_1} \cap [w]_{C_2}$ trivially holds. Note that $[w]_{C_i}$ is a language in $\mathsf{coEq}(A^*/C_i)$ for $i = 1, 2$, and hence, included in $\mathcal{R}(A)$. We conclude that $[w]_{C_1\cap C_2}$ is a language in $\mathcal{R}(A)$. By Lemma 4.30, $\mathsf{coEq}(A^*/C_1 \cap C_2)$ is included in $\mathcal{R}(A)$.

(i) Finally, consider two alphabets $A$ and $B$, and let $L$ be a language in $\mathcal{R}(B)$. Let $\eta$ denote the quotient morphism $\eta\colon B^* \to \mathsf{free}\langle L\rangle$ and let $\varphi\colon A^* \to B^*$ be a monoid morphism such that $\eta \circ \varphi$ is surjective. We have that $\varphi^{-1}(L)$ is a language in $\mathcal{R}(A)$, hence, by the first item of this proof, we have that $\mathsf{coEq}(A^*/\mathsf{Eq}\langle\varphi^{-1}(L)\rangle)$ is included in $\mathcal{R}(A)$. Let us check $\mathsf{ker}(\eta \circ \varphi) = \mathsf{Eq}\langle\varphi^{-1}(L)\rangle$. We will only check that $\mathsf{Eq}\langle\varphi^{-1}(L)\rangle$ is included in $\mathsf{ker}(\eta\circ\varphi)$; for the other inclusion, see the proof done in the first part of this Theorem. Let $(v, w)$ be a pair in $\mathsf{Eq}\langle\varphi^{-1}(L)\rangle$, we claim that $(\varphi(v), \varphi(w))$ is a pair in $\mathsf{Eq}\langle L\rangle$. As $\eta \circ \varphi$ is surjective, for every word $u \in B^*$, there exists some word $u' \in A^*$, with $(u, \varphi(u')) \in \mathsf{Eq}\langle L\rangle$. Let $L_u$ be a language in $\langle L\rangle$.

$$\begin{aligned}
L_{u\varphi(v)} &= L_{\varphi(u)\varphi(v)} && (\eta \circ \varphi \text{ surjective})\\
&= L_{\varphi(uv)} && (\text{monoid homomorphism})\\
&= \{x \in B^* \mid \varphi(u'v)x \in L\}\\
&= \{x \in B^* \mid \varepsilon \in L_{\varphi(u'v)x}\}\\
&= \{x \in B^* \mid \varepsilon \in L_{\varphi(u'v)\varphi(x')}\} && (\eta \circ \varphi \text{ surjective})\\
&= \{x \in B^* \mid \varphi(u'vx') \in L\} && (\text{monoid homomorphism})\\
&= \{x \in B^* \mid u'vx' \in \varphi^{-1}(L)\}\\
&= \{x \in B^* \mid u'wx' \in \varphi^{-1}(L)\} && ((v, w) \in \mathsf{Eq}\langle\varphi^{-1}(L)\rangle)\\
&= \cdots\\
&= L_{u\varphi(w)},
\end{aligned}$$

thus, $\mathsf{ker}(\eta \circ \varphi) = \mathsf{Eq}\langle\varphi^{-1}(L)\rangle$.

Finally, $\mathcal{R}$ is a formation of languages. □

Therefore, our work subsumes the Eilenberg result for formations of finite monoids presented in [15] as we do not require the monoids to be finite nor the languages to be regular.

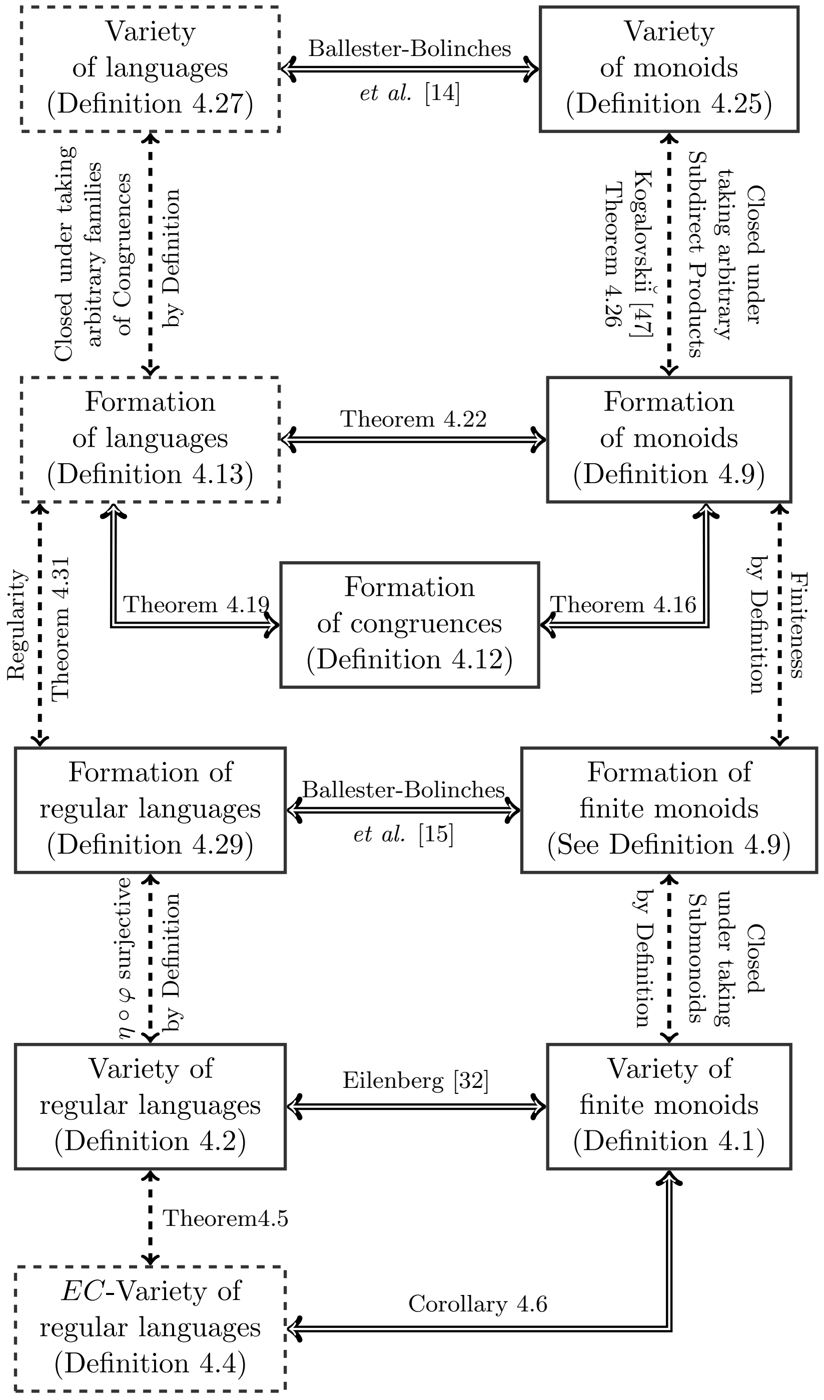
To better summarise all the notions and relations presented in this Chapter we will introduce a schematic representation of all the results. In this diagram, notions are presented inside boxes. A box drawn with dashed lines indicates that the notion inside the box is defined using equations and coequations. Two concepts are bound with an arrow if there exists a relation between these concepts. A double arrow labelled as follows

$$\begin{tikzcd}\boxed{A} \arrow[r, Leftrightarrow, "X"] & \boxed{B}\end{tikzcd}$$

indicates the existence of an Eilenberg's theorem between concepts $A$ and $B$ that is proven in $X$. A dashed arrow labelled as follows

$$\begin{tikzcd}\boxed{A} \arrow[r, leftrightarrow, dashed, "K", "X"'] & \boxed{B}\end{tikzcd}$$

indicates that concepts $A$ and $B$ under conditions $K$ are equivalent notions by a result proven in $X$. The notions related with languages appear on the left hand side of the diagram. The notions involving congruences are situated in the middle column of the diagram and, finally, all the notions related with monoids appear on the right hand side of the diagram. There is also a conceptual separation in the files of this diagram; concepts on the lower half of the diagram involve regularity on the languages and finiteness on the monoids, whereas the upper half does not assume this restrictions.



A first look at the diagram shows that every class of languages considered in all the Eilenberg results presented in this dissertation is defined in terms of equations and coequations or it is equivalent, under some conditions, to a class of languages defined in terms of equations and coequations. This underscores the importance of these constructions in the study of Eilenberg's related results. The differences appearing in the different classes of monoids have a counterpart in the classes of languages that are better explained in terms of congruences.

Regarding this issue, less is known about congruences. We made a significant step for the recognition of the role that monoids have in this kind of results. A description of the lattice structure of these families of congruences must clearly be a priority.

For example, if $M$ is a monoid generated with one element, we know that $M$ is isomorphic to a quotient $\mathbb{N}/\theta_{n,r}$ on the free monoid $\mathbb{N}$ on one letter, with $n$ and $r$ natural numbers, $r \geq 1$ and the congruence $\theta_{n,r}$ defined as

$$\theta_{n,r} = \Delta_{\mathbb{N}} \cup \{(p,q) \in \mathbb{N} \times \mathbb{N} \mid p, q \geq n \text{ and } p \equiv q \mod r\},$$

where $(n, r)$ denotes the type of the monoid $M$. For this kind of congruences, since they have a closed form, we can determine many relations in terms of the parameters. Note that for any possible parameter considered $n, r$ the quotient $\mathbb{N}/\theta_{n,r}$ represents a finite monoid. Degenerated cases include $(n \geq 0,\ r = 1)$ or $(n = 0, r \geq 1)$ . Those particular cases are finite monoids with zero and finite cyclic groups, respectively.

Given $n, m, r, s \in \mathbb{N}$ with $r, s \geq 1$. It holds:

$$\theta_{m,s} \subseteq \theta_{n,r} \quad \Leftrightarrow \quad n \leq m \text{ and } r \mid s.$$

From the last description we can derive the following equations:

$$\theta_{m,s} \cap \theta_{n,r} = \theta_{\max(m,n),\mathrm{lcm}(r,s)}, \qquad \theta_{m,s} \vee \theta_{n,r} = \theta_{\min(m,n),\mathrm{gcd}(r,s)},$$

where $\max(m, n)$ stands for the *maximum* between $m$ and $n$ and $\mathrm{lcm}(r, s)$ stands for the *least common multiple* between $r$ and $s$. On the right hand side, $\min(m, n)$ stands for the *minimum* between $m$ and $n$ and $\gcd(r, s)$ stands for the *greatest common divisor* between $r$ and $s$.

We don't have references for the congruences on arbitrary free monoids with more than one generator whose quotient is finite. That's why we cannot expect such closed results on arbitrary formations of monoids. However, we should investigate how some conditions on the classes of monoids have a counterpart in the corresponding lattices of congruences. All in all, this research line deserves further attention.

Gran!
Eres tan gran
que les absències et dignifiquen,
et fan més gran.

Gran — Senior i el Cor Brutal

# CHAPTER 5

# A description based on languages of the final non-deterministic automaton

## 5.1 Introduction

The study of the behaviour of non-deterministic automata has traditionally focused on the languages which can be associated to the different states. We can assign to every state of an automaton an associated language, consisting of all words which send this state to a final or terminal state. Traditionally, many authors have considered as the behaviour of a state of an automaton simply its associated language. Contrarily to what happens with deterministic automata, it is well-known that is not possible to describe the behaviour of non-deterministic automata from one state by considering just the language associated to that state. This is a consequence of ignoring the different decisions that may be taken from each state. However, when we take into account the different branches or decisions that may be taken at every state, the behaviour problem becomes manageable.

From this point of view, automata are regarded as labelled transition systems or coalgebras for suitable endofunctors on the category Set. In this scope, the idea of the behaviour of the states of the coalgebra is related to the notion of bisimilarity. We can say that two states have the same behaviour when they are bisimilar. Under very general hypotheses, which hold for automata, when a category of coalgebras possesses a final object, two states are bisimilar if and only if both states have the same image by the unique homomorphism into the final object. This motivates the interest in studying the final objects in some categories of coalgebras, like automata.

In this chapter we present a description of a final object for the category of non-deterministic automata with the help of some structures defined in terms of languages that take into account the different decisions which can be made at every state, that is, the branches that can be taken. Our construction emphasises the role of languages as natural objects to describe the behaviour of automata. In this case, the behaviour of the automata can be described with the help of the concept of bisimilarity.

Up to now, most known descriptions of final coalgebras are of a very general theoretical nature or are given as a quotient of a coalgebra by the bisimilarity relation (see [16, 25, 73, 74]). When they are applied to the functor $2 \times R$ associated to non-deterministic automata, it seems that they do not give a clear idea of the role of languages, which are incontestably a central notion in this theory, in the final automaton. Hence the question of whether languages can be used to describe the behaviour of non-deterministic automata as labelled transition systems remains open. The aim of this chapter is to give a positive answer to this question. This also allows us to characterise bisimilarity of states of automata in terms of languages, which has been a long-standing unsolved problem in this theory. As a consequence, we obtain a characterisation of bisimilarity of states of automata in terms of languages and a method to minimise non-deterministic automata with respect to bisimilarity of states.

We have done our best to keep the chapter self-contained. Accordingly, Section 5.2 introduces known results on bisimulations and final coalgebras. Our main result is presented in Theorem 5.16. We conclude the chapter by justifying why our description is the most natural one and by establishing some questions for future research.

## 5.2 Final non-deterministic automata

It seems desirable to find, like in the case of deterministic automata, a description which emphasises the role of languages as natural objects to describe bisimilarity of states in non-deterministic automata. This is the aim of this Section. Intuitively, we can say that two states of two automata are bismilar when they are not distinguishable from the observer point of view, in other words, when the "observable behaviours" of both automata from both states are the same. The following result summarises some of the properties of bisimulations in automata. This is a particular case of some general results about coalgebras presented in [63].

**Theorem 5.1.** *Let $(X, \alpha)$, $(Y, \beta)$, and $(Z, \gamma)$ be three automata over the same alphabet $A$.*

1. *The union of a family of bisimulations between $X$ and $Y$ is again a bisimulation;*
2. *The relational composition of two bisimulations between $X$ and $Y$, and $Y$ and $Z$, repectively, is a bisimulation between $X$ and $Z$;*
3. *The diagonal relation $\Delta_X = \{(x, x) \mid x \in X\}$ is a bisimulation on $X$;*
4. *The relational inverse of a bismulation between $X$ and $Y$ is a bismulation between $Y$ and $X$.*

As a consequence, there exists a largest bisimulation between two automata over the same alphabet, namely the union of all bisimulations between them.

**Theorem 5.2** ([63, Theorem 2.5])**.** *Let $(X, c, \alpha)$ and $(Y, d, \beta)$ be two non-deterministic coloured automata over the same alphabet. A map $f\colon X \to Y$ is a homomorphism between non-deterministic automata if and only if its graph $\mathsf{G}(f) = \{(x, f(x)) \mid x \in X\}$ is a bisimulation between $X$ and $Y$.*

**Theorem 5.3** ([63, Theorem 2.5])**.** *Let $(X, c, \alpha)$ and $(Y, d, \beta)$ be two non-deterministic coloured automata over the same alphabet. A map $f\colon X \to Y$ is a homomorphism between non-deterministic automata if and only if its graph $\mathsf{G}(f) = \{(x, f(x)) \mid x \in X\}$ is a bisimulation between $X$ and $Y$.*

**Theorem 5.4** ([63, Theorem 9.2])**.** *Every bisimulation of the final non-deterministic coloured automaton $(T, e, \tau)$ is contained in the diagonal $\Delta_T = \{(t,t) \mid t \in T\}$. In other words, two bisimilar states are equal.*

An automaton satisfying the above condition (two bisimilar states are equal) is called *simple*. A way to check bisimilarity between two states of two automata is to check whether both states have the same images under the unique homomorphisms into the final automaton.

**Theorem 5.5.** *Let $(X, c, \alpha)$ and $(Y, d, \beta)$ be two non-deterministic coloured automata and let $(T, e, \tau)$ denote the final non-deterministic coloured automaton with homomorphisms $!$ and $!'$ from $X$ and $Y$, respectively, to $T$. Two states $x \in$ and $y \in Y$ are bisimilar if and only if $!(x) =!'(y)$.*

*Proof.* Suppose that $!(x) =!'(y) = t$, then $(x, t) \in R = \mathsf{G}(!)$ and $(y, t) \in R' = \mathsf{G}(!')$. Hence, $(x, y)$ belongs to the bisimulation $(R')^{-1} \circ R$ by Theorem 5.1 and so $x$ and $y$ are bisimilar. Conversely, suppose that $x$ and $y$ are bisimilar, that is $(x, y)$ belongs to a bisimulation $V$. Denote by $R = \mathsf{G}(!)$ and $R' = \mathsf{G}(!')$ the graphs of $!$ and $!'$ respectively. Hence $(!(x), !'(y))$ belongs to the bisimulation $R' \circ V \circ R$ on $T$ by Theorem 5.1. By Theorem 5.4 $!(x) =!'(y)$, as desired. □

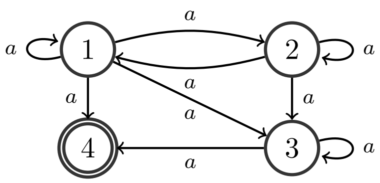
In Section 3.2 we presented the final deterministic automaton $(2^{A^*}, \varepsilon?, \tau)$ of languages over a fixed alphabet $A$. In particular it is shown that two states of a deterministic automaton are bisimilar if and only if the corresponding accepted languages coincide.

Consider now non-deterministic automata. It is easy to see that bisimilar states accept the same language: Suppose that $x$ and $y$ are bisimilar states of the non-deterministic coloured automata $(X, c, \alpha)$ and $(Y, d, \beta)$, respectively, and $w = a_1 a_2 \cdots a_n \in o_c(x)$, the language associated to $x$ in the first automaton. Then there exists a finite sequence of states

$$(x_0, x_1, \cdots, x_n)$$

such that $x_0 = x$, $x_i \in (x_{i-1})_{a_i}$ for $1 \leq i \leq n$, and $c(x_n) = 1$. By bisimilarity, there exists a sequence of states $y_0, y_1, \cdots, y_n$ such that $y_0 = y$ and $y_i \in (y_{i-1})_{a_i}$ such that $x_i$ is bisimilar to $y_i$ for $1 \leq i \leq n$. By definition of bisimulation then either $x_i$ and $y_i$ are both final or none of them is final. Since $x_n$ is final, it follows that $y_n$ is also final and so $w \in o_d(y)$, the language associated to the $y$ in the second automaton. A similar argument shows that $o_d(y) \subseteq o_c(x)$ and so $o_c(x) = o_d(y)$. However, this is not sufficient to identify bisimilar states, as the following example shows.

**Example 5.6.** For a set $X = \{1, 2, 3, 4\}$ with four elements, consider the non-deterministic coloured automaton $(X, c, \alpha)$ over an alphabet $A = \{a\}$ with a single letter, with transitions given by $1_a = X$, $2_a = \{1, 2, 3\}$, $3_a = \{3, 4\}$, and $4_a = \emptyset$, and 4 as the unique final state. This automaton is depicted below



We can see that $o_c(1) = o_c(3) = aa^*$, $o_c(2) = a^2a^*$, and $o_c(4) = \varepsilon$ (as we did in Chapter 3 we identify the regular languages with their corresponding regular expressions). However, 1 and 3 are not bisimilar. To see this, we note that from 1 we can make a transition to 2, with language $a^2a^*$, but from 3 we can only make transitions to 3 and 4, with respective languages $aa^*$ and $\varepsilon$. However, by the previous remark, 2 cannot be bisimilar to neither 3 nor 4.

In order to construct the final non-deterministic automaton we need to introduce the following concepts.

**Definition 5.7.** A *language sequence* over an alphabet $A$ is a finite sequence of the form

$$(L_0, a_1, L_1, a_2, L_2, \cdots, L_{r-1}, a_r, L_r)$$

where $L_i$ are languages in $2^{A^*}$ for $0 \leq i \leq r$, $a_i \in A$ for $1 \leq i \leq r$, and $a_iL_i \subseteq L_{i-1}$ for $1 \leq i \leq r$. The number $r$ is called the *length* of the language sequence. A sequence formed by a unique language $L_0$ will be called a language sequence of length zero.

**Definition 5.8.** A language sequence $(L_0, a_1, L_1, \cdots, L_{r-1}, a_r, L_r)$ over $A$ is said to be a *prefix* of the language sequence $(K_0, b_1, K_1, \cdots, K_{s-1}, b_s, K_s)$ over the same alphabet $A$ when $r \leq s$ and $L_j = K_j$ for $0 \leq j \leq r$ and $a_j = b_j$ for $1 \leq j \leq r$.

**Definition 5.9.** A *language tree* is a (possible empty) set of language sequences $T$ satisfying the following conditions:

1. Every prefix of a language sequence in $T$ belongs to $T$.

2. Given a language sequence $s = (L_0, a_1, L_1, \cdots, L_{r-1}, a_r, L_r)$ in $T$, the set

   $$N_s = \{z \in T \mid z \text{ is of length } k+1 \text{ and } s \text{ is a prefix of } z\}$$

   is finite and

   $$L_k \setminus \{\varepsilon\} = \bigcup\{a_{k+1}L_{k+1} \mid (L_0, a_1, L_1, \cdots, L_{r-1}, a_r, L_r) \in N_s\} \tag{5.1}$$

   when $N_s = \emptyset$, this union is understood to be $\emptyset$, and so $L_k = \{\varepsilon\}$ or $L_k = \emptyset$.

3. If $T$ is not empty, then there is a unique language sequence in $T$ of length zero. This language is called *initial language* of the language tree.

**Definition 5.10.** A *chain of language trees* over an alphabet $A$ is a finite sequence

$$(T_0, a_0, T_1, a_1, T_2, \ldots, T_{r-1}, a_r, T_r)$$

in which $T_i$ is a non-empty language tree over $A$ for $0 \leq i \leq r$, $a_i \in A$ for $1 \leq i \leq r$ such that $\{(L_0, a_0, L_1, \ldots, L_t) \mid (L_1, \ldots, L_t) \in T_1\} \subseteq T_0$. The *initial language* of a chain of language trees $(T_0, a_0, T_1, \ldots, T_r)$ is the initial language of the first language tree $T_0$. The number $r$ is called the *length* of the chain of language trees. The sequence $T_0$ of a single non-empty language tree over $A$ will be considered a chain of language trees of length zero.

**Definition 5.11.** A chain of language trees $(T_0, a_1, T_1, \ldots, T_{r-1}, a_r, T_r)$ over $A$ is said to be a *prefix* of the chain of language trees

$$(U_0, b_1, U_1, \ldots, U_{s-1}, b_s, U_s)$$

over the same alphabet $A$ when $r \leq s$ and $T_j = U_j$ for $0 \leq j \leq r$ and $a_j = b_j$ for $1 \leq j \leq r$.

Now we are in a position to define the states of the final automaton.

**Definition 5.12.** A *tree of chains of language trees* over an alphabet $A$ is a set of chains of language trees $\mathcal{T}$ satisfying:

1. Every prefix of a chain of language trees in $\mathcal{T}$ is also in $\mathcal{T}$.
2. Given a chain of language trees $U = (T_0, a_1, T_1, \ldots, T_{k-1}, a_k, T_k) \in \mathcal{T}$, the set
$$N_U = \{V \in \mathcal{T} \mid V \text{ is of length } k+1 \text{ and } U \text{ is a prefix of } V\}$$
is finite and
$$T_k = \bigcup \{\mathrm{c}(L_k, a_{k+1}, T_{k+1}) \mid (T_0, a_1, T_1, \ldots, T_k, a_{k+1}, T_{k+1}) \in N_U\}$$
where $L_k$ is the initial language of $T_k$ and
$$\mathrm{c}(L_k, a_{k+1}, T_{k+1}) = \{(L_k, a_{k+1}, M_0, b_0, M_1, \ldots, M_r) \mid (M_0, b_0, M_1, \ldots, M_r) \in T_{k+1}\}.$$
3. There is a unique chain of language trees in $\mathcal{T}$ of length zero. Its initial language is called the *initial language* of $\mathcal{T}$ and denoted by $\mathsf{Init}(\mathcal{T})$.

**Definition 5.13.** The *language tree automaton* over the alphabet $A$ is the non-deterministic coloured automaton

$$(X_{\mathcal{L}}, \varepsilon?, \tau)$$

such that:

1. $X_{\mathcal{L}}$ is the set of all possible trees of chains of language trees over $A$,
2. a tree of chains of language trees $\mathcal{T}$ satisfies $\varepsilon?(\mathcal{T}) = 1$ if and only if the empty word $\varepsilon$ belongs to $\mathsf{Init}(\mathcal{T})$, and
3. given a tree of chains of language trees $\mathcal{T}$ and $a_1 \in A$, $\mathcal{T}_{a_1}$ consists of all trees $\mathcal{U}$ of chains of language trees of the form
$$\mathcal{U} = \{(T_1, a_2, T_2, \ldots, T_{k-1}, a_k, T_k) \mid (T_0, a_1, T_1, a_2, T_2 \ldots, T_{k-1}, a_k, T_k) \in \mathcal{T}\},$$
where all chains of language trees of $\mathcal{T}$ begin with the language tree $T_0$.

Our next goal is to show that the language tree automaton over the alphabet $A$ is a final object for the category of automata over the alphabet $A$. This will require checking that given an automaton $(X, c, \alpha)$, there exists a unique homomorphism between $(X, c, \alpha)$ and $(X_{\mathcal{L}}, \varepsilon?, \tau)$. We begin by introducing this homomorphism.

**Definition 5.14.** Let $(X, c, \alpha)$ be a non-deterministic coloured automaton. Let $x_0 \in X$. A sequence

$$(x_0, a_1, x_1, a_2, x_2, \dots, x_{r-1}, a_r, x_r)$$

with $x_i \in X$, $0 \le i \le r$, $a_i \in A$, $1 \le i \le r$, and $x_i \in (x_{i-1})_{a_i}$ for $1 \le i \le r$ will be called a *state sequence* in $(X, c, \alpha)$.

**Construction 5.15.** Let $(X, c, \alpha)$ be a non-deterministic coloured automaton. For each state $x \in X$, let $L_x = o_c(x)$ denote the language accepted by $X$ when it starts from $x$. For each state sequence

$$(x_0, a_1, x_1, a_2, x_2, \dots, x_{r-1}, a_r, x_r),$$

Consider the language sequence $(L_{x_0}, a_1, L_{x_1}, a_2, L_{x_2}, \dots, L_{x_{r-1}}, a_r, L_{x_r})$. Let $T_{x_0}$ be the set of all possible sequences which can be obtained in this way from all sequences of states starting with $x_0$. Note that $T_{x_0}$ is a language tree because $L_x \backslash \{\varepsilon\} = \bigcup \{aL_y \mid y \in x_a, a \in A\}$ in an automaton for every state $x$. Now for each state sequence $(x_0, a_1, x_1, a_2, x_2, \dots, x_{r-1}, a_r, x_r)$ we consider

$$(T_{x_0}, a_1, T_{x_1}, a_2, T_{x_2}, \dots, T_{x_{r-1}}, a_r, T_{x_r}),$$

which is a chain of language trees. Then the set $\mathcal{Q}_{x_0}$ of all chains of language trees which can be obtained from all possible state sequences starting with $x_0$ is a tree of chains of language trees.

**Theorem 5.16.** *Let $(X, c, \alpha)$ be a non-deterministic coloured automaton. The map $\phi \colon X \longrightarrow X_{\mathcal{L}}$ which assigns to each state $x \in X$ the tree of chains of language trees $\mathcal{Q}_x$ presented in Construction 5.15 induces a homomorphism of automata between $(X, c, \alpha)$ and $(X_{\mathcal{L}}, \varepsilon?, \tau)$.*

*Proof.* It is clear that if $y \in x_a$, then $\mathcal{Q}_y \in (\mathcal{Q}_x)_a$. Conversely, suppose that $\mathcal{U} \in (\mathcal{Q}_x)_a$. Then $\mathcal{U}$ is a tree of chains of language trees that has been obtained by removing the first element and $a$ in all language sequences in the chains of language trees in $\mathcal{Q}_x$ which begin with $(T_{x_0}, a)$. But then we get that $\mathcal{U}$ is one of the trees of chains of language trees $\mathcal{Q}_y$ with $y \in x_a$. Therefore the map $\phi$ respects the transitions. Now assume that $c(x_0) = 1$. Then $\varepsilon \in L_{x_0}$. Moreover $\mathsf{Init}(\mathcal{Q}_{x_0}) = L_{x_0}$ and since $\varepsilon$ is one of the elements of this language, $\varepsilon?(\mathcal{Q}_{x_0}) = 1$. On the other hand, if $\mathcal{Q}_x$ is a final state, then $\varepsilon \in \mathsf{Init}(\mathcal{Q}_x)$. Hence $\varepsilon$ is in the language accepted by $(X, c, \alpha)$ when it starts from $x$ and so $c(x) = 1$. □

**Theorem 5.17.** *Let $\Psi$ be a homomorphism between an automaton $(X, c, \alpha)$ and $(X_{\mathcal{L}}, \varepsilon?, \tau)$. Then $\Psi$ coincides with the homomorphism $\Phi$ of Theorem 5.16. As a consequence, $(X_{\mathcal{L}}, \varepsilon?, \tau)$ is a final object in the category of automata over the alphabet $A$.*

*Proof.* For each state $x \in X$, we will use the shorthand $L_x = o_c(x)$ for the language accepted by $X$ starting from $x$. The proof will consist of checking that for every state $x_0 \in X$, $L_{x_0} = \mathsf{Init}(\mathcal{Q}_0)$, where $\mathcal{Q}_0 = \psi(x_0)$. This will be used later to prove that $\psi$ and $\phi$ coincide. For the reader's convenience, we break the proof into separately stated steps.

1. Let $x_0 \in X$. Then $L_{x_0} \subseteq \mathsf{Init}(\mathcal{Q}_0)$.

   Let $w$ be a word in $L_{x_0}$. If $w = \varepsilon$, then $x_0$ is a final state and so $\psi(x_0)$ is also a final state; in particular, $\varepsilon \in \mathsf{Init}(\mathcal{Q}_0)$ where $\mathcal{Q}_0 = \psi(x_0)$. Suppose that $w = a_1a_2 \dots a_r$. Then there exists a state sequence

   $$(x_0, a_1, x_1, a_2, x_2, \dots, x_{r-1}, a_r, x_r)$$

   such that $x_r$ is final. Then $(\mathcal{Q}_0, a_1, \mathcal{Q}_1, a_2, \mathcal{Q}_2, \dots, \mathcal{Q}_{r-1}, a_r, \mathcal{Q}_r)$, where $\mathcal{Q}_i = \psi(x_i)$, $0 \leq i \leq r$, is a state sequence in $X_{\mathcal{L}}$ and $\mathcal{Q}_r$ is final. Hence $\varepsilon \in \mathsf{Init}(\mathcal{Q}_r)$ and so $a_r \in \mathsf{Init}(\mathcal{Q}_{r-1})$, $a_{r-1}a_r \in \mathsf{Init}(\mathcal{Q}_{r-2})$, and, by induction, we see that $w = a_1a_2 \dots a_r \in \mathsf{Init}(\mathcal{Q}_0)$. Therefore $L_{x_0} \subseteq \mathsf{Init}(\mathcal{Q}_0)$.

2. Conversely, $\mathsf{Init}(\mathcal{Q}_0) \subseteq L_{x_0}$.

   Consider $w \in \mathsf{Init}(\mathcal{Q}_0)$. If $w = \varepsilon$, then $\mathcal{Q}_0$ is final and so $x_0$ is final. Therefore $\varepsilon \in L_{x_0}$. Suppose now that $w = a_1a_2 \dots a_r$. Note that $\mathcal{Q}_0$ is a tree of language trees and so $\mathcal{Q}_0$ is composed of chains of language trees $(T_0, b_1, T_1, \dots, T_{s-1}, b_s, T_s)$ satisfying the conditions of Definition 5.12. Now each $T_i$ is a language tree and so it is composed by language sequences $(L_0, c_1, L_1, \dots, L_{t-1}, c_t, L_{t-1})$ satisfying the conditions of Definition 5.10. Let $T_0$ be the unique prefix of length zero of all chains of language trees of $\mathcal{Q}_0$ and let $L_0$ be the unique prefix of length zero of $T_0$. By the condition of Equation (5.1) in Definition 5.9, there exists a language $L_1$ such that $a_2 \dots a_r \in L_1$, and the language sequence $(L_0, a_1, L_1)$ is in $T_0$, there exists a language $L_2$ such that $a_3 \dots a_r \in L_2$ and $(L_0, a_1, L_1, a_2, L_2) \in T_0$, and, by induction, we see that there exists a language $L_r$ such that the empty word $\varepsilon$ is in $L_r$ and $(L_0, a_1, L_1, a_2, L_2, \dots, L_{r-1}, a_r, L_r) \in T_0$. By Definition 5.12 (2), we obtain that there exists a language tree $T_1$ such that the language sequence $(L_1, a_2, L_2, \dots, L_{r-1}, a_r, L_r)$ is in $T_1$ and $(T_0, a_1, T_1)$ is a chain of language trees in $\mathcal{Q}_0$, and, once again by induction, we find that there exists a language tree $T_r$ such that the language sequence $(L_r)$ belongs to $T_r$ and $(T_0, a_1, T_1, a_2, T_2, \dots, T_{r-1}, a_r, T_r)$ is a chain of language trees in $\mathcal{Q}_0$. By Definition 5.13 (3), there exists a tree of chains of language trees $\mathcal{Q}_1$ such that $(T_1, a_2, T_2, \dots, T_{r-1}, a_r, T_r)$ is a tree of chain of language trees in $\mathcal{Q}_1$ and $(\mathcal{Q}_0, a_1, \mathcal{Q}_1)$ is a state sequence in $X_{\mathcal{L}}$, and so on, with another inductive argument, we find the existence of a tree of chains of language trees $\mathcal{Q}_r$ such that $(T_r) \in \mathcal{Q}_r$ and $(\mathcal{Q}_0, a_1, \mathcal{Q}_1, \dots, \mathcal{Q}_{r-1}, a_r, \mathcal{Q}_r)$ is a state sequence in $X_{\mathcal{L}}$. The state $\mathcal{Q}_r$ is final, because $\varepsilon \in L_r = \mathsf{Init}(\mathcal{Q}_r)$. Since $\psi$ is a homomorphism of automata, there exists a state sequence $(x_0, a_1, x_1, \dots, x_{r-1}, a_r, x_r)$ in $X$ such that $\psi(x_i) = \mathcal{Q}_i$ for $1 \leq i \leq r$ and $x_r$ is final, because $\mathcal{Q}_r$ is final. It follows that $w \in L_{x_0}$. This shows that $\mathsf{Init}(\mathcal{Q}_0) \leq L_{x_0}$ for all $x_0 \in S$.

3. The homomorphism $\psi$ coincides with $\phi$.

   Now let $(x_0, a_1, x_1, \dots, x_{r-1}, a_r, x_r)$ be a state sequence in $X$. Since $\psi$ is a homomorphism of automata,

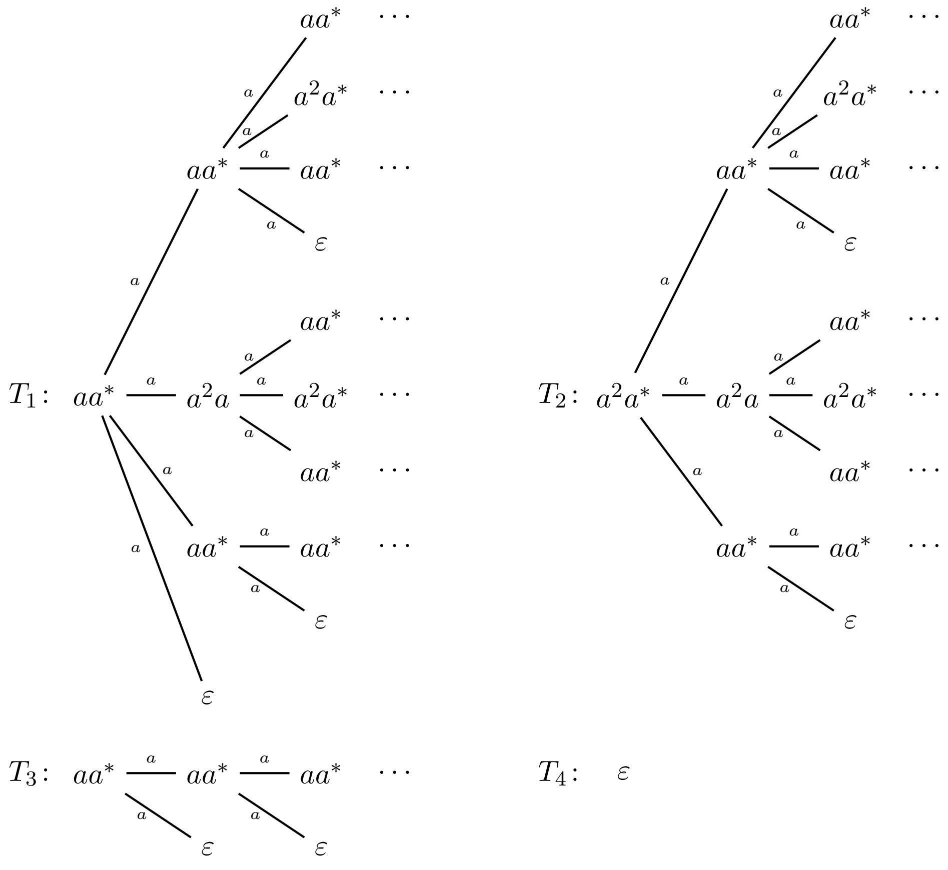
   $$(\psi(x_0), a_1, \psi(x_1), \dots, \psi(x_{r-1}), a_r, \psi(x_r))$$

   is a state sequence in $X_{\mathcal{L}}$. By using an argument similar to the one used in the previous paragraph and the fact that the initial language of $\psi(x)$ is $L_x$, we see that the tree of language sequences $T_0$ of the prefix of length zero of $\mathcal{Q}_0 = \psi(x_0)$ contains the language sequence $(L_{x_0}, a_1, L_{x_1}, \ldots, L_{x_{r-1}}, a_r, L_{x_r})$. Now let $(L_0, a_1, L_1, \ldots, L_{r-1}, a_r, L_r)$ be a language sequence in the tree of language sequences $T_0$ of the prefix of length zero of $\psi(x_0)$. The ideas of the previous paragraph show that there is a chain of trees of language trees $(T_0, a_1, T_1, \ldots, T_{r-1}, a_r, T_r)$ in which the initial language of $T_i$ is $L_i$ for $0 \leq i \leq r$, and that there exists a state sequence in $X_{\mathcal{L}}$ of the form $(\mathcal{Q}_0, a_1, \mathcal{Q}_1, \ldots, \mathcal{Q}_{r-1}, a_r, \mathcal{Q}_r)$ with $\mathsf{Init}(\mathcal{Q}_i) = L_i$ for $0 \leq i \leq r$. The fact that $\psi$ is a homomorphism implies that there exists a state sequence $(x_0, a_1, x_1, \ldots, x_{r-1}, a_r, x_r)$ in $X$ with $\mathcal{Q}_i = \psi(x_i)$ and so the language sequence $(L_0, a_1, L_1, \ldots, L_{r-1}, a_r, L_r)$ coincides with $(L_{x_0}, a_1, L_{x_1}, \ldots, L_{x_{r-1}}, a_r, L_{x_r})$. It follows that $\psi = \phi$. □

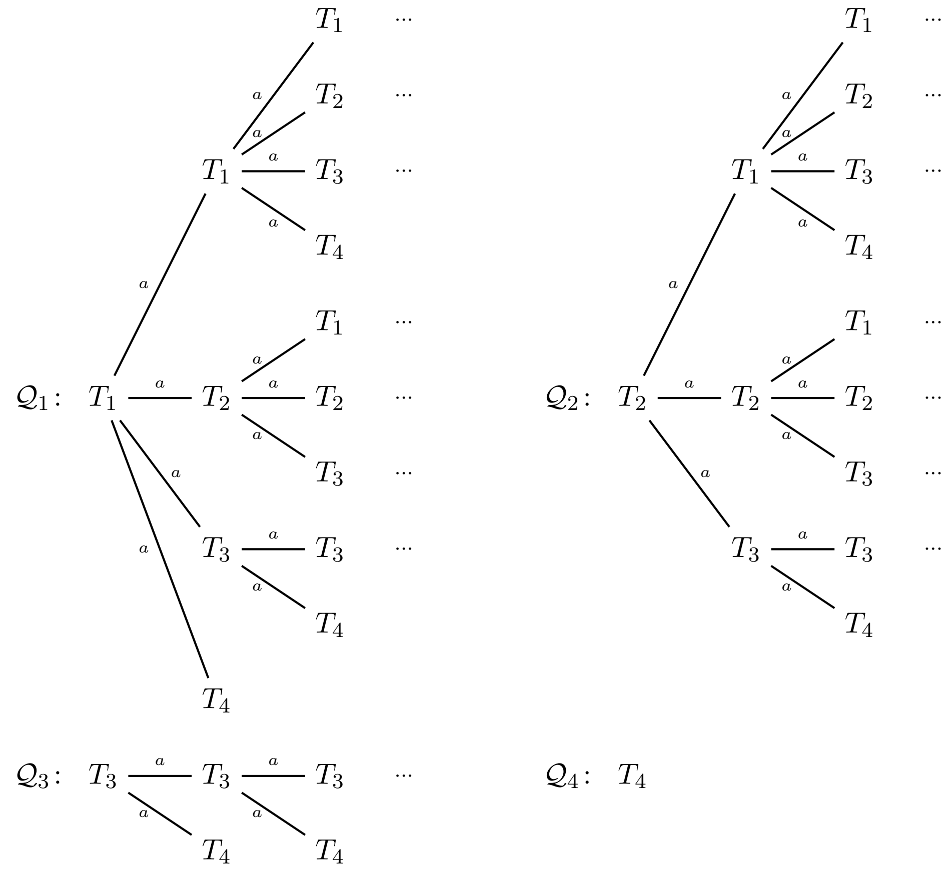
Theorems 5.5 and 5.17 yield the following result.

**Corollary 5.18.** *Given two automata $(X, c, \alpha)$ and $(Y, d, \beta)$ over the same alphabet $A$, two states $x \in X$ and $y \in Y$ are bisimilar if and only if the trees of chains of language trees obtained from $x$ and $y$ according to Construction 5.15 coincide.*

**Example 5.19.** One might think that the final automaton could be constructed in an easier way with the trees of language sequences. This is false. To see this, we consider the automaton of Example 5.6. We cannot determine from the chain of languages $(aa^*, a, aa^*, \ldots, aa^*, a, aa^*)$ whether it corresponds to the state sequence $(1, a, 1, a, \ldots, 1, a, 1)$, to $(1, a, 1, a, \ldots, 1, a, 3)$, or to $(3, a, 3, a, \ldots, 3, a, 3)$. We can represent the images of the states of $(X, c, \alpha)$ in $(X_{\mathcal{L}}, \varepsilon?, \tau)$ as follows.

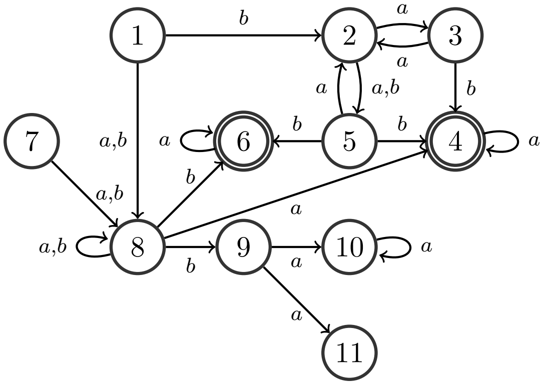


Intuitively, what we do to obtain the images in $(X_{\mathcal{L}}, \varepsilon?, \tau)$ of each state is to substitute each element of the tree by the complete tree which can be formed from this element. It can be depicted as follows



$\mathcal{Q}_1$: $T_1$
$T_1$ $T_2$ $T_3$ $T_4$
$T_1$ ··· $T_2$ ··· $T_3$ ··· $T_4$
$T_1$ ··· $T_2$ ··· $T_3$ ···
$T_3$ ··· $T_4$
$\mathcal{Q}_2$: $T_2$
$T_1$ $T_2$ $T_3$
$T_1$ ··· $T_2$ ··· $T_3$ ··· $T_4$
$T_1$ ··· $T_2$ ··· $T_3$ ···
$T_3$ ··· $T_4$
$\mathcal{Q}_3$: $T_3$ $T_3$ $T_3$ ··· $T_4$ $T_4$
$\mathcal{Q}_4$: $T_4$
a

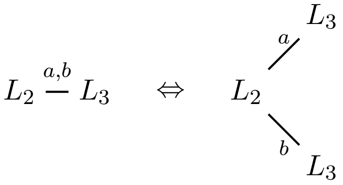
**Example 5.20.** Consider now the automaton $(X, c, \alpha)$ , with $X = \{1, \ldots, 11\}$, $A = \{a, b\}$, 4 and 6 as final states, and transition structure represented in the following diagram

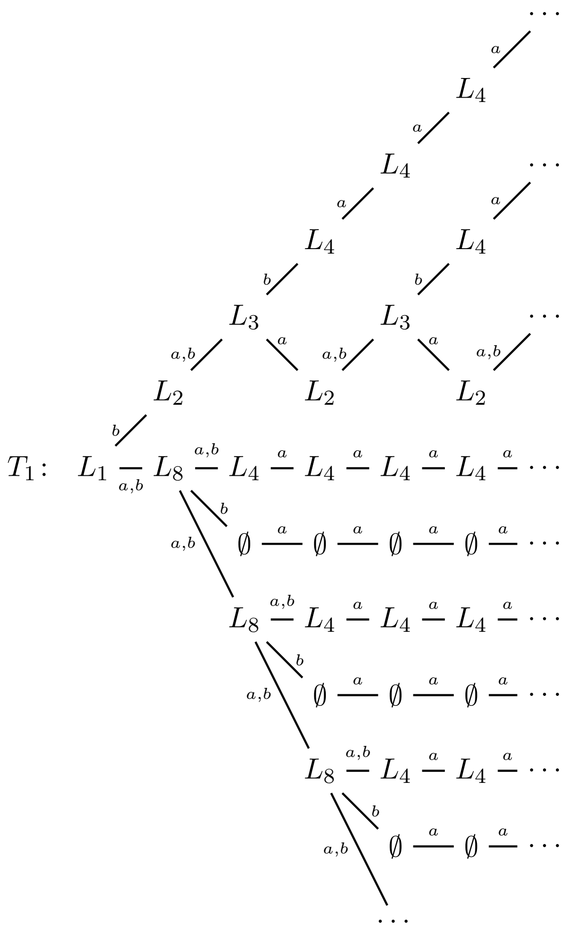


We can use the Automata package [30] of the computer algebra system GAP [34] to check that the languages associated with each of the states are

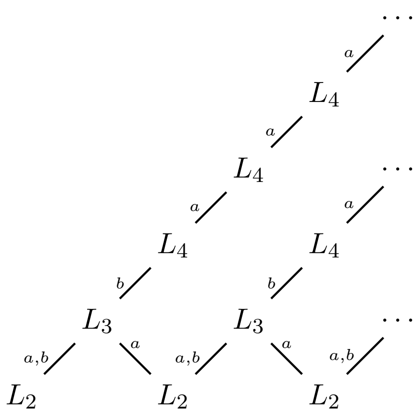
$$\begin{aligned} L_1 = L_7 &= (a+b)^2(a+b)^*, \\ L_2 &= ((a+b)a)^*(a+b)ba^*, \\ L_3 = L_5 &= (a(a+b))^*ba^*, \\ L_4 = L_6 &= a^*, \\ L_8 &= (a+b)(a+b)^*, \\ L_9 = L_{10} = L_{11} &= \emptyset. \end{aligned}$$

As we did above, $L_i = o_c(i)$ for $1 \le i \le 11$. The trees of language sequences corresponding to each state are represented below. A branch labelled with more than one letter is an abbreviation of a multiple branch of the form

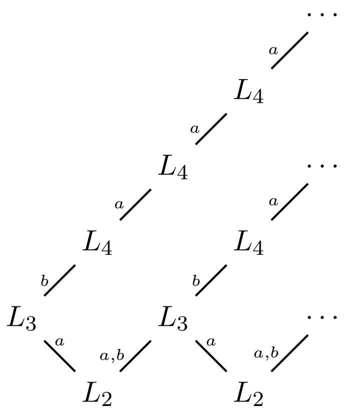




$T_2\colon$

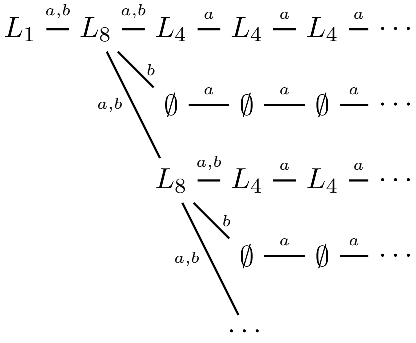


$T_3 = T_5\colon$

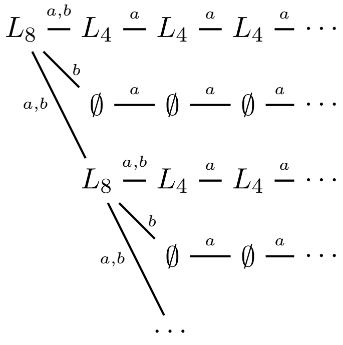


$T_4 = T_6\colon\quad L_4 \overset{a}{-} L_4 \overset{a}{-} L_4 \overset{a}{-} L_4 \overset{a}{-} L_4 \overset{a}{-} \cdots$

$T_7\colon$


$L_1$ a,b $L_8$ a,b $L_4$ a $L_4$ a $L_4$ a ⋯
b
a,b
∅ a ∅ a ∅ a ⋯
$L_8$ a,b $L_4$ a $L_4$ a ⋯
b
a,b
∅ a ∅ a ⋯
⋯

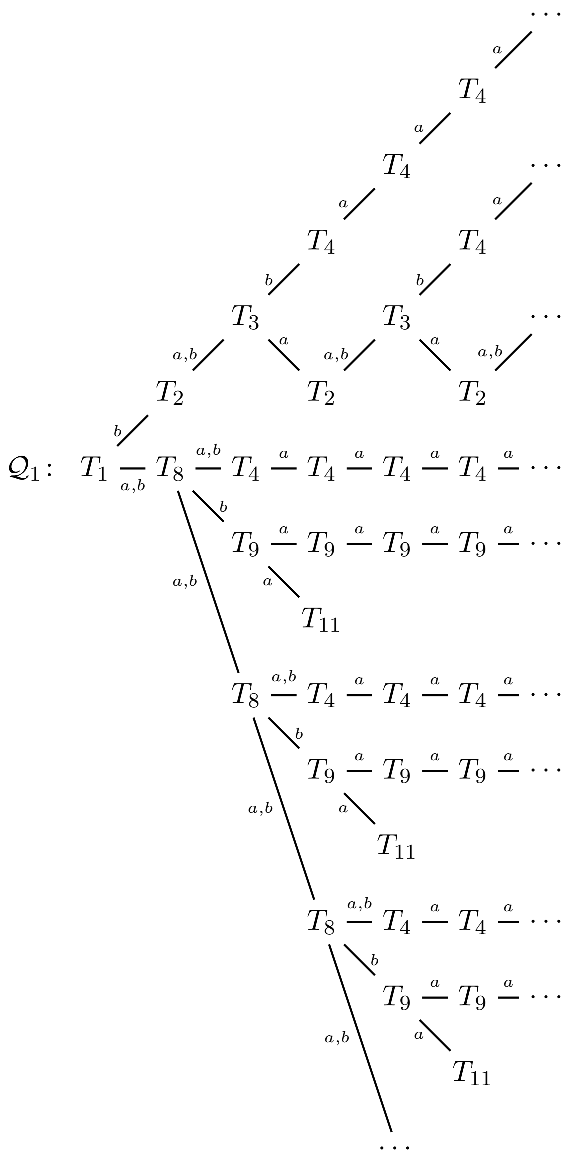
$T_8\colon$

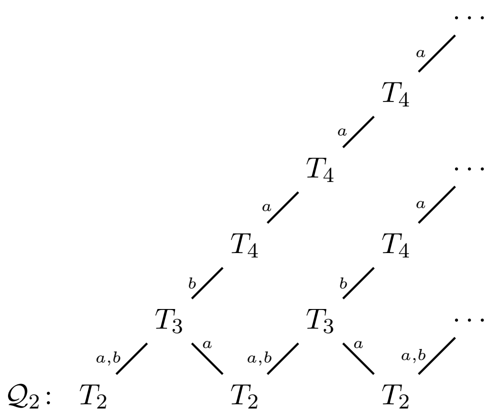

$L_8$ a,b $L_4$ a $L_4$ a $L_4$ a ⋯
b
a,b
∅ a ∅ a ∅ a ⋯
$L_8$ a,b $L_4$ a $L_4$ a ⋯
b
a,b
∅ a ∅ a ⋯
⋯

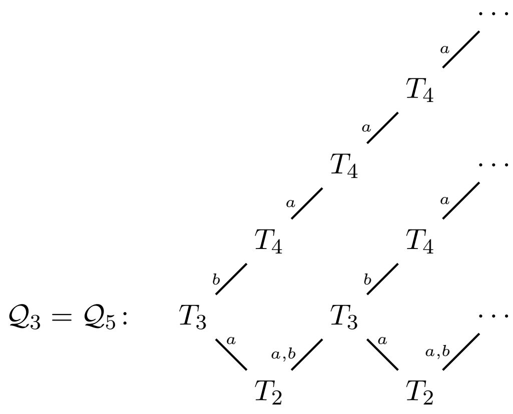
$$T_9 = T_{10}\colon \quad \emptyset \overset{a}{\text{---}} \emptyset \overset{a}{\text{---}} \emptyset \overset{a}{\text{---}} \emptyset \overset{a}{\text{---}} \emptyset \overset{a}{\text{---}} \cdots$$

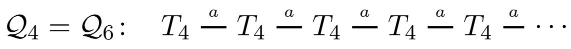
$$T_{11}\colon \quad \emptyset$$

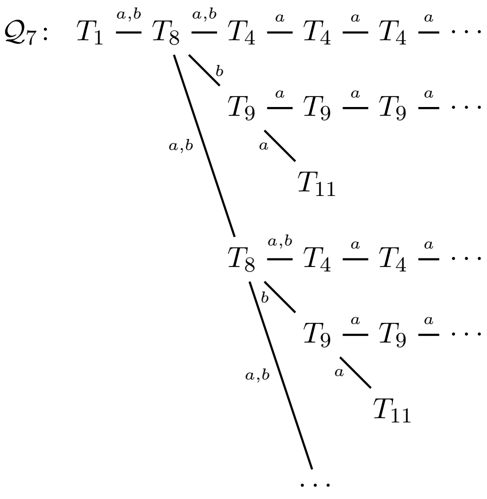
Although the language sequences reachable from the states 9 and 10 are the same, these two states cannot be bisimilar, because from 9 we can make a transition with $a$ to the state 11, which has no transitions, but from 10, the only state we can reach is 10, which has a transition labelled with $a$ to this state. This distinction appears when we consider the trees of chains of language trees, which are given below

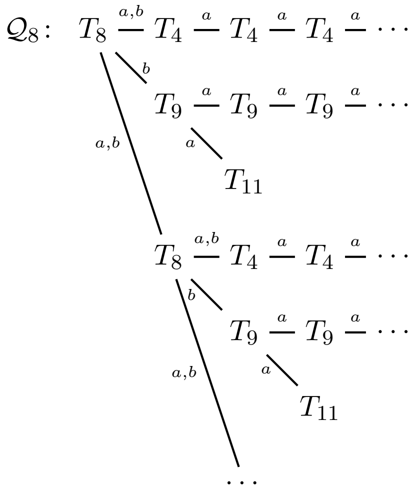


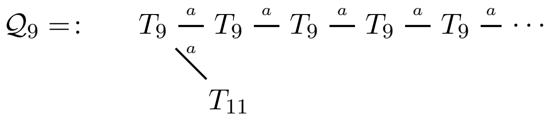

$\mathcal{Q}_2$:
$T_2$
$T_3$
$T_4$
$T_2$
$T_3$
$T_4$
$T_2$
a,b
a
b
a
...


$\mathcal{Q}_3 = \mathcal{Q}_5$:
$T_3$
$T_2$
$T_4$
$T_3$
$T_2$
a,b
a
b
...


$\mathcal{Q}_4 = \mathcal{Q}_6$:
$T_4$
a
...


$\mathcal{Q}_7$:
$T_1$
$T_8$
$T_4$
$T_9$
$T_{11}$
a,b
a
b
...

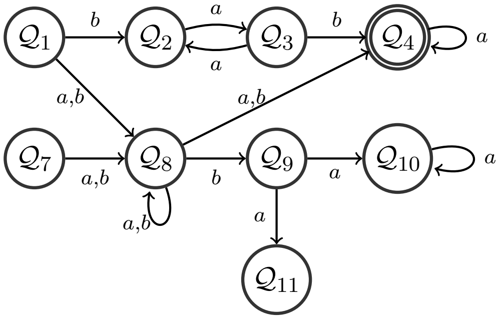




$$\mathcal{Q}_{10} =: \quad T_9 \overset{a}{—} T_9 \overset{a}{—} T_9 \overset{a}{—} T_9 \overset{a}{—} T_9 \overset{a}{—} \cdots$$

$$\mathcal{Q}_{11}\colon\ T_{11}$$

The image of $(X, c, \alpha)$ in the final automaton under the homomorpishm introduced in Theorem 5.17 can be depicted as follows (the states which are not image of any state of $\mathcal{A}$ are not shown).



Note that the only final state is $\mathcal{Q}_4$, because the only language containing $\varepsilon$ was $L_4$, the initial language of $\mathcal{Q}_4$. Of course, this also follows from the fact that the final states of $X$, 4 and 6, are mapped into $\mathcal{Q}_4$. This atuomaton can be regarded as the smallest simple automaton showing the same state behaviour as $X$.

Of course, the subautomaton of $X$ composed by the states 9, 10 and 11 and the corresponding transitions is enough to show that the trees of language sequences are not enough to describe the final automaton. We have presented this more complicated example to show how to work with alphabets consisting of more than one letter.

**Example 5.21.** Consider now the automaton $(X, c, \alpha)$ given by $X = \{1, 2, 3, 4\}$, $A = \{a\}$, $1_a = \{1, 2, 3, 4\}$, $2_a = \{1, 2, 3\}$, $3_a = \{3, 4\}$, $4_a = \emptyset$, and no final states. This automaton is like the one in Example 5.6, but with no final states. Obviously, all states have associated the empty language $\emptyset$. The trees of languages associated to this automaton are like the ones represented in Example 5.6, but with all languages replaced by $\emptyset$. In this case, only the branching information of the automaton is used. The corresponding images in the final automaton look like the ones represented there with the trees $T_i$ coming from the ones of Example 5.6. The automaton is also simple.

This technique of considering non-deterministic automata for a language of one letter and no final states can be used to simulate coalgebras for the finite power-set functor $\mathcal{P}_\omega$. Since all languages are empty, the languages turn out to be irrelevant in our discussion for this kind of automata.

## 5.3 Discussion and future work

We have obtained a description for the final object in the category of non-deterministic coloured automata in terms of languages. We have also proved that the observational behaviour of an automaton (bisimilarity) can be described in terms of the languages accepted from each state. In our approach, it is just an equality of sets obtained from the languages associated with the states of the automaton. This generalises a known fact for deterministic automata, as the language automaton $(2^{A^*}, \varepsilon?, \tau)$ introduced in the beginning of Chapter 3, but which did not seem evident for non-deterministic automata as we have seen in Example 5.6.

In the following paragraphs, we shall present some descriptions of final coalgebras for some functors in the category $\mathsf{Set}$. Bonsangue, Rutten, and Silva (see [25, 73, 74]) have considered categories of coalgebras for Kripke polynomial functors in the category $\mathsf{Set}$ of sets and functions, which include automata, and have described the subcoalgebra of the final coalgebra containing the images of the corresponding finite coalgebras. In their description, they construct a set of expressions based on the elementary components of the functor and an equivalence relation between these expressions. The quotient set of these expressions modulo this equivalence relation admits a structure of a coalgebra for this functor which turns out to be the subcoalgebra of the final coalgebra containing the images of the finite coalgebras.

The finite power-set functor $\mathcal{P}_\omega$ and other related functors on the category **Set** have deserved special attention. A non-ordered finitely branching tree is said to be *extensional* if subtrees rooted at distinct children are not isomorphic. From one tree, it is possible to obtain an extensional quotient by identifying two identical subtrees of nodes of the tree and repeating it for a possibly transfinite number of steps. We say that two trees are *extensionally equivalent* when they reduce to the same extensional tree, and are *similar* when the trees of depth $n$ obtained by truncation are extensionally equivalent for all $n$. Barr [16] described the final $\mathcal{P}_\omega$-coalgebra as the quotient coalgebra of the coalgebra composed of all extensional finitely branching trees modulo this relation of similarity. Another relevant description of the final coalgebra for the power-set functor was given by Worrell in [80] (see also Adámek *et al.* [3]). Let us call a tree $t$ *strongly extensional* if for every $n$ there exists $m \geq n$ such that the truncation of depth $n$ of $t$ coincides with the truncation of depth $n$ of the result of taking the truncation of depth $m$ of $t$ and collapsing it with respect to extensional equivalence. The set $T$ of all finitely branching, strongly extensional trees has a coalgebra structure $\alpha\colon T \to \mathcal{P}_\omega(T)$

assigning to every tree the set of all maximal proper subtrees. This $\mathcal{P}_\omega$-coalgebra is final.

Kozen [48] has presented a combinatorial description of final coalgebras on **Set**. In his work, the role of the functor is played by what he calls a *type signature*, which is a directed multigraph whose nodes are designated as *universal* or *existential*. Universal nodes, denoted by rectangles, correspond to product constructors, while existential nodes, denoted by diamonds, correspond to coproduct constructors. If $\mathcal{F}$ is a type signature, an $\mathcal{F}$*-realisation* is a directed multigraph $G$ together with a multigraph homomorphism $l\colon G \to \mathcal{F}$, called a *typing*, satisfying the following properties:

- If $l(u)$ is existential, then there is exactly one edge of $G$ with source $u$.
- If $l(u)$ is universal, then $l$ is a bijection between the edges of $G$ with source $u$ and the edges of $\mathcal{F}$ with source $l(u)$.

A homomorphism of $\mathcal{F}$-realisations is a multigraph homomorphism that commutes with the types. Let $\mathcal{F}$ be a type signature with nodes $V_\mathcal{F}$. An $\mathcal{F}$*-coalgebra* is a $V_\mathcal{F}$-indexed collection of pairs $(X_s, \alpha_s)$, where the $X_s$ are sets and the $\alpha_s$ are set functions

$$\alpha_s\colon X_s \longrightarrow \begin{cases} \sum_{\operatorname{src} e = s} X_{\operatorname{tgt} e}, & \text{if } s \text{ is existential,} \\ \prod_{\operatorname{src} e = s} X_{\operatorname{tgt} e}, & \text{if } s \text{ is universal,} \end{cases}$$

where $\operatorname{src} e$ and $\operatorname{tgt} e$ denote, respectively, the source and the target of the arc $e$.

A morphism of $\mathcal{F}$-coalgebras is a $V_\mathcal{F}$-indexed collection of set maps $h_s$ that commute with the $\alpha_s$ in the usual way. This corresponds to the traditional definition of a coalgebra for an endofunctor on $\mathbf{Set}^V$. If the type signature is *accessible*, that is every node is accessible from a fixed node, then it is possible to find an endofunctor $F$ on **Set** such that the categories of $F$-coalgebras and $\mathcal{F}$-coalgebras are naturally isomorphic.

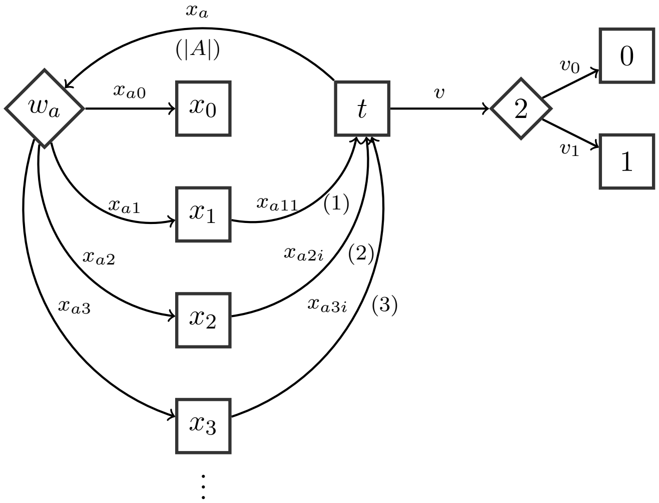
Kozen showed the existence of a pair of functors between the category of $\mathcal{F}$-coalgebras and the category of $\mathcal{F}$-realisations, one in each direction, that are inverses up to natural isomorphisms. He proves that these two categories are equivalent and, as a consequence, we can obtain a description of the final $\mathcal{F}$-coalgebra from the final $\mathcal{F}$-realisation.

The final object for the category of $\mathcal{F}$-realisations is showed to be the realisation $(R_\mathcal{F}, l_\mathcal{F})$ defined as follows. A node of $R_\mathcal{F}$ is a set $Y$ of finite paths in $\mathcal{F}$ such that:

1. $Y$ is non-empty and prefix-closed;
2. all paths in $Y$ have the same first node, called $l_\mathcal{F}(Y)$;
3. if $p$ is a path in $Y$ of length $n$ and its tail node is existential, then there exists exactly one path of length $n+1$ in $Y$ extending $p$;
4. if $p$ is a path in $Y$ of length $n$ and its tail node is universal, then all paths of length $n+1$ extending $p$ are in $Y$.

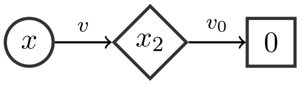
The arcs of $R_\mathcal{F}$ are defined as follows. Let $Y$ be a set of paths in $\mathcal{F}$ and $e$ an arc of $\mathcal{F}$. The *Brzozowski derivative* of $Y$ with respect to $e$ is the set $\mathrm{D}_e(Y)$ of paths obtained by removing the initial edge $e$ from all paths in $Y$ starting with that edge. If $Y$ is a node of $R_\mathcal{F}$ and $\mathrm{D}_e(Y)$ is non-empty, we add exactly one edge $\langle Y, e\rangle$ from $Y$ to $\mathrm{D}_e(Y)$ in $R_\mathcal{F}$ and we make $l_\mathcal{F}(\langle Y, e\rangle) = e$. As shown in [48, Theorem 3.2], this realisation is a final object in the category of $\mathcal{F}$-realisations.

We have not found in [48] the description of a type signature corresponding to non-deterministic automata. Nevertheless, from the examples in this paper we see that a possible signature type for non-deterministic automata is the graph drawn on below, where the nodes with label $t$, 0, 1, and $x_i$, $i \in \mathbb{N} \cup \{0\}$, are universal and the node labelled as 2 and the nodes $w_a$, $a \in A$, are existential; for every $a \in A$ there exists an arc $x_a$ from $t$ to $w_a$ and an arc $v$ from $t$ to 2; there is an arc $v_0$ from 2 to 0 and an arc $v_1$ from 2 to 1; from $w_a$ to $x_i$, $i \in \mathbb{N} \cup \{0\}$, there is an arc $x_{ai}$, and from $x_i$ to $t$, $i \in \mathbb{N}$, there are $i$ arcs labelled as $x_{aij}$, $1 \leq j \leq i$.

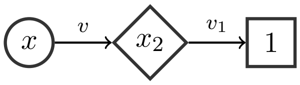


In the following we will describe a non-deterministic coloured automaton $(X, c, \alpha)$ as an $\mathcal{F}$-realisation $(G, l)$. We introduce a procedure to construct a multigraph starting from the graph of the automaton. To every state $x$ in the graph, depending on its nature, we will add the following multigraphs:

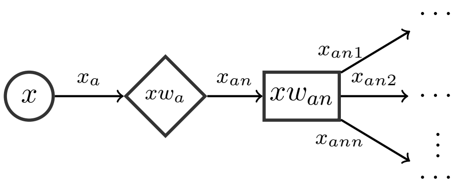
- If $x$ is not a final state, we add:



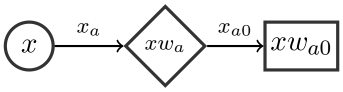
- If $x$ is a final state, we add:



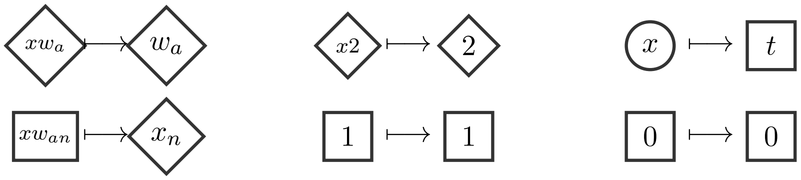
- For every input letter $a \in A$, if $x$ has $n$ $a$-labelled outgoing arcs, we replace them by:



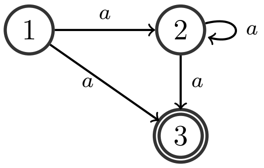
- If $x$ has no $a$-labelled outgoing arcs, we add:



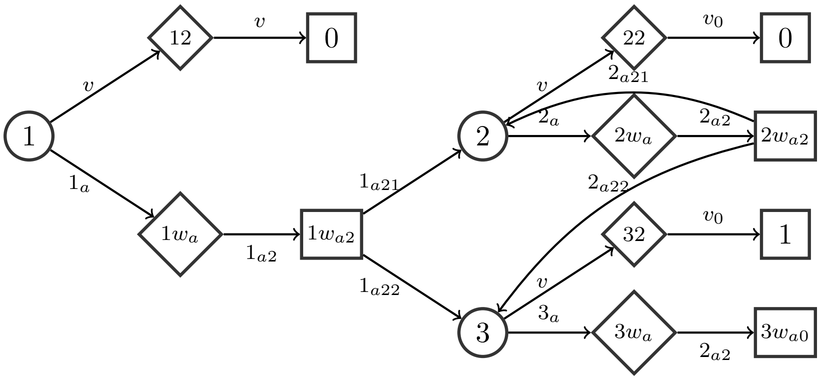
This procedure will give us a multigraph. To complete the description of the realisation we specify its typing $l$ on the final realisation as follows:



**Example 5.22.** Let us exemplify the last procedure on the small automaton $(X, c, \alpha)$ with set of states $X = \{1, 2, 3\}$, alphabet $A = \{a\}$, 3 as the unique final state, and transitions depicted below.

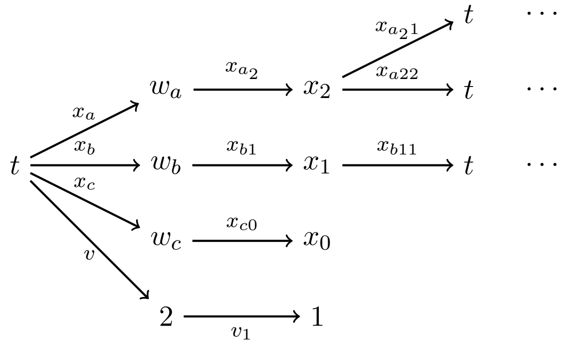


The result of the previous procedure is represented on the following diagram.



The previous description of the final $\mathcal{F}$-realisation applied to this signature type is the first description we know for the final object for the category of non-deterministic automata that is not given in terms of equivalence classes of a bisimilarity relation, in the sense that in the final automaton, bisimilarity is just a set equality. Kozen also shows at the end of the paper [48] how to characterise the elements of the final realisation as labelled trees.

As we have mentioned, automata can be regarded as $\mathcal{F}$-realisations in the sense of Kozen [48] for the type signature $\mathcal{F}$ presented above. We now outline how to pass from Kozen's description to our description and vice versa. The nodes of the final $\mathcal{F}$-realisation can be regarded as trees like in the example below, which corresponds to the image in the final automaton for the alphabet $A = \{a, b, c\}$ of a final state with two transitions labelled by $a$, a transition labelled by $b$ and no transitions labelled by $c$.



We can associate to this state the language corresponding to all words $a_1 \dots a_k$ such that there exists a path starting with $t$ whose edges are labelled

$$(x_{a_1}, x_{a_1 i_1}, x_{a_1 i_1 j_1}, \dots, x_{a_k}, x_{a_k i_k}, x_{a_k i_k j_k}, v, v_1).$$

We can generate the corresponding language tree by replacing each $t$ by the corresponding language, the path composed by three edges $(x_a, x_{ai}, x_{aij})$ by $a$ and by deleting the paths composed by the edges $(v, v_0)$ or $(v, v_1)$, as well as the paths composed by the edges $(x_a, x_{a0})$. By substituting each state $t$ by the corresponding language tree, we get the tree of chains of language trees of our construction. Conversely, given a state of the language tree automaton and a letter $a \in A$, we can associate to it the following paths:

- First, the path composed by $(v, v_1)$ if $\varepsilon$ belongs to its initial language and $(v, v_0)$ otherwise.

- Let $a \in A$.

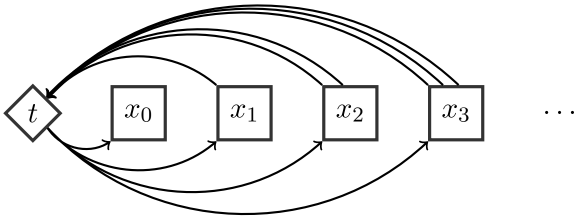
  - If there are no transitions labelled with $a$ from this state, we simply add the path $(x_a, x_{a0})$.

  - If there are $i$ transitions labelled with $a$ from this state, we assign the paths whose edges have the labels $(x_a, x_{ai}, x_{aij})$, for $1 \leq j \leq i$, followed by all paths corresponding to the images of the transition of this state by $a$ obtained with this method.

The coinduction principle (see Rutten [63]) guarantees that this construction is possible.

Our structures derive from the branching information of the automata, with the states substituted by their corresponding languages and, in some sense, follow the same ideas of Barr [16] and Worrell [80] about the branching information. However, even some natural candidates for the states of the final non-deterministic automaton, as the one presented in Example 5.19, based only on the branching information of the automata with the states replaced by their corresponding languages, are not good enough for our purposes.

A slight modification of this type signature, drawn below, gives the type signature corresponding to the $\mathcal{P}_\omega$-coalgebras, where $\mathcal{P}_\omega$ is the finite power-set functor.



Its final realisation can be obtained from the strongly extensional trees of Worrell [80] by replacing the edges of the form $x \to y$ by a path $t \to x_i \to t$, where $i$ is the number of children of $x$, and a leaf $x$ is replaced by a path $t \to x_0$. Hence the strongly extensional trees are recovered with this description. This construction can be compared with the interesting construction of Example 5.21. In fact, the description of infinite trees modulo bisimilarity presented by Barr in [16] or the strongly extensional trees of Worrell in [80], are recovered with our description.

Barr [16] and Worrell [80] have presented a description of final objects in $\mathcal{P}_\omega$-coalgebras by means of suitable infinite trees modulo bisimilarity, which exploit their branching information. However, if we want to describe bisimilarity by means of the final object, this approach is not sufficient, because it could be like a *petitio principii.* A precise description of the relation is achieved in this chapter by means of the description of the language tree automaton and the homomorphism from a given automaton to the language tree automaton. Nevertheless, as we have mentioned in the previous section, we obtain trees isomorphic to the strongly extensional trees of Worrell [80] when we use automata to simulate $\mathcal{P}_\omega$-coalgebras.

Some recent descriptions of minimisations of non-deterministic automata have been presented by Brzozowski and Tamm [27] and Adámek, Bonchi *et al.* [2]. We mention them here because they are based on the languages associated to every state of the automaton. However, their way of minimising automata differs from ours, since they only pay attention to the languages associated to every state instead of bisimilarity, as we do. We present them here in order to show the differences with our approach. The problem considered there is the following. Given a regular language $L$ over an alphabet $A$, minimal deterministic automata can be considered as *canonical* acceptors of the given language $L$. The question is whether it is possible to find an analogous canonical non-deterministic automaton. In [27], the quotients $L_1, L_2, \ldots, L_n$ of the form $L_w$ of a given regular language $L$ are considered. The non-empty intersections of languages of the form $\widehat{L_1} \cap \cdots \cap \widehat{L_n}$ such that $\widehat{L_i}$ is equal to $L_i$ or to its complement $\overline{L_i}$ in which at least one of the $L_i$ is not complemented are called the *atoms* of $L$. The non-deterministic automaton having the atoms of $L$ as languages as states is called the *átomaton* of $L$.

For a non-deterministic finite automaton, its *determinisation* is the deterministic finite automaton obtained by the well-known subset construction, where only subsets (including the empty subset) reachable from the initial state of the automaton $X$ are used. In [27], the authors show that the determinisation of the átomaton of a regular language $L$ coincides with the minimal deterministic automaton associated to this language.

In [2], a coalgebraic point of view of this kind of description is presented. However, non-deterministic automata are considered there as coalgebras for the functor $A \times Id + 1 \colon \mathbf{Rel} \longrightarrow \mathbf{Rel}$, where $\mathbf{Rel}$ denotes the category of sets and relations. The final object in this category is $A^*$, and the unique morphism is the relation which assigns to each state all the words sending this state to an accepting state. Under this interpretation, bisimilarity is just language equality. This point of view is different from the one used in this chapter. Equivalent descriptions of this automaton can be found in both papers and in the references inside them.

For the case of automata, regarded as labelled transition systems, the description presented here do not give, in our opinion, a clear idea of the role of languages in the final automaton. The description of the language tree automaton is indeed a generalisation of the description of the language automaton. In the case of a deterministic automaton $(X, c, \alpha)$, for each state $x$ and each letter $a \in A$, there exists a unique transition

$$x \xrightarrow{a} x_a$$

and the corresponding languages satisfy the relation $o_c(x_a) = o_c(x)_a$. This property also holds in the language automaton $(2^{A^*}, \varepsilon?, \tau)$, in which the transitions have the form

$$L \xrightarrow{a} L_a$$

This implies that the language sequences associated to state sequences in a deterministic automaton are uniquely determined by their initial languages. The same can be affirmed about language trees, chains of language trees, and trees of chains of language trees associated to state sequences of deterministic automata, which are also uniquely determined by their initial languages.

The computation of the image of a non-deterministic automaton in the language tree automaton solves a problem of minimisation of automata by Corollary 5.18. The image of a given automaton is a simple automaton, that is, given two different states, they are not bisimilar. The corresponding minimisation problem for deterministic automata is solved by means of the equality of the languages recognised from the states. Other known algorithms are available to identify bisimilarity and so to construct this image into the final automaton, like state partition algorithms (see, for instance, [1]).

We must observe that our automata are not necessarily finite. In fact, the final automaton is infinite. The same thing happens with the final deterministic automaton, whose states are all languages: it is infinite and non-regular languages can appear as states. However, the set of all states reachable from one state by the action of one letter is kept finite in order to make sure that the states of the final object form a set.

A future research line in this subject could be to apply these techniques to study final coalgebras for other structures which can have languages associated with the states in a natural way. This could be an alternative approach to the descriptions of [16, 24, 25, 48, 73, 74, 80]. For instance, the ideas of Example 5.21 show a possible way to construct the final object for the category of all coalgebras associated with the finite power-set functor.

Another possible future research line could be finding alternative semantics for other structures admitting a coalgebra structure. As an example of what we mean, we might consider the Hennessy-Milner logic. Let $(X, c, \alpha)$ be a non-deterministic coloured automaton. We can define a multi-modal logic $\mathsf{M}(X, c, \alpha)$ with an atomic proposition $p$ whose semantics is given by set of formulas $\mathcal{L}$ defined by the grammar

$$\phi ::= \mathtt{tt} \mid \neg\phi \mid \phi_1 \wedge \phi_2 \mid \Diamond_a \phi \mid p$$

where $a$ varies over $A$. This logic is called *Hennessy-Milner logic* because it was introduced by Hennessy and Milner in [42, 43] (see also [76] for more details). The usual interpretation of the formulas is given by the modelling relation $\models \subseteq X \times \mathcal{L}$ defined by

- $x \models \mathtt{tt}$,

- $x \models \phi_1 \wedge \phi_2$ if and only if $x \models \phi_1$ and $x \models \phi_2$,
- $x \models \neg\phi$ if and only if $\neg(x \models \phi)$,
- $x \models \Diamond_a \phi$ if and only if there exists $y \in x_a$ such that $y \models \phi$,
- $x \models p$ if and only if $c(x) = 1$,

The extension of the Hennessy-Milner logic with fixed point operators is the modal $\mu$-calculus. A detailed study of the Hennessy-Milner logic and the modal $\mu$-calculus, as well as bisimilarity and different semantics for them, can be found in [76]. The trees of chains of language trees over $X$ defined from the underlying automaton $(X, c, \alpha)$ could be used to give an alternative semantics for the Hennessy-Milner logic.

> Ell va dir que en casos com aquest
> no es tracta de ser més guapo
> o de ser més lleig,
> sinó d'estar convençut de fer-ho.
>
> Jean-Luc — Els Amics de les Arts

# CHAPTER 6

## Conclusions

Automata are very interesting mathematical objects that had engaged the attention of researchers over the past few decades. These structures can be studied from many perspectives as they are located at the crossroads between several growing mathematical fields, mainly boosted from Computer Science; researchers in formal language theory, representation theory, computation and monoid theory, among many other examples, may benefit from automata theory. Despite its apparent simplicity, these structures had already proven succesful in the study of several interesting problems.

In our work, we wanted to highlight the dual nature of these structures that are, at the same time, an example of algebra and coalgebra. In this thesis we introduced the ground notions to what we wanted to be a systematic study of automata both from an algebraic and a coalgebraic perspective. Accordingly, in Chapter 2 we introduced some related notation, terminology and basic well-known results used in this thesis.

In Chapter 3 we introduced the algebra-coalgebra description of an automaton. The interesting cases of the free monoid with concatenation and the set of all languages with the Brzozowski derivative were shown to be the initial algebra and the final coalgebra, respectively, for the case of deterministic automata. The notions of *equations* and *coequations* were there defined and from these concepts we constructed the free and cofree automata associated to a given automaton. They represent, respectively, the largest set of equations and the smallest set of coequations satisfied by the automaton considered. When suitably restricted, these constructions based on equations and coequations are proved to be functorial. The main contribution of Chapter 3 is Theorem 3.24, which presents a description of a dual equivalence between the categories of congruence quotients with epimorphisms, living in the algebraic side, and preformations of languages with monomorphisms, living in the coalgebraic side. For working purposes, this duality gives rise to the concise expression of Corollary 3.25

$$L \in \mathsf{coEq}(A^*/C) \qquad \Leftrightarrow \qquad C \subseteq \mathsf{Eq}\langle L\rangle$$

for every congruence $C$ in $A^*$ and every language $L$ in $2^{A^*}$. In Chapter 3 we also present some examples of the usefulness of these constructions and we introduced the notion of equational bisimulations, which is a method that can be implemented to prove that a language satisfies a given set of equations. Moreover, the syntactic monoid, the Myhill-Nerode relation and the syntactic congruence associated to a formal language, among other scattered notions already appearing in the literature, are presented within a common framework.

In Chapter 4 we have seen how Theorem 3.24 relates to the Eilenberg's variety theorem. In the first place, we gave an alternative definition of the original notion of variety of regular languages in terms of equations and coequations. This new definition clearly underscores the role of congruences in this kind of results. An appropiate variation of this definition can be used to prove an Eilenberg's theorem for formations of non-necessarily finite monoids. In this case we relate three concepts that are proven to be equivalent; formations of monoids, formations of congruences, and formations of languages.

The relation between formation of monoids and formations of congruences, although natural, seems completely new and it opens up a truly new research line relating classes of monoids and lattices of congruences in the free monoids. The second correspondence is given for formations of congruences and formations of languages. In this case, we have seen that most of the steps in the proofs heavily depend on the use of Corollary 3.25. This Chapter clearly underscores the deep relation between monoids, congruences in the free monoid (identities between words), and languages. These bindings, already presented in Chapter 3 for objects, also hold for classes. As we mentioned earlier, we see Chapter 4 as the best possible example to show how useful the duality result is. We give and end to this Chapter by showing that varieties of monoids, in the sense of Birkhoff, can also have an Eilenberg's like theorem. What we see is that these classes are a very special kind of formation. From an algebraic point of view they are clearly determined by the existence of a locally-free object for each alphabet, given by the quotient under the residual. This residual, in turn, being the smallest possible congruence whose quotient is in the variety. It follows that the corresponding formation of congruences is, for each alphabet, a principal filter. And, finally, on the langauge side, cofree of the congruence quotient under the residual is shown to be the greatest preformation of languages containing all languages whose syntactic monoid is in the variety.

The last Chapter contains an incursion to the coalgebraic study of non-deterministic automata. Although some other authors have been given descriptions of the final non-deterministic automata, we saw that none of them explicitly showed their relation with languages recognised by an automaton. For the case of deterministic automata this connection is clearly presented; the automata homomorphism of an automaton in the final object is given by its observations under the different colourings. This is not the case for non-deterministic automata (see example 5.6) and more complex structures need to be considered. In our case, we show that bisimilarity in a non-deterministic automaton is captured with trees of chains of language trees. This characterisation, althought based on languages, does not seem to work for a duality result as we did in Chapter 3. Non-determinism seems to vanish the strong connection between colourings and considering sets of states in the automaton.

## 6.1 Further research directions

This dissertation on automata is, by all means, not extensive and some recent results on the field have not been added to the present study.

One of the main topics for further research is related with the use of topological techniques on duality results. Stone duality, for example, is used to obtain further stronger results for regular languages and finite monoids. The connection between profinite words and Stone spaces was already discovered by Almeida [6, 7]. However, it was Pippenger in [56] the first to formulate it in terms of Stone duality. They both observed that the Boolean algebra of regular languages over $A^*$ is dual to the Stone space $\widehat{A^*}$ of profinite words. This duality extends to a one-to-one correspondence between Boolean algebras of regular languages and quotients of $\widehat{A^*}$.

On the other hand, the modern rendering of the "implicit identities" as stated by Reiterman [59] appear in the work done by Gehrke [36, 37] and Pin [38] as profinite identities. In their work they show that *lattices of languages* are precisely those classes of regular languages being defined by profinite identities [38]. It follows from these works the strong connection between classes of regular languages, finite monoids and sets of profinite identities. This approach is also very useful to establish effective decision procedures.

This correspondence strongly depends on profinite techniques. Recall that the profinite monoid $\widehat{A^*}$ can be constructed both as the completion of an ultrametric defined on $A^*$ or as the projective limit of all finite monoids whose generators are in $A$ (See [38] and [8], respectively). Indeed, our results in the monoid side refer to objects ($A^*/C$, for some congruence $C$ on $A^*$) and the results on [56], [38], [36] and [37] refer to limit constructions (profinite monoid). Indeed, the monoid $\widehat{A^*}$ cannot be written as $A^*/C$ for some congruence $C$ on $A^*$ and, therefore, our results per se do not apply.

However, the functorial approach we present here could be used to retrieve a similar situation. Projective limits (the profinite monoid $\widehat{A^*}$) and inductive limits (the set of all regular languages $Reg(A^*)$) are categorical limits in which all arrows involved are epi or mono, respectively. So far, we know that the category $\mathcal{A}_m$ has inductive limits, whereas $\mathcal{A}_e$ require an additional argument to guarantee that the mediating map from the limit to the monoids is epi. At this point, it seems necessary to appeal to topological arguments (see for instance [60, Lemma 3.1.27]). If such limits in both $\mathcal{A}_m$ and $\mathcal{A}_e$ exist, our equivalence will preserve both limits and colimits and we will retrieve a similar result on limit constructions. All in all, this line of future work deserves further study.

The present paper already contains some contributions that encourage us to continue working along these lines. The first relevant insight is that we are able to deal with *infinite* automata and non-regular languages. It lies in the fact that the duality we find is the (conceptually simpler) discrete duality between sets and complete atomic Boolean algebras. The latter duality is also used in [61], where it was lifted to a dual equivalence between deterministic automata and so-called Boolean automata. We hope to retrieve some of the results presented in the papers [6, 7, 56, 38, 36, 37], specially Reiterman's characterisation in terms of profinite equations.

In connection with this research line, it would be interesting to see how some conditions on formations reflect on the corresponding filters of congruences, as we discussed at the end of Chapter 4. Specially, finiteness conditions seems to be good candidates for retrieving similar situations as those we found in the papers on profinite techniques. All in all, further limit constructions of non-necessarily finite monoids need to be investigated.

## 6.2 Derived works

Some of the main contributions appearing in this dissertation have been submitted or accepted for publication as research papers. The corresponding bibliographical references are enumerated below.

- [65] J.J.M.M. Rutten, A. Ballester-Bolinches, and E. Cosme-Llópez. **Varieties and covarieties of languages (preliminary version).** In D. Kozen and M. Mislove, editors, Proceedings of MFPS XXIX, volume 298 of *Electron. Notes Theor. Comput. Sci.*, pages 7–28, 2013.

- [11] A. Ballester-Bolinches, E. Cosme-Llópez, and R. Esteban-Romero. **A description based on languages of the final non-deterministic automaton.** *Theoretical Computer Science*, 536(0):1 – 20, 2014.

- [14] A. Ballester-Bolinches, E. Cosme-Llópez, and J. Rutten. **The dual equivalence of equations and coequations for automata**. *Information and Computation*, 244:49 – 75, 2015.

- [13] A. Ballester-Bolinches, E. Cosme-Llópez, R. Esteban-Romero, and J. Rutten. **Formations of monoids, congruences, and formal languages**. 2015. Submitted.

During the development of this thesis, other relevant problems have been addressed. We present these other works for any possible reference purposes. Although not directly, they also deal with problems related with those presented here.

- [68] J. Salamanca, A. Ballester-Bolinches, M.M. Bonsangue, E. Cosme-Llópez, and J.J.M.M. Rutten. **Regular varieties of automata and coequations**. In Ralf Hinze and Janis Voigtländer, editors, Mathematics of Program Construction, volume 9129 of *Lecture Notes in Computer Science*, pages 224–237. Springer International Publishing, 2015.

- [12] A. Ballester-Bolinches, E. Cosme-Llópez, and R. Esteban-Romero. **Group extensions and graphs**. *Expositiones Mathematicae*, 2015. Accepted manuscript (unedited version) available online: 20th July 2015.

- A. Ballester-Bolinches, E. Cosme-Llópez, and P. Jiménez-Seral. **Some contributions to the theory of transformation monoids**. Submitted, 2015.

- A. Ballester-Bolinches, E. Cosme-Llópez, and S. F. Kamornikov. **On subgroup functors of finite soluble groups**. Submitted, 2015.

> Has esgarrat un feix d'hores cercant
> un mitjà que t'empente en avant, però
> et disperses perquè no hi ha horitzons,
> ni confins, et rellisca en les mans, i ho
> tems, estrany, perquè no és real, no hi
> ha final, no, torna a ser tot igual.
>
> Russell — Sitja

# Bibliography

[1] L. Aceto, A. Ingolfsdottir, and J. Srba. The algorithmics of bisimilarity. In D. Sangiorgi and J. Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*, pages 100–172. Cambridge University Press, 2012.

[2] J. Adámek, F. Bonchi, M. Hülsbusch, B. König, S. Milius, and A. Silva. A Coalgebraic Perspective on Minimization and Determinization. In *Foundations of Software Science and Computational Structures - 15th International Conference, FOSSACS 2012, Held as Part of the European Joint Conferences on Theory and Practice of Software, ETAPS 2012, Tallinn, Estonia, March 24 - April 1, 2012. Proceedings*, pages 58–73, 2012.

[3] J. Adámek, S. Milius, S. Moss, and L. Sousa. Power-set functors and saturated trees. In M. Bezem, editor, *Computer Science Logic (CSL'11)- 25th International Workshop 20th Annual Conference of the EACSL*, volume 12 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 5–19. Leibniz-Zentrum für Informatik., 2011.

[4] J. Adámek, S. Milius, R. Myers, and H. Urbat. Generalized Eilenberg Theorem I: Local Varieties of Languages. In A. Muscholl, editor, *Foundations of Software Science and Computation Structures*, volume 8412 of *LNCS*, pages 366–380, 2014.

[5] J. Adamek, S. Milius, R. Myers, and H. Urbat. Varieties of Languages in a Category. *ArXiv e-prints*, Jan. 2015.

[6] J. Almeida. Residually finite congruences and quasi-regular subsets in uniform algebras. *Portugaliae mathematica*, 46(3):313–328, 1989.

[7] J. Almeida. *Finite Semigroups and Universal Algebra*. Series in algebra. World Scientific, 1994.

[8] J. Almeida. Profinite semigroups and applications. In *Structural theory of Automata, Semigroups, and Universal Algebra*, pages 7–18, 2003.

[9] M. Arbib and E. Manes. Adjoint machines, state-behaviour machines, and duality. *Journal of Pure and Applied Algebra*, 6:313–344, 1975.

[10] M. Arbib and H. Zeiger. On the relevance of abstract algebra to control theory. *Automatica*, 5:589–606, 1969.

[11] A. Ballester-Bolinches, E. Cosme-Llópez, and R. Esteban-Romero. A description based on languages of the final non-deterministic automaton. *Theoretical Computer Science*, 536(0):1 – 20, 2014.

[12] A. Ballester-Bolinches, E. Cosme-Llópez, and R. Esteban-Romero. Group extensions and graphs. *Expositiones Mathematicae*, 2015. Accepted manuscript (unedited version) available online: 20th July 2015.

[13] A. Ballester-Bolinches, E. Cosme-Llópez, R. Esteban-Romero, and J. J. M. M. Rutten. Formations of monoids, congruences, and formal languages. CWI Technical Report FM-1504, CWI, June 2015. Submitted.

[14] A. Ballester-Bolinches, E. Cosme-Llópez, and J. Rutten. The dual equivalence of equations and coequations for automata. *Information and Computation*, 244:49 – 75, 2015.

[15] A. Ballester-Bolinches, J.-É. Pin, and X. Soler-Escrivà. Formations of finite monoids and formal languages: Eilenberg's theorem revisited. *Forum Math.*, 26(6):1731–1761, 2012.

[16] M. Barr. Terminal coalgebras in well-founded set theory. *Theoretical Comput. Sci.*, 114(2):299–315, 1993.

[17] C. Behle, A. Krebs, and S. Reifferscheid. Typed monoids: An eilenberg-like theorem for non regular languages. In *Proceedings of the 4th International Conference on Algebraic Informatics*, CAI'11, pages 97–114, Berlin, Heidelberg, 2011. Springer-Verlag.

[18] N. Bezhanishvili, C. Kupke, and P. Panangaden. Minimization via duality. In C.-H. L. Ong and R. J. G. B. de Queiroz, editors, *WoLLIC*, volume 7456 of *LNCS*, pages 191–205, 2012.

[19] G. Birkhoff. On the Structure of Abstract Algebras. *Mathematical Proceedings of the Cambridge Philosophical Society*, 31:433–454, 1935.

[20] F. Bonchi, M. Bonsangue, M. Boreale, J. Rutten, and A. Silva. A coalgebraic perspective on linear weighted automata. *Information and Computation*, 211:77–105, 2012.

[21] F. Bonchi, M. Bonsangue, H. Hansen, P. Panangaden, J. Rutten, and A. Silva. Algebra-coalgebra duality in Brzozowski's minimization algorithm. *ACM Transactions on Computational Logic*, 15(1), 2014.

[22] F. Bonchi, M. Bonsangue, J. Rutten, and A. Silva. Brzozowski's algorithm (co)algebraically. In R. Constable and A. Silva, editors, *Logic and Program Semantics.*, volume 7230 of *LNCS*, pages 12–23, 2012.

[23] F. Bonchi and D. Pous. Checking NFA equivalence with bisimulations up to congruence. In *Proc. POPL*, pages 457–468, 2013.

[24] M. M. Bonsangue, J. J. M. M. Rutten, and A. Silva. Coalgebraic logic and synthesis of Mealy machines. In R. M. Amadio, editor, *FoSSaCS08*, volume 4962 of *Lecture Notes in Computer Science*, pages 231–245. Springer, 2008.

[25] M. M. Bonsangue, J. J. M. M. Rutten, and A. Silva. An algebra for Kripke polynomial coalgebras. In A. Pitts, editor, *LICS 2009*, volume 5504 of *Lecture Notes in Computer Science*, pages 49–58. IEEE Computer Society, 2009.

[26] J. Brzozowski. Derivatives of regular expressions. *Journal of the ACM*, 11(4):481–494, 1964.

[27] J. Brzozowski and H. Tamm. Theory of Átomata. In G. Mauri and A. Leporati, editors, *Developments in Language Theory*, volume 6795 of *Lecture Notes in Computer Science*, pages 105–116. Springer Berlin Heidelberg, 2011.

[28] J. Conway. *Regular Algebra and Finite Machines*. Dover Books on Mathematics. Dover Publications, 2012.

[29] M. Dekkers. Stone duality. An application in the theory of formal languages. Master's thesis, Radboud Universiteit Nijmegen, the Netherlands, December 2008.

[30] M. Delgado, S. Linton, and J. J. Morais. *Automata (version 1.13)*, 2004. `http://cmup.fc.up.pt/cmup/mdelgado/automata/`.

[31] S. Eilenberg. *Automata, languages and machines (Vol. A)*. Pure and applied mathematics. Academic Press, 1974.

[32] S. Eilenberg. *Automata, languages and machines (Vol. B)*. Pure and applied mathematics. Academic Press, 1976.

[33] Z. Ésik and M. Ito. Temporal logic with cyclic counting and the degree of aperiodicity of finite automata. *Acta Cybern.*, 16(1):1–28, 2003.

[34] The GAP Group. *GAP – Groups, Algorithms, and Programming, Version 4.7.7*, 2015.

[35] W. Gaschütz and U. Lubeseder. Kennzeichnung gesättigter formationen. *Mathematische Zeitschrift*, 82(3):198–199, 1963.

[36] M. Gehrke. Stone duality and the recognisable languages over an algebra. In K. et al., editor, *Algebra and Coalgebra in Computer Science (CALCO 2009)*, volume 5728 of *LNCS*, pages 236–250, 2009.

[37] M. Gehrke. Duality and recognition. In Murlak and Sankowski, editors, *Mathematical Foundations of Computer Science*, volume 6907 of *LNCS*, pages 3–18, 2011.

[38] M. Gehrke, S. Grigorieff, and J.-É. Pin. Duality and equational theory of regular languages. In *Proceedings ICALP*, volume 5126 of *LNCS*, pages 246–257, 2008.

[39] G. Grätzer. *Universal Algebra*. Mathematics and Statistics. Springer, 2008.

[40] P. Grillet. *Semigroups: an introduction to the structure theory*. Marcel Dekker, Inc., New York, 1995.

[41] Y. Q. Guo, C. M. Reis, and G. Thierrin. Relatively f-disjunctive languages. *Semigroup Forum*, 37:289–299, 1988.

[42] M. Hennessy and R. Milner. On Observing Nondeterminism and Concurrency. In *Proceedings of the 7th Colloquium on Automata, Languages and Programming*, pages 299–309, London, UK, 1980. Springer-Verlag.

[43] M. Hennessy and R. Milner. Algebraic Laws for Nondeterminism and Concurrency. *J. ACM*, 32(1):137–161, Jan. 1985.

[44] R. Kalman. On the general theory of control systems. *IRE Transactions on Automatic Control*, 4(3):110–110, 1959.

[45] R. E. Kalman, P. L. Falb, and M. A. Arbib. *Topics in Mathematical Systems Theory.* McGraw Hill, 1969.

[46] S. Kleene. Representation of events in nerve nets and finite automata. In Shannon and McCarthy, editors, *Automata Studies*, pages 3–41. Princeton Univ. Press, 1956.

[47] S. P. Kogalovskiĭ. On Birkhoff's theorem. *Uspekhi Mat. Nauk*, 20(5):206–207, 1965.

[48] D. Kozen. Realization of Coinductive Types . *Electronic Notes in Theoretical Computer Science* , 276(0):237 – 246, 2011. Twenty-seventh Conference on the Mathematical Foundations of Programming Semantics (MFPS XXVII) .

[49] Y. Liu, K. P. Shum, and Y. Q. Guo. Relatively regular languages and thin codes. *European J. Combin.*, 29:261–267, 2008.

[50] S. Mac Lane. *Categories for the Working Mathematician*, volume 5 of *Graduate Texts in Mathematics.* Springer-Verlag, New York-Heidelberg-Berlin, second edition, 1998.

[51] E. Manes and M. Arbib. *Algebraic Approaches to Program Semantics.* The AKM Series in Theoretical Computer Science. Springer New York, 1986.

[52] H. Neumann. *Varieties of Groups.* Ergebnisse der Mathematik und ihrer Grenzgebiete. Springer-Verlag, 1967.

[53] J.-É. Pin. *Varieties of formal languages.* North Oxford, London and Plenum, New York, 1986. Translation of *Variétés de langages formels*, Masson, 1984.

[54] J.-É. Pin. A variety theorem without complementation. *Russian Mathematics (Izvestija vuzov. Matematika)*, 39:80–90, 1995.

[55] J.-É. Pin. *Mathematical Foundations of Automata Theory*, 2014.

[56] N. Pippenger. Regular languages and stone duality. *Theory of Computing Systems*, 30(2):121–134, 1997.

[57] M. O. Rabin and D. Scott. Finite Automata and Their Decision Problems. *IBM J. Res. Dev.*, 3(2):114–125, Apr. 1959.

[58] C. M. Reis and H. J. Shyr. Some properties of disjunctive languages on a free monoid. *Information and Control*, 37(3):334–344, 1978.

[59] J. Reiterman. The Birkhoff theorem for finite algebras. *Algebra universalis*, 14(1):1–10, 1982.

[60] J. Rhodes and B. Steinberg. *The q-theory of finite semigroups.* Springer Monographs in Mathematics. Springer, New York, 2009.

[61] F. Roumen. Canonical automata via duality. Master's thesis, Radboud Universiteit Nijmegen, the Netherlands, 2011.

[62] J. Rutten. Automata and coinduction (an exercise in coalgebra). In D. Sangiorgi and R. de Simone, editors, *Proceedings of CONCUR'98*, volume 1466 of *LNCS*, pages 194–218, 1998.

[63] J. Rutten. Universal coalgebra: a theory of systems. *Theoretical Computer Science*, 249(1):3–80, 2000. Fundamental Study.

[64] J. Rutten. Behavioural differential equations: a coinductive calculus of streams, automata, and power series. *Theoretical Computer Science*, 308(1–3):1–53, 2003.

[65] J. Rutten, A. Ballester-Bolinches, and E. Cosme-Llópez. Varieties and covarieties of languages (preliminary version). In D. Kozen and M. Mislove, editors, *Proceedings of MFPS XXIX*, volume 298 of *Electron. Notes Theor. Comput. Sci.*, pages 7–28, 2013.

[66] J. J. Rutten. Automata, Power Series, and Coinduction: Taking Input Derivatives Seriously (Extended Abstract). Technical report, CWI (Centrum Wiskunde & Informatica), Amsterdam, The Netherlands, 1999.

[67] J. Sakarovitch. *Elements of Automata Theory.* Cambridge University Press, 2009.

[68] J. Salamanca, A. Ballester-Bolinches, M. Bonsangue, E. Cosme-Llópez, and J. Rutten. Regular varieties of automata and coequations. In R. Hinze and J. Voigtländer, editors, *Mathematics of Program Construction*, volume 9129 of *Lecture Notes in Computer Science*, pages 224–237. Springer International Publishing, 2015.

[69] J. Salamanca, M. M. Bonsangue, and J. J. M. M. Rutten. Equations And Coequations For Weighted Automata. In *Proceedings of International Symposium on Mathematical Foundations of Computer Science 2015 (MFCS 15)*, Lecture Notes in Computer Science, pages 1 – 21. Springer, 2015.

[70] M. Schützenberger. On the definition of a family of automata. *Information and Control*, 4(2–3):245 – 270, 1961.

[71] M. Schützenberger. On finite monoids having only trivial subgroups. *Information and Control*, 8(2):190–194, 1965.

[72] L. A. Shemetkov and A. N. Skiba. *Formations of algebraic systems.* Modern Algebra, Moscow, 1989. In Russian, with an English summary.

[73] A. Silva. *Kleene coalgebra.* PhD thesis, Radboud Universiteit Nijmegen, Dec. 2010.

[74] A. Silva, M. M. Bonsangue, and J. J. M. M. Rutten. Non-deterministic Kleene coalgebra. *Log. Meth. Comput. Sci.*, 6(3):23/1–39, 2010.

[75] I. Simon. Piecewise Testable Events. In *Proceedings of the 2Nd GI Conference on Automata Theory and Formal Languages*, pages 214–222, London, UK, 1975. Springer-Verlag.

[76] C. Stirling. Bisimulation and logic. In D. Sangiorgi and J. Rutten, editors, *Advanced Topics in Bisimulation and Coinduction*, pages 173–196. Cambridge University Press, 2012.

[77] M. H. Stone. Applications of the theory of Boolean rings to general topology. *Trans. Amer. Math. Soc.*, 44:375–481, 1937.

[78] M. H. Stone. The representation of Boolean algebras. *Bull. Amer. Math. Soc.*, 44:807–816, 12 1938.

[79] H. Straubing. On logical descriptions of regular languages. In S. Rajsbaum, editor, *LATIN 2002: Theoretical Informatics*, volume 2286 of *Lecture Notes in Computer Science*, pages 528–538. Springer Berlin Heidelberg, 2002.

[80] J. Worrell. On the final sequence of a finitary set functor. *Theoret. Comput. Sci.*, 338:184–199, 2005.

[81] D. Zhang, Y. Guo, and K. P. Shum. On some decompositions of r-disjunctive languages. *Bull. Malays. Math. Sci. Soc.*, 2(3):727–746, 2014.

# Index

Faré que tota sa meva vida
sigui un viatge com aquest,
en de dia trec sa barca,
en sa nit arròs de peix.
No puc demanar res més;
he resolt tots els problemes
dient “tanmateix”.

En s’estiu — Antònia Font