A STUDY OF MULTISET ORDERING FOR CONSTRUCTING ALTERNATIVE P SYSTEMS AND THEIR APPLICATIONS

BY

Chinedu Matthew PETER
B.Sc (A.B.U, 2004), M.Sc (A.B.U, 2011)

PhD/SCI/22765/2012-2013

A THESIS SUBMITTED TO THE SCHOOL OF POSTGRADUATE STUDIES, AHMADU BELLO UNIVERSITY, ZARIA, IN PARTIAL FULFILMENT OF THE REQUIREMENTS FOR THE AWARD OF A DOCTORATE DEGREE IN MATHEMATICS

DEPARTMENT OF MATHEMATICS,
FACULTY OF PHYSICAL SCIENCES,
AHMADU BELLO UNIVERSITY,
ZARIA, NIGERIA

JANUARY, 2018

DECLARATION

I declare that the work in this Thesis entitled "A STUDY OF MULTISET ORDERING FOR CONSTRUCTING ALTERNATIVE P SYSTEMS AND THEIR APPLICATIONS" has been carried out by me in the Department of Mathematics. The information derived from the literature has been duly acknowledged in the text and a list of references provided. No part of this Thesis was previously presented for another degree or diploma at this or any other Institution.

Chinedu Matthew Peter

_____	_____	_____
Name of Student	Signature	Date

CERTIFICATION

This Thesis entitled **"A Study of Multiset Ordering for Constructing Alternative P Systems and their Applications"** By Chinedu Matthew PETER (PhD/SCI/22765/2012-2013) meets the regulations governing the award of the degree of Doctor of Philosophy of the Ahmadu Bello University, and is approved for its contribution to knowledge and literary presentation.

_____ _____
Professor D. Singh Date
Chairman, Supervisory Committee

_____ _____
Dr. A. M. Ibrahim Date
Member, Supervisory Committee

_____ _____
Professor S. B. Junaidu Date
Member, Supervisory Committee

_____ _____
Professor M. S. Audu Date
External Examiner

_____ _____
Professor A. A. Tijjani Date
Head of Department

_____ _____
Professor S. Z. Abubakar Date
Dean, School of Postgraduate Studies

DEDICATION

This thesis is dedicated to my beloved mother, Mrs. Fidelia Okwuchukwu Ogbuehi, of sweet memory.

ACKNOWLEDGMENT

I give glory to Almighty God for the life, good health and strength He has been showering on me and my loved ones and for His grace throughout the period during which this research was being carried out.

I appreciate my supervisors Prof. Dasharath Singh, Dr. A. M. Ibrahim and Prof. Sahalu B. Junaidu for their time and efforts in ensuring that this thesis is at its best, and for the knowledge I gained from them during my undergraduate and master levels. I will also not forget the P. G. coordinator, Dr. H. M. Jibril and the Assistant P. G. coordinator, Mr. M. S. Shagari.

I will not forget the efforts of my colleagues Prof. Johnson O. Fatokun, Dr. Yusuf Omeiza, Dr. (Mrs.) Obumnadike, Dr (Mrs.) Olanrewaju, Mr. Richard Tyokyaa, Mr. Abdullazeez Shehu, Mr. Balami H. Misrali, Dr. Usman Bello, Mrs. Rosemary Dima and other colleagues of the Department of Mathematical Science and IT, Federal University, Dutsinma.

My appreciation also goes to Mr. Alisi Ikechukwu (Applied Chemistry Depertment, FUDMA), Mrs. Bilkisu O. Abdulrahman (Biochemistry Department, FUDMA), Prof. Sunday E. Atawodi (Biochemistry Department, A.B.U, Zaria) through whom I gained immensely during interactive sessions with each of them and to my course mates Dr. Joel Taura and Dr. (Mrs.) Funmilola Balogun for their motivations.

I will ever appreciate the love, care, prayers, financial help and moral support of my Dad Evang. Peter N. Ogbuehi, my Mum Mrs. Fidelia O. Ogbuehi, my siblings Mr.

Ugochukwu Peter, Mrs. Ijeoma Peter, Mrs. Amaka Chukwudi, Bro. Okwuchukwu Peter, Bro. Chekwube Peter, Mrs. Ngozi Peter and Miss Obianuju Peter. I wish also to appreciate my wife Mrs. Callista Ebere Peter for her love and care and also for her efforts in reviewing some of the papers.

Finally, to the individuals whose names I may not be able to mention and who have contributed directly or indirectly towards the success of this thesis, I say thank you and may God reward you richly.

ABSTRACT

In this thesis, basic ingredients of partially ordered multisets were employed in which some of its formalisms were further developed and a set of rules formulated which explicitly characterize the chemical and physical functions, among others, of the processes taking place in a biological cell. In the sequel, a grid form of the Jouannaud-Lescanne set-based and submultiset-based multiset orderings have been developed. Relatively, more applicable definitions of the multiset orderings and some of their systematic relationships were presented. A possibility of extending the set-based multiset ordering to simple multisets of incomparable objects has also been observed. A consequence of partial ordering – a partial ordering which is restricted to incomparable element – was studied and a number of its examples presented. By using this concept, a variant of *alternative P system* using *specialized* rules was developed, and was shown to adequately simulate the activities of the biological cells as opposed to the conventional P systems whose rules represent the activities of the biological cells in an unspecialized manner. It is shown that with only one membrane and three rules, a *nondeterministic* alternative P system based on ideal ordering is able to characterize the family of recursively enumerable languages. Finally, some *deterministic* P systems based on weak rule priority were defined to carry out arithmetic operations and exemplified.

TABLE OF CONTENTS

Title Page ---------- i

Declaration ---------- ii

Certification ---------- iii

Dedication ---------- iv

Acknowledgment ---------- v

Abstract ---------- vii

Table of Contents ---------- viii

List of Figures ---------- xiii

List of Tables ---------- xiv

CHAPTER ONE: GENERAL INTRODUCTION----------1

1.0 Introduction ---------- 1

1.1 Statement of the Problem ---------- 8

1.2 Justification ---------- 8

1.3 Aim and Objectives ---------- 9

1.4 Methodology ---------- 10

1.5 Organization ---------- 10

CHAPTER TWO: LITERATURE REVIEW----------11

CHAPTER THREE: FUNDAMENTALS OF MULTISET, ORDER RELATION FORMAL LANGUAGE THEORY AND P SYSTEM ----------21

3.0 Introduction ----------21

3.1 Preliminaries of Multiset ---------- 22

3.2 Some Properties Holding for Multiset Operations ----------24

3.3 Some Basic Terminologies on Order Relations ---------- 26

3.4 Elementary Formal Language Prerequisites ---------- 32

3.4.1 Strings and languages ---------- 32

3.4.2 Operations with strings and languages ----------33

3.5 Chomsky Grammars ----------36

3.6 Some Elements of Computability ----------38

3.6.1 Automata ---------- 38

3.6.2 Register machine ---------- 40

3.6.3 Turing machine ----------42

3.7 Membrane Systems ----------45

CHAPTER FOUR: PARTIALLY ORDERED MULTISETS FOR THE PROPOSED ALTERNATIVE P SYSTEM ---------- 59

4.0 Introduction----------59

4.1 The Concept of Set-based Grid of a Partially Ordered Multiset----------60

4.1.1 Set-based difference grid----------65

4.1.2 Grid approach to Jouannaud-Lescanne's set-based multiset ordering ----------69

4.2 Redefined Submultiset-based Multiset Ordering via Grid ----------73

4.2.1 The concept of submultiset-based grid of a partially ordered multiset ----------74

4.2.2 Submultiset-based difference grid ---------- 79

4.2.3 Grid approach to the Jouannaud-Lescanne's submultiset-based multiset ordering ----81

4.3 Some Results Holding for Multiset Ordering ----------84

4.4 Some Extensions of Set-based Ordering to Simple Multisets of Incomparable Objects----------90

4.5 The Concept of 'Ideal' Ordering ----------96

4.5.1 Ideal ordering (a consequence of the properties of partial ordering) ----------96

4.5.2 The structure of an isosceles triple ---------- 98

4.5.3 Examples of ideal ordering/set ---------- 101

4.5.4 The structure of an ideal set ---------- 105

CHAPTER FIVE: AN APPLICATION OF IDEAL ORDERING TO ALTERNATIVE P SYSTEMS ---------- 107

5.0 Introduction ---------- 107

5.1 A Biochemical Interpretation of the Rules ---------- 109

5.1.1 Mixture ----------109

5.1.2 Null reaction ----------110

5.1.3 Reversible chemical reaction ----------111

5.1.4 Catalytic reaction ---------- 111

5.1.5 Physical change ---------- 111

5.1.6 Incomparability with respect to reaction ----------112

5.1.7 An approach for non-determinism----------112

5.1.8 Derived reactions ---------- 112

5.2 Turing Computability ----------114

5.3 A Nondeterministic Alternative P System with Specialization Rules Based on Ideal Ordering ----------115

5.4 The Computational Power of Cooperative Alternative P System with Specialization Rules in One Membrane based on Ideal Ordering ---------- 117

5.5 Some Applications of P Systems to Arithmetic Operations on Non-negative Integers – a Case of Determinism Based on Weak Rule Priority ----------119

5.5.1 Weak versus strong rule priority relations ----------119

5.5.2 Addition P system ---------- 121

5.5.3 Subtraction P system ---------- 122

5.5.4 Multiplication P system ----------123

5.5.5 Multiplication P system exemplified ----------125

5.5.6 Division P system ---------- 129

5.5.7 Division P system exemplified ----------132

CHAPTER SIX: SUMMARY CONCLUSION AND RECOMMENDATIONS---------138

6.0 Summary ----------138

6.1 Conclusion----------139

6.2 Recommendations for future research ----------140

INDEX OF RESULTS OBTAINED ----------142

REFERENCES ----------144

LIST OF FIGURES

Figure 1: Finite State Automaton ---------- 39

Figure 2: A Membrane Structure ----------47

Figure 3: Application of an Evolution rule $r : u \to v$ According to some Priority Relations ---------- 54

Figure 4: Diagrammatic Representation of a Multiset *M* for Set-based Multiset Ordering ----------61

Figure 5: Diagrammatic Representation of a Multiset *M* for Submultiset-based Multiset Ordering ----------74

Figure 6: The Structure of the Isosceles Triple of A (with a Peak) ----------99

Figure 7: The Structure of the Isosceles Triple of X (with a Trough) ----------99

Figure 8: Venn Diagrams Illustrating Set Examples of Ideal Ordering with a Trough ------ 101

Figure 9: Venn Diagrams Illustrating Set Examples of Ideal Ordering with a Peak ---------101

Figure 10: The Structure of a Typical Ideal Set ---------- 106

Figure 11: The Configurations of Addition P System ----------121

Figure 12: The Configurations of Subtraction P System ----------122

Figure 13: Initial Configuration of a Multiplication P System ----------124

Figure 14: Initial Configuration of a Division P System ----------130

LIST OF TABLES

Table 1: The Difference Grid of $\{5,4,2,2,1, a, a, a, a, b, b\}$ and $\{6,3,3, a, a, a, b, b, c\}$ ---------66

Table 2: The Grid of $\{5,4,2,2,1, a, a, a, a, b, b\}$ ----------66

Table 3: The Grid of $\{6,3,3, a, a, a, b, b, c\}$ ---------- 66

Table 4: The Difference Grid of $\{a, a, e\}$ and $\{a, a, e, e\}$ ---------- 91

Table 5: The Supposed Set-based Difference Grid of $\{e, e\}$ and $\{a\}$ ----------91

Table 6: The difference grid of $\{a, a, e\}$ and $\{a, a, e, e\}$---------- 92

CHAPTER ONE

GENERAL INTRODUCTION

1.0 Introduction

P system is a computational device which is designed to model biologically inspired processes. It abstracts the way in which chemicals interact or cells behave. The device was first submitted as a report in Păun (1998) and subsequently published in Păun (2000), and hence the name. Although inspired by biology, the primary research interest in P systems is concerned with their use as a computational model rather than for biological modelling (Păun & Rozenberg, 2002) although this is also being investigated (Păun, 2006, Ardelean and Cavaliere, 2003).

It is worth mentioning some of the basic features of models investigated in P system. The compartments of a cell contains molecules and other substances swimming in an aqueous solution. Thus, the suggestion is to work with objects whose multiplicities matter, hence with multisets. This is a data structure with peculiar characteristics, not new but not exhaustively investigated in Mathematics and applied in Computer Science.

Also of interest in our research is the notion of multiset ordering. We redefine the Jouannaud-Lascanne's mulitset ordering based on the concept of grid. To this end, we define the concepts of grid, difference grid and grid-based orderings. We extend

the set-based multiset ordering to accommodate simple multisets of incomparable objects which are a systematic way of proving that the grid-based ordering is stronger than the conventional set-based ordering. Also, we consider the concept of *ideal ordering* – a partial ordering whose property is restricted to the partially ordered relation and the accompanying incomparability relation on three elements. The main idea behind the use of ordering in the model of P system is to introduce the *rule* representing the so-called *derived reaction*. This rule together with those representing physical change, chemical reaction, catalytic reaction and mixture are responsible for the evolution of the variant of P system introduced.

We apply the concept of P system to arithmetic operations on non-negative integers. The P systems obtained are of deterministic type based on the weak priorities for rule application. It is verified that two membranes suffice. Moreover, there are at most four objects for multiplication and five objects for division throughout the computation processes. The P systems so obtained serve as bases upon which other complex operations may be built, and upon which basic operations may be extended to cover the set of positive and negative real numbers.

It may be noted that many other natural computing devices have been in existence, namely, neural networks, (Bishop, 1995), genetic algorithm, (Davis, 1991) and DNA computing (Păun *et al.*, 1998). None of these areas considers the cell itself as its object of research, and pays attention to the structural properties of membranes and compartmentalization. Moreover, nature computes not only at the

neural or genetic or molecular level, but also at the cellular level (Păun, 2006). This is what membrane computing is concerned about. Since the primary research interest in P systems is concerned with their use as a computational model the area of study can be seen as an extension of DNA computing (or more generally, molecular computing) from the one processor level to a distributed computing model.

P system is one of the most widely researched areas in the line of emerging computing devices inspired by natural biological phenomena. Essentially, the study of the cell cannot be understood without considering membranes. As a matter of fact, parts of a biological system are delimited by membranes. It begins with the membranes which enclose the internal organs of the cell – the nucleus, the Golgi apparatus, etc, and the outermost membrane covering a cell, also called the *skin membrane.* It goes a long way to the skin of an individual organism which separates the organism from the environment. There is also the concept of a virtual membrane which delimits, for example, a part of an *ecosystem*. This gives the flavor of distributed computing to this model. There are different but well delimited computing units that coexist and are hierarchically arranged in a complex system, from a single processor level to the *World Wide Web*. Such distributed symbol processing systems are mirrored in mathematical terms by exploiting a grammar system and multiset and its associated algebra, automata systems and register machines.

Since the biological membranes are known to be semi-permeable, membranes prevent certain chemicals and allow other chemicals to pass through it in a selective

manner. Sometimes the movement is only in one direction, and sometimes it is bi-directional. This can be a simple selection by size, in the case of small molecules, or a much more non-trivial selection, through protein channels, which not only select, but can also pass molecules from a region of low concentration to a region of higher concentration and vice versa, probably coupling molecules as they move, in a process called symport or antiport. Of course, variants could be developed in which the system has so-called opaque membranes, opaque in the sense that no movement across such membranes is allowed. It may be natural that this variant is rarely studied. An important role of membranes in membrane computing is that each membrane is seen as a separator of two regions, the region inside the membrane and the region outside the membrane.

A basic ingredient of membrane computing is the notion of *membrane structure*. It is a hierarchical architecture composed of non-intersecting membranes placed inside a unique membrane called the skin membrane. Non-intersecting in the sense that the arrangement of two membranes is either that one contains the other or they are disjoint. Such architecture is usually represented by a Venn diagram for simplicity. However, formal definition of a membrane structure exists.

Next, we consider the notion of a super-cell system. A super-cell system is a membrane structure with objects placed in the regions delimited by the membranes. Several copies of a particular object can appear. This gives rise to the use of multisets, which simply are sets whose elements can have multiplicities associated with them.

A P system is a super-cell which has the ability to evolve. In other words, a P system is a membrane structure with objects in some of the regions specified by the membranes, with specified evolution rules for the regions and input and output membranes. The rules in these regions are responsible for the evolution of the objects. The P system pre-specifies input and output membranes. The input and output membranes contain respectively, the input objects before a computation and the output objects after the computation. Any object, alone or together with one more object, can evolve – be transformed into other objects, be passed from one region to the other through a membrane, and can dissolve the membrane into which it is placed. All objects can evolve at the same time, in parallel. In other words if a rule is applied to a particular object, it should be applied as much as possible to the copies of the object in the membrane containing such rule. Moreover, all membranes are active at the same time. That is, all the membranes in the system work in parallel. It can be therefore seen that there exists two levels of parallelism in membrane computing.

A computation in a membrane system is obtained by starting with an initial state called *initial configuration*. Objects are placed in the input membrane. Such objects define the input for the computation. Rules are then applied in a maximally parallel manner to the objects placed inside the regions delimited by the membranes, non-deterministically choosing the rules. This is done in parallel for all the membranes whose rules can be applied at that moment. A computation halts when no

rules can further be applied in the system. The output of the computation is obtained by the number and *multiplicity* of the objects inside a predefined output membrane.

The rules which make the objects in these membranes to evolve may be given priorities as to which to be applied first and which to be applied next and so on. Also rules which have no priorities assigned to them are chosen non-deterministically. The system is said to be *cooperative* if there are rules which specify the evolution of several objects at the same time and non-cooperative if all the rules specify that only one object evolve at the same time. An intermediate case is where there are certain objects called catalysts which do not evolve alone, but help in the evolution of other objects, and are not modified by the rules. We call the systems of the type so far explained *transition P systems* as distinguished from other variants which are also considered in various texts.

It is worth mentioning that the cooperative P systems and systems with catalysts have computational completeness. That is, they can characterize the recursively enumerable sets of natural numbers. It is with this notion that the super-cell structure is used as a support for a computing device based on any type of objects and any type of evolution rules associated with them. In the above case, the objects are single symbols, taken from a set called an alphabet. Also considered is a case where the objects are strings. In this case an infinite set of objects can be used, which can evolve in many ways. Such ways are defined by rules which process strings. Rules such as *insertion*, *deletion*, *rewriting*, *splicing* and *point mutation*, just to

mention a few, have been used. Splicing is a technical term in molecular biology and genetics meaning the operation of modifying the nascent pre-messenger RNA transcript in which *introns* are removed and *exons* are joined.

It is worth mentioning that a P system is a computing device with input and output relation and pertains to natural computing. One is interested in an output given a particular input. P systems are universal in the sense that their computational ability can be compared to that of a Turing machine. Hence, we say that they are computationally complete. Interesting researches in membrane computing are already ongoing most of which lay emphases on the computational completeness, power and efficiency of its variants.

An element which is pre-requisite for researches in membrane computing is the theory of formal language. In fact, Kitano (2002) views membrane computing as a branch of formal language theory. It is worth mentioning that the proof techniques, the generative style, the rewriting-type of multiset processing rules are very much similar to formal language theory. As a result, most papers on P systems start by collecting a long list of notations and results from formal language theory. Basic regulations on the *context-free grammars*, such as *matrix grammars* (with appearance checking), are not only used, but are promisingly very powerful and raise new and interesting research questions. The use of matrix grammars is rather useful in many cases, because they involve contest-free grammars, which is the most widely investigated and applied class of Chomsky grammars and they have beautiful normal

forms and more importantly, they are very powerful results, proved decades ago. See Rozenberg and Salomaa (1997), Martin-Vide and Păun (2001) and Kitano (2002) for detailed reading. In particular, Kitano (2002) is handy for interested authors of membrane computing.

1.1 Statement of the Problem

Quasi-general P systems have been modelled strictly by means of rules which represent the processes taking place in a biological cell without explicitly specifying the biological, chemical or physical function of each rule as is obtainable in the said cell. It can therefore be said that the P systems have not been able to adequately simulate the activities of the biological cell, and thus cannot be said to truly carry out the computational capabilities of the cells in question. Bearing in mind such shortcomings, we intend to model an alternative P system, which is capable of simulating biological activities at cellular level and can compute all recursively enumerable sets of natural numbers with limited number of rules, each of which would function in a specialized manner.

1.2 Justification

P system is one of the most researched areas in the field of natural computing. However, the known variants of P systems basically make use of the application of rules to objects within regions and from one region to another without accounting for the biological functions (reactions) of each of the rules used. The variant of P system in this thesis considers physical change, chemical reaction, mixtures and derived

reaction among others. The variant of P system is therefore more realistic. Moreover, it uses the mathematical tool of multiset ordering, thereby rendering the variant of P system more elegant in its application. It uses limited number of rules and only one membrane to achieve computational completeness.

1.3 Aim and Objectives

The aim of this research is to develop an alternative P system based on an *ideal* consequence of partial ordering on multisets which would be capable of modelling biological systems.

The main objectives of this research are to:

i. Construct a grid-based multiset orderings and compare them with their existing partition-based counterparts.
ii. Investigate the possibility of extending the set-based multiset ordering to simple multisets of incomparable objects.
iii. Examine the concept of *ideal* ordering – a property restricted partial ordering on multisets, with examples.
iv. Construct a nondeterministic alternative P system.
v. Investigate the computational completeness of the proposed alternative P systems.
vi. Design algorithms to compute the four basic arithmetic operations in deterministic P systems based on the weak rule priority.

1.4 Methodology

A grid approach was used to redefine the set-based and submultiset-based orderings which was originally defined by Jouannaud and Lascanne. Also, we used the induction method of proof to show that the set-based ordering can be extended to simple multisets of incomparable objects. A multiset ordering approach was followed in conjunction with physical change, chemical reaction, derived reaction, mixture and the null reaction to model a process taking place in a living organism. Moreover, a register machine was used to prove that the model of P system presented is computationally complete. For the application of P system to arithmetic operations on non-negative integers, we used the method of weak rule priority to achieve determinism for the system.

1.5 Organization

This thesis is subdivided into six chapters. Chapter one gives a general introduction to the study. In Chapter Two, a comprehensive review of the literature of P systems and multiset orderings relevant to the research are presented. In Chapter Three, some of the fundamentals of P system and multiset orderings were observed. In particular, the concept of ideal ordering is introduced and some proposed results are investigated. In Chapter Four, a deterministic alternative P system based on the concept of ideal ordering is presented. In Chapter Five, we present the application of P systems in modelling arithmetic operations based on the weak rule priority. Chapter six contains conclusions and summary.

CHAPTER TWO

LITERATURE REVIEW

Păun (1998) introduces a branch of natural computing which tries to imitate nature in the way it computes, by learning new computing models and computing paradigms experimented for billions of years by nature and by implementing them in computations. The system considers the notion of membranes in a network called membrane structure, usually presented in the form of a Venn diagram. Objects are placed in the regions delimited by the membranes; these objects are allowed to evolve by the evolution rules placed within the regions. The overall system gives rise to a device which works similar to a computing machine called a P system.

Krishna and Rama (2001a) showed that evolution rules applied to sequential rewriting could be extended to the parallel case as an improvement to Păun (1998). The aim of the extension is quite natural and more appropriate from the point of view of biology, because a cellular substance can be the subject of many chemical reactions acting on it at the same time. A result of this extension is that if the rules that are applied on the same string have mixed target indications, then consistency problems for the communication of the evolved string occur, because there are contradictory indications about the region where the string should be at the next step. The approach used in this chapter to face the problem is to define deadlock states, which cause further processing and communication of the strings to stop. Parallel rewriting P

systems which use some different parallel rewriting methods, and whose configurations may present deadlock states or not, are compared to Lindenmayer system and matrix grammars without appearance checking as well as to parallel rewriting P systems which use a different type of parallelism.

Păun *et al.* (2002) dealt with P systems which use different parallelism methods for strings rewriting, but the assumption here is that no target conflicts can arise after a parallel application of rules. Indeed, the rules to rewrite a certain string are chosen according to one of several different parallel rewriting methods, but only the rules which agree on the target membrane can be applied on a common string. It analyzes the computational power of such kind of P systems, comparing them to Lindenmayer systems and obtained a final hierarchy to the chosen parallel rewriting method.

A somewhat alternative P system was introduced in Freund and Oswald (2002) called PX systems. PX system is a new model of membrane systems where the application of the rules is controlled by an underlying state-transition diagram. In particular, such a variant of membrane systems is obtained by combining the basic features of P systems with the features of X machines. PX systems are proved not to be computationally complete by showing that they are equivalent in power to a particular class of L systems. It also proposes an alternative approach to generate languages by means of P systems: building up an adequate representation for a string by means of an appropriate membrane structure and then generating the string by

visiting the membrane structure according to a well-specified strategy. To this aim, it considers P systems where the membrane structure can be modified during a computation by using the operation of membrane creation, membrane division, membrane duplication, and membrane dissolution. The output of a computation may be obtained either by visiting the tree associated with the membrane structure or by following the traces of a specific object, called traveler or sending out the objects. For each of these approaches, characterizations of recursively enumerable languages are provided that are based on P systems that use different sets of operations for modifying the membrane structure. Also, Freund and Oswald (2002) reports several results concerning the computational power of the various features of population P systems.

Furthermore, Freund and Oswald (2002) deals with the issues concerning the use of membrane systems as a modelling tool for biological systems. Specifically, it presents a general framework for modelling with membrane systems that is based on a model where rules have associated a finite set of attributes and a corresponding function. Attributes and functions are meant to provide those extra features which are necessary to close the gap between the abstractness of more standard P system models and the reality of the phenomenon to be modelled. The behavior of such P systems is defined in terms of bounded parallelism, which precisely formalizes the idea of a membrane system as a system where a certain number of components evolve in parallel at the same time by means of a certain number of rules applied inside each

one of these components. It also deals with a particular instance of the general model, a particular class of P systems is considered where rules have associated a real constant as an attribute and the corresponding function is used to compute a value of probability depending on this constant and on certain multisets of objects defining the context where the rule is applied. For these P systems, a precise strategy for applying the rules is devised: in each step, the next rule to be applied is selected depending on the aforementioned values of probabilities.

The evolution-communication model was analyzed in Păun (2002) and the obtained results are listed. Both the generating and accepting cases are investigated, as well as a few restrictions, such as determinism and time freeness. A special variant is introduced, called proton pumping P systems, and the corresponding results are also presented. The same Păun (2002) introduce P systems having only symport/antiport objects between regions. The history of the research latest results is presented, both on cell-like P systems and tissue P systems. It also investigates what can be computed by symport/antiport P systems with a small alphabet of objects. Again, both cell-like P systems and tissue P systems are considered. In its chapter 7, the same text dwells upon some applications of P systems especially of evolution-communication P systems in the Sorting Theory. Application of symport/antiport P systems with membrane division in Complexity Theory and an application of proton pumping P systems in multiset rewriting theory are also presented.

The question of how errors spread and propagate in cooperative systems has been studied in a variety of fields. Chapter 4 of Păun *et al.* (2001) is devoted to several bio-inspired generalizations of the classical P system model and is motivated by the issue of resilience in face of errors. It presented systems that are immune to errors regarding the execution times of rules and the degree of the rewriting parallelism. It introduced and studied timed P systems with promoters/inhibitors where a time is associated with the execution/application of any rule of the system in any given configuration. It then find a proper subclass of such systems able to generate the family of sets of vectors of numbers regardless of the values of the time associated to the rules.

In many applications, P systems have considerable advantages. Among them are their inherently discrete nature, parallelism, transparency, scalability non-determinism and expressivity to model phenomena. On the application of P system, Păun *et al.* (2001) studies two applications of membrane computing, namely static sorting and circuits simulations. Sorting numbers represents one of the most studied problems in computer science. Boolean circuits embody their notion of massively parallel signal processing and are usually encountered in many parallel algorithms. Many important problems such as sorting, integer arithmetic and matrix multiplication are known to be computable by small size Boolean circuits much faster than ordinary sequential digital computers. Păun *et al.* (2001) showed that membrane computing allows one to simulate such Boolean circuits in a polynomial time.

In Păun and Păun (2005), we saw the first results related to P systems using rules from brane calculi and evolution rules based only on proteins placed on membranes. It also considered a class of P systems with objects placed in both compartments and on membranes, with the evolution of the former being controlled by the latter, which can also evolve, either by membrane operations or, directly by object evolution rules. Computational universality is obtained for several combinations of such possibilities. It also investigates P systems with only one protein on a membrane, P systems with minimal number of rules, P systems working in accepting and computing mode, etc. As an application of P systems with proteins on membranes, Păun and Păun (2005) gave a solution to the Satisfiability problem in polynomial time, using exponential space. The thesis Păun and Păun (2005), in its Chapter Six, combined two central ideas used in membrane computing. Namely, working with string objects processed by context-free rewriting rules (such systems are known not to be universal (Păun *et al.*, 2005)) without target indications associated with the rules, and moving the strings across membranes by means of symport rules. The passage of strings across membranes is controlled by means of regular expressions which define language of strings able to move in/out a region of a system. This reminiscent of EC (evolution-communication) P systems from Ibarra (2005) and Martin-Vide and Păun (2000). can also be related to the brane calculi – membrane computing bridging, in the sense that one combines the evolution of objects (described by strings this time) from the compartments of a membrane structure with the control imposed by the proteins placed on membranes.

We list here some of the variants of P system already developed. In Păun (2001), new approximate algorithm for optimization problems, called membrane algorithm were proposed. In Nishida (2004), Tissue P systems with symport/antiport rules of one symbol are computationally complete. In Ciobanu *et al.* (2003), Cient-server P system were proved to have the same expressive power as Turing machines.

Krishna and Păun (2005) introduced the notion of mobile membranes, to the aim of defining a new type of communication which has as biological counterpart the functioning of the Golgi apparatus in living cells. In order to simulate the inspiring natural features, it considers P systems with simple membrane structures, with string objects representing proteins or any structured substance, and with operations on strings of a biochemical nature only. It analyzes the computational power and the efficiency of this kind of systems. It also showed that all recursively enumerable languages can be generated with a finite number of membranes. In either case, communication with mobile membranes is associated or not to classical communications. It also proved the possibility of solving NP-complete problems in polynomial time, and gave a direct construction for the Hamiltonian Path Problem.

Kelemen *et al.* (2004) showed that the Hamiltonian Path Problem could be solved in polynomial time by P systems with active membranes where the membranes are only divided into two new membranes. It also showed how the Hamiltonian path problem could be solved in polynomial time by a variant of P systems, with the possibility of dividing non-elementary membranes under the influence of objects

present in them. That the computational completeness of such systems does not necessarily depend on membrane division or membrane dissolution has also been shown. Other texts dealing with P systems with active membranes are Zandron *et al.* (2001), Alhazov and Pan (2004), Alhazov *et al*. (2004) and Păun (2004). In Păun (1999) , P automata was introduced and it was shown that any recursively enumerable language could be represented as an image of the set of accepted input multiset sequences of a P automaton, under a certain projection. Freund *et al.* (2003) and Freund (2004) are some of the materials on P automata.

Though membrane computing started as endeavor to solve some problems in computer science, and not as a model of the cell biology, it turned out to be a potential tool for simulating not only local phenomena in the cell but possibly also simulating the cell as a whole (Tomita, 2001).

So far from the biological point of view, P systems have seen applications to respiration in bacteria Suzuki *et al.* (2003), photosynthesis (Nishida, 2002), immune system Ardelean and Cavaliere (2003) and the application of multiset re-writing P systems to artificial life (Suzuki and Tanaka, 2000).

Manca *et al*. (2001) studied a variant of P system recently introduced by Krishna and Rama (2001b), which has the power to also replicate the strings when rewriting them. Freund (1999) and Freund (2001) considered sequential variants of P systems. He considered the notion of generalized P systems, which is the most

general model of sequential P systems. It allows for the simulation of graph controlled grammars of arbitrary type based on productions working on single objects. Freund also investigated the power of deterministic P systems described in Freund and Păun (2003), in which an input is provided (by the user of the system, placed in the environment) in the form of a number and the system accepts this number or not, provided that a halting computation exists starting from the initial configuration. It examines the possibility to recognize all Turing computable sets of numbers by means of deterministic P systems. In Păun (2006), Ionescu *et al.* (2006) and Cavaliere *et al.* (2009), a consideration of the use of spikes in P systems was dealt with, which mimics the way that neurons communicate with each other. In neural P systems with a non-synchronized use of rules, a neuron can apply or not apply its rules which are enabled by the number of spikes it contains. Spikes can be introduced. The result of the computation is the number of spikes sent out by the system. The power of the system is not decreased by the non-determinism introduced in the function of the system. Also, Turing machine equivalence is attained. On P systems with polarizations, Alhazov and Pan (2004) and Alhazov *et al.* (2004) addressed the problem of removing the polarizations of membranes from P systems with active membranes. This is achieved by changing the labels of the membranes by means of communication rules or by rules for membrane division. P systems with active objects are also considered in some variants – active in the sense that some objects create membranes. This new variant of P system is capable of solving the Hamiltonian Path Problem in linear time. This variant can be seen in Mutyam and Kriithivasan (2001).

A variant of P system called splicing has also been investigated in the literature. Some splicing also includes cutting and recombination of string objects. Splicing P system tends to mimic the modified version of nascent pre-messenger RNA transcript in which introns are removed and exons are joined. Variants of splicing P systems can be found in Păun and Păun (2001), Frisco (2000) and Freund (1999).

Some ramifications of multiset orderings exist in the literature. Knut's dominance (Knuth, 1973) seems to be the earliest known to usher in multiset ordering. However, Dershowitz and Manna (1979) is the earliest reference known for introducing the Dershowitz-Manna ordering and using it to prove the termination of programs and term rewriting systems. The Dershowitz-Manna ordering was also used in Singh and Peter (2011) to define *tree structure* for membrane computing. Huet and Oppen gave a definition of multiset ordering in Huet and Oppen (1980),– a refinement of the Dershowitz-Manna ordering. Jouannaud and Lescanne (1982) introduced some multiset orderings based on multiset partition. The multiset orderings proved to be stronger than the Dershowitz-Manna ordering.

CHAPTER THREE

FUNDAMENTALS OF MULTISET, ORDER RELATION, FORMAL LANGUAGE THEORY AND P SYSTEM

3.0 Introduction

It can be recalled that the main aim of the research is to develop an alternative P system based on multiset ordering which would be capable of modelling biological processes – our primary research being the use of the alternative P system as a computational device rather than for biological modelling. Recall also, that our methodology involves a multiset ordering approach which will be followed to model a process involving both physical and chemical reactions taking place in a living organism. Moreover, our objectives involve the construction of multiset model for the proposed alternative P system.

To achieve our goal, we highlight some prerequisites in the areas of multiset and order relations on multisets as tools necessary for the study of variants of alternative P system presented. Secondly, we explore the basics of the theory of formal language as an instrument used in investigating the completeness of computational models both of biological and physical orientations. The central objective of the presentation is to critically study some of the variants of P systems studied in the past, particularly having biological inspiration and thereby investigate a computationally more powerful P system, called *alternative P system*.

3.1 Preliminaries of Multiset

Definition 3.1 *Multiset*

A *multiset* (*mset*, for short) is an ordered collection of *objects*, in which, unlike a set, multiple occurrences of objects are admitted. Formally, a multiset M, built upon a set S (called a *domain set* or *base set* of M) is a mapping $M: S \longrightarrow \mathbb{N}$, (where $\mathbb{N}$ is the set of all non-negative integers) and $M(x)$ denotes the number of occurrences of x in M. The *empty multiset* over S, defined $M(x) = 0\ \forall x \in S$, is denoted by $\emptyset$. An object x is a member of M if $M(x) > 0$. Each individual occurrence of x in M is called an *element* of M. The set of all finite multisets built upon a domain set S is denoted by $\mathfrak{M}(S)$. It follows that a multiset M is a set in which $M(x) = 1$ for each $x \in S$. The *cardinality* of a multiset, denoted by $|M|$, or *Card*(M) is the sum of the multiplicities of all the objects in M. The *root* or *support* of a multiset M, denoted by M^* is defined as $M^* = \{x \in M: M(x) > 0\}$. The elements of the root set of a multiset are called the *generators* of that multiset. We shall assume all the multisets dealt with to be finite.

Definition 3.2 *Types of multisets*

A multiset is called *regular* if all its objects occur with the same *multiplicity*, for instance, the multiset $\{a, a, a, b, b, b, 3,3,3\}$ is a regular multiset. A multiset is called *simple* if all its objects are the same, for example, $\{a, a, a, a\}$ is a simple multiset containing a as its only object. Clearly, the root set of every simple multiset contains a single object. See Singh *et al*. (2007).

Definition 3.3 *Multiset equality*

Let E be a base set of the multisets M and N, then $M = N$ if and only if $M(x) = N(x)\ \forall x \in E$ where $M(x)$ and $N(x)$ denote the multiplicities of x in M and N, respecively.

Definition 3.4 *Submultiset*

Given a multiset M over a domain set S, a multiset A over D is called a *submultiset* of M written as $A \subseteq M$ or $M \supseteq A$ if $A(x) \leq M(x)$ for all $x \in D$, where $A(x)$ and $M(x)$ are the multiplicities of x in the multisets A and M, respectively. Also, if $A \subseteq M$ and $A \neq M$, then A is called a *proper submultiset* of M. A multiset is called the *supermultiset* in relation to its submultiset. A *whole submultiset* of a given multiset contains all multiplicities of common elements; while a *full submultiset* contains all objects of the supermultiset, and accordingly, every multiset contains a unique full submultiset, its root set (Singh *et al*. (2007), for details).

Definition 3.5 *Union, Intersection and additive union or sum or merge*

Let A and B be two multisets of finite cardinality over a given domain set D.

i. The *union* of A and B denoted $A \cup B$ is the multiset defined by $\max(A(x), B(x))$. That is, an object x occurring m times in A and n times in B, occurs $\max(m, n)$ times in $A \cup B$. For example, if $A = [a, a, b, b, d, d, d]$, $B = [a, b, c, c]$, then $A \cup B = [a, a, b, b, c, c, d, d, d]$.

ii. The *intersection* of A and B denoted $A \cap B$ is the multiset defined by $\min(A(x), B(x))$. That is, an object x with multiplicity m in A and multiplicity n in B has multiplicity of $\min(m, n)$ in $A \cap B$. Using the last example, $A = [a, a, b, b, d, d, d]$ and $B = [a, b, c, c]$ implies that $A \cap B = [a, b]$.

iii. The *additive union or sum or merge* of two multisets A and B denoted $A \uplus B$ is the multiset defined by $A(x) + B(x)$, direct sum of two numbers. That is, an object *x* occurring m times in A and n times in B, occurs $m + n$ times in $A \uplus B$. For example, given that $A = [a, a, b, b, d, d, d]$, $B = [a, b, c, c]$, then $A \uplus B = [a, a, a, b, b, b, c, c, d, d, d]$.

Singh *et al*. (2007) for further details.

iv. The *difference* of multisets A and B denoted $A - B$ is the multiset defined by $\max(A(x) - B(x), 0)$ for all x in D (Tella *et al.*, 2014).

3.2 Some Properties Holding for Multiset Operations

i. Commutativity: $A \cup B = B \cup A$,

$A \cap B = B \cap A$

$A \uplus B = B \uplus A$

ii. Associativity: $A \cup (B \cup C) = (A \cup B) \cup C$

$A \cap (B \cap C) = (A \cap B) \cap C$

$A \uplus (B \uplus C) = (A \uplus B) \uplus C$

iii. Identity: $A \cup \emptyset = A$

$A \cap \emptyset = \emptyset$

$A \uplus \emptyset = A$

iv. Idenpotency: $A \cup A = A$

$A \cap A = A$

however, $A \uplus A \neq A$

v. Distributivity: $A \cap (B \cup C) = (A \cap B) \cup (A \cap C)$

$A \cup (B \cap C) = (A \cup B) \cap (A \cup C)$

$A \uplus (B \cup C) = (A \uplus B) \cup (A \uplus C)$

$A \uplus (B \cap C) = (A \uplus B) \cap (A \uplus C)$

$A \cap (A \uplus B) = A$

$A \cup (A \uplus B) = A \uplus B$

$A \uplus B = (A \cup B) \uplus (A \cap B)$

The above identities can be verified from the interpretation of $\cup$, $\cap$ and $\uplus$ as the maximum, minimum and (direct) sum of two natural numbers. We also note that $\uplus$ is stronger than both $\cup$ and $\cap$ in the sense that neither $\cup$ nor $\cap$ distributes over $\uplus$, whereas $\uplus$ distributes over both $\cup$ and $\cap$, see Syropoulos (2000).

3.3 Some Basic Terminologies on Order Relation

Definition 3.6 *Order relation*

Let S be a set and R a relation on S. Then R is a *quasi-order* (or *pre-order*) if it is reflexive and transitive; a *proper* (or *strict*) order if it is irreflexive and transitive; a *partial order* (or simply an *order*) if it is reflexive, antisymmetric and transitive; a *total order* if it is a partial order and *connected* (or *determinate*). A *strict partial order* if it is *irreflexive*, *antisymmetric* and *transitive*. A strict order is usually denoted by the symbol $\prec$ while an order (a non strict partial order) is always denoted by the symbol $\preccurlyeq$ to indicate reflexivity.

It is therefore clear that every ordering is a relation. In most texts, the usual ordering is symbolized by '$<$' and is often used on the set of real numbers, whereas the symbol '$\prec$' is used for the generality of the items that could be considered as elements of a set (including real numbers). Moreover, some texts consider symbols such as '$\ll$' or '$\prec\prec$' for the orderings that exists among sets, multisets or similar structures. The symbols in question could be subscripted or superscripted as necessary as in $\prec_1$ or $\ll_M$ to distinguish among various variations of the orderings. Specifically, these symbols are used for the strict orderings in the order relations defined above.

A counterpart is where the symbols are attached the equality sign and are used for the non-strict aspect of the orderings. Such symbols include '$\leq$', '$\leqslant$' or '$\preccurlyeq$'. However, some texts do not include the equality sign to the more complex '$\ll$' or '$\prec\prec$' even when they do not mean the strict cases as long as the intended meaning is

understood. It is more so to keep the symbols as simple as possible. In this thesis, we shall employ such notations for the strict case of the ordering and explain the type of ordering as well as say whether the ordering is between sets or multisets in their various forms, where no subscript has been specified to lay such emphasis. Moreover, we shall use $\preccurlyeq$ to mean partial ordering (as opposed to the strict case $\prec$) among the basic elements of a set or multiset in which equality can be the case. In other words, whenever the symbol $\prec$ is used, it is interpreted to mean the strict case of the order $\preccurlyeq$ without further emphasis. We shall write '$\ll=$' to include equality where necessary and avoid the use of symbols such as '$\lesssim$' and '$\preccurlyeq\preccurlyeq$' to avoid ambiguity.

Definition 3.7 *Set ordering*

We define set ordering as follows: A set S is greater than a set T (written $S \gg T$) if and only if $\forall x \in T \backslash S, \exists y \in S \backslash T$ such that $y \succ x$.

As an illustration, assume, in the following examples, that alphabets represent incomparable elements while numbers represent comparable elements. Assume also that the alphabets are incomparable to the numbers. By this definition, the following relations hold:

$\{4\} \gg \{3,2,1,0\}$

$\{a, b, c\} \gg \{a, b\}$

$\{3,2, a, b, c\} \gg \{3, a, b, c\}$

$\{4, a, b, c\} \gg \{3,2, a, b, c\}$.

It can be verified that the set ordering so defined is a partial ordering. We used this definition when describing the ordering that exists among the so-called *references* in a set-based grid later.

Definition 3.8 *Disrelation*

We define a concept which may be well known to the reader, and which is useful for the construction of our variant of alternative P system. Let R be a relation on a set M. If there exists at least two different elements of M which are not related with respect to R, then the two elements are said to be *unrelated* in M with respect to R. This concept is known as *disrelation*. In other words, if $a, b \in M$ are such that a is not related to b with respect to R and b is not related to a with respect to R and $a \neq b$ then a and b are said to be unrelated in M with respect to R . Symbolically, we write as follows:

$$a \#_R b = \neg(a \; R \; b \text{ or } a = b \text{ or } b \; R \; a)$$

where the negation symbol '¬' is used to mean 'not' and is read as such. In extension, we have the definition of *incomparablility relation* below.

Definition 3.9 *Incomparability relation*

Let $\prec$ be a partial ordering on a set *S*. We denote by $x \#_{\prec} y$, the following relation:

$$\neg(x \prec y \text{ or } x = y \text{ or } y \prec x) \qquad (1)$$

read as "x and y are *incomparable* under '$\prec$'" for some elements x and y of S. Moreover, if there exist incomparable sets U and V, that is $\neg(U \ll V \text{ or } V \ll U \text{ or } U = V)$, then we write $U \# V$. Furthermore, we write $U \neg\ll V$ to mean U is not less than V. The same operators are valid for multisets. The symbol # is similarly used with elements of multisets to mean incomparability with respect to the orderings such symbols represent. Particularly, one writes # to mean incomparability with respect to $\prec$, the order symbol of the multiset in question where the subscript $\prec$ is omitted with the intention that the ordering to which incomparability is associated is understood. Except for emphasis, we shall usually write # without the subscripts. Thus,

$$a \# b = \neg(x \prec y \text{ or } x = y \text{ or } y \prec x)$$

Definition 3.10 *Inordering*

Two elements x and y are *comparably unequal* with respect to the ordering $\prec$ if and only if either $x \prec y$ or $y \prec x$. In other words x and y are comparably unequal if they satisfy the property:

$$\neg(x = y \text{ or } x \# y)$$

In this light, a multiset is said to be *unordered* if it contains no pairs of comparably unequal elements. Given that $a \# b,\ b \# c$ and $a \# c$, a typical unordered multiset is $M = \{a, a, a, b, b, c\}$.

Definition 3.11 *The Dershowitz-Manna definition of multiset ordering*

Let *S* be a domain set of multisets *M* and *N*. Then $M \lll N$ if there exist two multisets *X* and *Y* in $\mathfrak{M}(S)$ satisfying the following:

i. $\emptyset \neq Y \subseteq N$,

ii. $M = (N \backslash Y) \uplus X$, and

iii. $(\forall x \in X)(\exists y \in Y)[y \succ x]$.

In other words, $M \lll N$ if *M* is obtained from *N* by removing at least one element (those in *Y*) from *N*, and replacing each such element y by zero or any finite number of elements (those in *X*), each of which is strictly less than (in the sense of the ordering $\prec$) one of the elements *y* that have been removed. Informally, we say that *M* is smaller than *N* in this case. Similarly, $\ggg$ on $\mathfrak{M}(S)$ can be defined. For example, let $S = (\{0, 1, 2, \ldots\} = \aleph)$. Then, under the corresponding multiset ordering $\lll$ over $\aleph$, each of the following multisets {3,4}, {3,2,2,1,1,1,4,0} and {3,3,3,3,2,2} is less than the multiset {3,3,4,0}. The empty multiset $\emptyset$ is smaller than any multiset. See Dershowitz and Manna (1979).

Definition 3.12 *The Huet-Oppen definition of multiset ordering*

Let M and N be multisets over a base set E, then

$M \lll N$ iff $M \neq N$ &

$[M(x) > N(x) \Longrightarrow (\exists y \in E)x \prec y \;\&\; M(y) < N(y)]$

One importance of this definition is its usefulness in the ease of disproving an order relation as we shall discover in some of the proofs in this text. We shall denote this ordering by $\ll_{HO}$ (Huet and Oppen, 1980). The Dershowitz-Manna definition and the Huet-Oppen definition are equivalent as shown in the following Theorem 3.1 below.

Theorem 3.1

The Dershowitz-Manna ordering and the Huet-Oppen ordering are equivalent.

Proof

Given that $\ll_{DM}$ and $\ll_{HO}$ are the Dershowitz-Manna and Huet-Oppen orderings, respectively. Assume $M \ll_{HO} N$, and define X and Y as follows:

$Y(y) = \max\{N(y) - M(y), 0\}$,

$X(x) = \max\{M(x) - N(x), 0\}$.

We prove (i) of Definition 3.11, $Y \subseteq N$ is clear by definition. Because $M \neq N$ there exists z such that $M(z) \neq N(z)$. If $M(z) < N(z)$, then $z \in Y$, if $M(z) > N(z)$, by Definition 3.12, there exists x, such that $z \prec y$ and $M(y) < N(y)$. Hence, $y \in Y$. In both cases, $Y \neq \emptyset$. (ii) is true by construction. To prove (iii), let $x \in X$. By hypothesis, there exists y satisfying $x \prec y$ and $X \neq Y$. Without loss of generality, we may assume Y and X disjoint, ie., $Y(y) > 0 \Longrightarrow X(y) = 0$ and $X(y) > 0 \Longrightarrow Y(y) = 0$. Otherwise, Y would be replaced by $Y - Z$ and X by $X - Z$ where $Z(y) = \min(\{Y(y), X(y)\}$. Conversely, assume $M(x) > N(x)$. By (ii), $M(x) = N(x) +$

$X(x) - Y(x)$. This implies $X(x) > Y(x) > 0$, which means $x \in X$. By (iii) where exist $y \in Y$, i.e, $Y(y) > 0$, such that $x \prec y$. The value of M in y is $M(y) = N(y) - Y(y) + X(y) = N(y) - Y(y)$ because $X(y) = 0$. Thus, $M(y) < N(y)$. □

This proof was provided in Jouannaud and Lescanne (1982). We have, however, presented an alternative proof in Theorem 4.24.

3.4 Elementary Formal Language Prerequisites

We introduce some formal language specific ingredients of P system, by giving basic notions and notations of formal language theory, as well as elementary concepts leading to automata and complexity theory. The reader is referred to Hopcroft (1979) and Martin-Vide and Păun (2001) among other relevant materials for further information.

3.4.1 Strings and languages

An alphabet is a finite and nonempty set of symbols. Given an alphabet V, we denote by V^*, the set of all strings of symbols from V. It is the free monoid generated by V under the operation of concatenation. The empty string is denoted by λ and it does not contain any symbol, the set of nonempty strings over V is denoted by $V^+ = V^* \backslash \{\lambda\}$.

A *language over* V is defined as an arbitrary subset of V^*. A language that does not contain the empty string is said to be *λ-free.* Given a string $x, x_1, x_2 \in V^*$, x_1 and x_2 are called *prefix* and a *suffix* of x if $x = x_1 x_2$. Moreover, if $x = x_1 x_2 x_3$, then for some $x_1, x_2, x_3 \in V^*$, x_2 is called a *substring* of x. We denote by $Pref(x)$,

$Suff(x)$ and $Sub(x)$ the set of all prefixes, suffixes and substrings of a string x, respectively.

The length of a string $x \in V^*$ is the number of all occurrences of symbols from V in x, and is denoted by $|x|$, while the number of occurrences in x of a symbol $a \in V$ is denoted by $|x|_a$. The set of symbols from V occurring in a string x is denoted by $alph(x)$; given a language $L \subseteq V^*$, we define $alph(L) = \bigcup_{x \in L} alph(x)$.

Given an alphabet $V = \{a_1, a_2, \dots, a_n\}$ to every string $x \in V^*$ we can associate the Parikh vector $\Psi_V(x) = (|x|_{a_1}, |x|_{a_2}, \dots, |x|_{a_n})$. Given a language $L \subseteq V^*$, we can also define the Parikh image of L as $\Psi_V(L) = \{\Psi_V(x) \mid x \in L\}$. If FL is a family of languages, then $PsFL$ denotes the family of Parikh images of languages in FL.

A set M of vectors in N^n, for some $n \geq 1$, is said to be linear if there are some vectors $v_1, v_2, \dots, v_m \in N^n, m \geq 0$, such that $M = \{v_0 + \sum_{i=1}^{m} \alpha_i v_i \mid \alpha_1, \dots, \alpha_m \in N\}$. That is, M is semilinear in N^n if $M \subseteq N^n$ and every vector in M is a linear combination of some vectors in N^n. A *semilinear* set is a finite union of linear sets. A language $L \subseteq V^*$ is *semilinear* if its Parikh image $\Psi_V(L)$ is a semilinear set.

3.4.2 Operations with strings and languages

The usual set operations can be extended in a natural way to languages, as well as several operations which are specific for strings and sets of strings:

i. *The union of two languages* $L_1, L_2 \subseteq V^*$ is the set $L_1 \cup L_2 = \{x \mid x \in L_1 \wedge x \in L_2\}$.

ii. *The intersection of two languages* $L_1, L_2 \subseteq V^*$ is the set $L_1 \cap L_2 = \{x \mid x \in L_1 \vee x \in L_2\}$.

iii. *The difference of two languages* $L_1, L_2 \subseteq V^*$ is the set $L_1 \backslash L_2 = \{x \mid x \in L_1 \wedge x \notin L_2\}$.

iv. *The complementation of a language* $L \subseteq V^*$ is the set $L^c = \{x \mid x \in V^* \wedge x \notin L\} = V^* \backslash L$.

v. *The concatenation of two languages* $L_1, L_2 \subseteq V^*$ is the set $L_1 L_2 = \{x_1 x_2 \mid x_1 \in L_1 \wedge x_2 \in L_2\}$.

vi. *The Kleene closure of a language* $L \subseteq V^*$ is the set $L^* = \bigcup_{i \geq 0} L^i$, where $L^i = L^{i-1} L$, that is the concatenation of L by itself i many times.

vii. *The right quotient* of a language $L_1 \subseteq V^*$ with respect to a language $L_2 \subseteq V^*$ is the set $L_1 / L_2 = \{x \in V^* \mid xy \in L_1 \text{ for some } y \in L_2\}$.

viii. *The left quotient* of a language $L_1 \subseteq V^*$ with respect to a language $L_2 \subseteq V^*$ is the set $L_1 \backslash L_2 = \{x \in V^* \mid yx \in L_1 \text{ for some } y \in L_2\}$.

ix. *The right derivative* of a language $L \subseteq V^*$ with respect to a string $w \in V^*$ is the set $\partial_w^r(L) = \{x \in V^* \mid xw \in L\}$.

x. *The left derivative* of a language $L \subseteq V^*$ with respect to a string $w \in V^*$ is the set $\partial_w^l(L) = \{x \in V^* \mid wx \in L\}$.

Let $\mathcal{L}(U^*)$ denote the set of all languages over an alphabet U. A *finite substitution* over an alphabet V is a mapping $\sigma: V^* \to \mathcal{L}(U^*)$ such that, for each symbol $a \in V, \sigma(a)$ is a finite nonempty language, with $\sigma(\lambda) = \{\lambda\}$ and $\sigma(x_1x_2) = \sigma(x_1)\sigma(x_2)$ for all string $x_1, x_2 \in V^*$. If none of the languages $\sigma(a), a \in V$, contains the empty string, then the substitution is called $\lambda - free$. If each $\sigma(a)$ consists of a single string, then the substitution is said to be a *morphism*. The *inverse morphism* is defined as $\sigma^{-1}: U^* \to \mathcal{L}(V^*)$, where $\sigma^{-1}(w) = \{x \in V^* \mid \sigma(x) = \{w\}\}$. A morphism $\sigma: V^* \to U^*$ is called a *coding* if $\sigma(a) \in U$ for all $a \in V$, it is called a *weak coding* if $\sigma(a) \in U \cup \{\lambda\}$ for each $a \in V$. The mapping defined on strings are extended to languages in the natural way, for instance if $\sigma: V^* \to U^*$ is a morphism and $L \subseteq V^*$ then $\sigma(L) = \{\sigma(x) | x \in L\}$.

A family FL of languages is closed under an operation γ if $\gamma(L) \in FL$ whenever $L \in FL$. A family FL of languages is closed under an n-ary operation γ_n if, for all languages $L_1, \dots, L_n$ in FL, the language $\gamma_n(L_1, \dots, L_n)$ is also in FL. Any language which can be obtained by using finitely many times the operation of union, concatenation and Kleene closure is called *regular*.

A family of languages is called an *abstract family of languages* (AFL, in short) if it is closed under the operation of union, concatenation, +-Kleene closure, intersection with regular languages, λ-free morphisms and inverse morphisms. An AFL is a *full-AFL* if it is closed under the operation of Kleene closure. If a family of languages is closed under none of the cited operations, then it is called an *anti-AFL*.

3.5 Chomsky Grammars

Definition 3.13

A *Chomsky grammar* is a quadruple $G = (N, T, S, R)$, where N, T are disjoint alphabets of non-terminal and terminal symbols, $S \in N$ is the axiom, R is a finite set of rewriting rules (or productions) of the form $u \to v$, with $u \in (N \cup T)^+$, $v \in (N \cup T)^*$, with the condition that u contains at least one non-terminal symbol.

Given a string $w = w_1 u w_2$ and a production $u \to v$ in R, we can rewrite u by means of v and we obtain the string $z = w_1 v w_2$. This operation is called a direct derivation in G and it is denoted by $w \Longrightarrow z$. The reflexive and transitive closure of the relation $\Longrightarrow$ is denoted by $\Longrightarrow^*$, meaning that we have $w \Longrightarrow^* z$ if either $w = z$, or $w \Longrightarrow z_1 \Longrightarrow z_2 \Longrightarrow \cdots \Longrightarrow z_k = z$ for some $z_1, z_1, \dots, z_k \in V^*$, $k \geq 1$. Each string $w \in (N \cup T)^*$ such that $S \Longrightarrow^* w$ is called a *sentential form*. The language generated by G is the set of sentential forms over the terminal alphabet defined by $L(G) = \{x \in T^* \mid S \Longrightarrow^* x\}$.

According to the form of the rules on the set R, the Chomsky grammars can be classified as follows:

Type-0 grammar: the productions are of the form $u \to v$, where $u \in (N \cup T)^+$, $v \in (N \cup T)^*$ and u contains at least one non-terminal symbol;

i. *Type-1 (context sensitive) grammar*: the productions are of the form $u \to v$, where $u = u_1 A u_2$, $v = u_1 z u_2$, with $u_1, u_2 \in (N \cup T)^*, A \in N, z \in$

$(N \cup T)^+$. The form of the production of a contex sensitive grammar can equivalently be given in terms of the length of the string for all $u \to v$ in R, the condition $|u| < |v|$ must hold;

ii. *Type-2 (context free) grammar*: the productions are of the form $A \to v$, where $A \in N, v \in (N \cup T)^+$;

iii. *Type-3 (regular) grammar*: the productions are of the form $A \to aB, A \to a$ (right regular) or $A \to Ba, A \to a$ (left regular), where $A, B \in N, a \in T$.

The families of languages generated by context sensitive, context free and regular grammars are denoted by CS, CF, REG, respectively, and are naturally called *context sensitive*, *context free* and *regular languages*. The family of languages generated by type-0 grammars are called recursively enumerable languages and denoted by *RE*. We also denote by *FIN* the family of finite languages. Among these families, the following strict inclusions (describing the Chomsky hierarchy) hold: $FIN \subset REG \subset CF \subset CS \subset RE$: The families of languages REG, CS, RE are full-AFL, while CS is an AFL.

Example 3.1

A list of languages in the Chomsky hierarchy:

$$L_1 = \{a^m b^n \mid m, n \geq 1\} \in REG$$

$$L_2 = \{a^n b^n \mid n \geq 1\} \in CF \backslash REG$$

$$L_3 = \{a^n b^n c^n \mid n \geq 1\} \in CS \backslash CF$$

$$L_4 = \{a^{2^n} \mid n \geq 1\} \in CS \backslash CF$$

$$L_5 = \{xcx \mid x \in \{a, b\}^*\} \in CS \backslash CF$$

3.6 Some Elements of Computability

3.6.1 Automata

Automata are computing devices which work in the opposite direction with respect to grammars: while grammars generate the strings of a language, automata can recognize the strings given as input, telling whether or not they belong to a specified language.

The basic families of languages of the Chomsky hierarchy are characterized by different kinds of recognizing automata. We present here only two types of automata, corresponding to regular and recursively enumerable languages (the families at the extremes in the Chomsky hierarchy).

Definition 3.14 ***Finite automata***

A (nondeterministic) finite automaton is defined by the construction $A = (Q, \Sigma, q_0, \delta, F)$, where Q is a finite set of states, Σ is a nonempty finite set of input symbols (such that $Q \cap \Sigma = \emptyset$), $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of final states, and $\delta: Q \times \Sigma \to P(Q)$ is the transition function. We denote by $\delta(q, a)$ the set of all states $p \in Q$ such that there is a transition from q to p on input a. If $card(\delta(q, a)) \leq 1$ for all $q \in Q, a \in \Sigma$, then the automaton is said to be deterministic

(in this case, if $\delta(q, a) \neq \emptyset$, then we can identify it with a state in Q, hence we can consider δ as a partial function from $Q \times \Sigma$ to Q).

We can picture a finite automaton as a device consisting of an input tape (which collects the symbols from Σ), a finite control which is in some state from Q, and a reading head scanning a symbol of the tape from left to right.

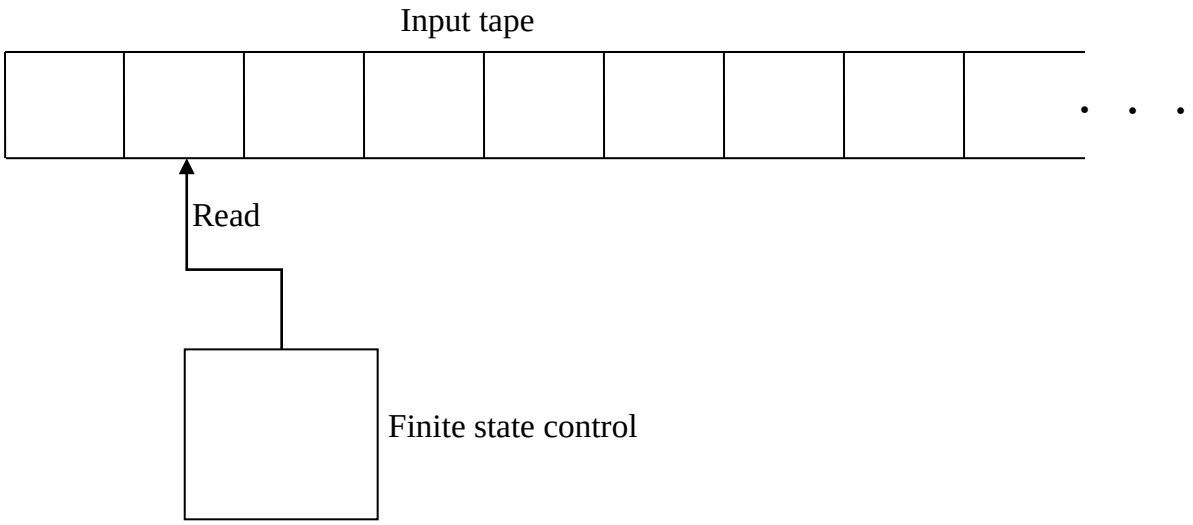


Figure 1: Finite state automaton

In one move, if the finite automaton is in the state $q \in Q$ and it is reading the input symbol $a \in \Sigma$, then it enters one of the next states in $\delta(q, a)$ and moves its head one symbol to the right.

To formally describe the behavior of the finite automaton on an input string and not only on an input symbol, we must consider an extension of the transition function, namely $\hat{\delta} : Q \times \Sigma^* \to \mathcal{P}(Q)$, such that

i. $\hat{\delta}(q, \lambda) = q$

ii. $\hat{\delta}(q, xa) = \bigcup_{q' \in \hat{\delta}(q,x)} \delta(q', a)$, where $x \in \Sigma^*, a \in \Sigma$ and $q, q' \in Q$.

The first condition disallows a change in state without any input, the case when the head is reading an empty cell on the tape. The second condition indicates that, starting in state q and reading the string x followed by the input symbol a, the finite automaton can be in any of the states which are reachable from the state q_0 after reading the symbol a. A string $w \in \Sigma^*$ is said to be accepted (or recognized) by the finite automaton if, starting with the finite control in the initial state q_0 and with the head reading the first symbol in w, the automaton reaches a final state after the last symbol in w has been scanned. Hence, the language recognized by A is the set of strings $L(A) = \{w \in \Sigma^* \mid \hat{\delta}(q_0, w) \text{ contains a state in } F\}$. Nondeterministic and deterministic automata are known to be equivalent in the following sense: they both characterize the family of regular languages REG.

3.6.2 Register machine

Informally speaking, a *register machine* consists of a specified number of registers which can hold any natural number, and which are handled according to a specified program consisting of labeled instructions; the registers can be increased or decreased by 1 – the decreasing being possible only if a register holds a number of greater than or equal to 1 (we say it is nonempty).

Formally, a deterministic *register machine* is a device $M = (m, B, l_0, l_h, R)$, where $m \geq 1$ is the number of registers, B is the set of instruction labeled, l_0 is the intial label l_h is the halting label, and R is the set of instructions labeled by elements from B(R is also called the *program* of the machine). The labeled instructions are of the following forms:

i. $l_1 : (\mathrm{ADD}(r), l_2)$ (add 1 to register r and go to the instruction with label l_2),

ii. $l_1 : (\mathrm{SUB}(r), l_2, l_3)$ (if register r is not empty, then substract 1 from it and go to the instruction with label l_2, otherwise go to the instruction with label l_3),

iii. $l_h : \mathrm{HALT}$ (the halt instruction which can only have the label l_h).

A register machine is used to recognize a number in the following manner: we start with all registers being empty, we introduce a number in a distinguished register (say, the first register), and we start to compute with the instruction labeled by l_0; if the computation reaches the instruction labeled by l_h : HALT (that is, it halts), then the number is recognized, otherwise it is not recognized. The set of all numbers recognized by M is denoted by $N(M)$. It is known (see Minsky (1967) and Freund and Oswald (2002)) that deterministic register machines (with three registers only, but this aspect is not important in what follows) recognize exactly the family *NRE*, of Turing computable numbers.

3.6.3 Turing machine

A *Turing machine* is a computational device consisting a finite state control, an input tape (which is infinite to the right) and a read/write head which can read symbols from the tape and also rewrite the scanned symbols. Formally, a (nondeterministic) Turing machine is a construction $M = (Q, \Sigma, \Gamma, \flat, q_0, F, \delta)$ where Q is a finite set of states, Σ is the tape alphabet, $\Gamma \subseteq \Sigma$ is the input alphabet, $\flat \in \Sigma \backslash \Gamma$ is the blank symbol, $q_0 \in Q$ is the initial state, $F \subseteq Q$ is the set of final states, and δ is a function from $Q \times \Sigma$ to $\mathcal{P}(Q \times \Sigma \times \{L, R\})$, whith L, R symbols not in Σ.

The mapping δ describes the moves of the machine: if $(q', b, D) \in \delta(q, a)$ for some $q, q' \in Q, a, b \in \Sigma, D \in \{L, R\}$, then the machine reads the symbol a in the state q and changes its state to q', it replaces the symbol a with symbol b and moves the head to the left if $D = L$, to the right if $D = R$. If $card\big(\delta(q, a)\big) \leq 1$ for all $q \in Q, a \in \Sigma$, then the Turing machine is said to be deterministic.

A Turing machine can be pictured as a finite automaton, though there are some differences in the functioning of the two devices, e.g. the Turing machine can move its head to the left as well as to the right of the tape, it can erase the scanned symbol (replacing it with the blank symbol).

A configuration of a Turing machine M is a triple $\langle x, q, y \rangle$, where $x \in \Sigma^*$ is the string written on the tape, $q \in Q$ is the state of M and $y \in \Sigma^*(\Sigma\{\flat\}) \cup \{\lambda\}$ is the portion of the string whose left-end is currently scanned by the head. Note that both x

and y may be empty, the blank symbol may appear in x, y, but not in the last position of y. We can define a direct transition relation $\dashv_M$ among two configurations of M (that is, a step of the computation) in the following way:

i. $\langle x, q, ay\rangle \vdash_M \langle xb, q', y\rangle$ if and only if $(q', b, R) \in \delta(q, a)$
(right move, using a symbol $a \in \Sigma$)

ii. $\langle x, q, \flat\rangle \vdash_M \langle xb, q', \flat\rangle$ if and only if $(q', b, R) \in \delta(q, \flat)$
(right move, using a symbol $a \in \Sigma$)

iii. $\langle xc, q, ay\rangle \vdash_M \langle x, q', cby\rangle$ if and only if $(q', b, L) \in \delta(q, a)$
(left move, using a symbol $a \in \Sigma$)

iv. $\langle xc, q, \flat\rangle \vdash_M \langle x, q', cb\rangle$ if and only if $(q', b, L) \in \delta(q, \flat)$
(left move, using a symbol $a \in \Sigma$)

where $x, y \in \Sigma^*, a, b, c \in \Sigma\backslash\{\flat\}, q, q' \in Q$. We denote by $\vdash_M^*$ the reflexive and transitive closure of the relation $\dashv_M$.

The language recognized by a Turing machine M is defined as $L(M) = \{w \in \Gamma^* \mid \langle \lambda, q_0, w\rangle \vdash_M^* \langle x, q, y\rangle$ for some $x, y \in \Sigma^*, q \in F\}$, that is the set of input strings over Γ for which the machine reaches a final state, starting from the initial state and scanning the first symbol of the input string.

There is an equivalent mode of defining the language accepted by a Turing machine: one considers the set of strings $w \in \Gamma^*$ such that the machine, starting from the initial state and with w as input, reaches a configuration where no further moves

are possible (we say that the machine has reached a *halting state*). Note that it is also possible that a Turing machine never reaches a halting state on some input, as its head is allowed to move both directions on the tape. Hence, given an input string w, it may happen that the Turing machine reaches a final state, or it halts in a non-final state, or it never halts (it enters an infinite loop). In the last two cases, the input string w is not accepted by the Turing machine.

Observe that, given a nondeterministic Turing machine M and an input string w, usually there are many different computations of M on input w. Each computation corresponds to a path in the tree associated to M, which can be naturally defined as follows: the root corresponds to the initial configuration of M; each node corresponds to a configuration of M, and an arc between two nodes corresponds to a direct transition between the configurations of M associated to such nodes; the leaves are (final or non-final) halting configurations of M. Note that there also might be infinite paths in the tree, corresponding to the possible non-halting computations of M.

It is well known that the power of deterministic and nondeterministic Turing machine is equivalent; they both characterize the family of recursively enumerable languages. An important aspect of Turing machines is that they can be seen as a model of the computing capability of a general-purpose computer. They characterize the family of languages generated by type-0 grammars and, in addition to being language acceptor; they may be viewed as computers of functions from integers to integers. Actually, they are able to compute any partial recursive function. The

importance of Turing machines is a direct consequence of the Church-Turing thesis: the intuitive notion of computable functions" is identified with the class of partial recursive functions. Thus, the model of Turing machine is computationally complete. Many other formalisms of computation have been defined, but none of them have been proven to be strictly more powerful than Turing machines. Another relevant characteristic of Turing machines is that there exists a Turing machine U that simulates the computation of an arbitrary Turing machine M on any of its input. Such machine is called a universal Turing machine. Thus, we can say that Turing machines are both complete and universal computing devices.

3.7 Membrane Systems

Membrane systems, or P systems, have been recently introduced in Păun (2000) as a class of parallel and distributed computing devices inspired by some biological features of real cells. Four fundamental features are used in this computing model: a membrane structure where objects evolve according to specified evolution rules, which also determine the communication of objects between membranes. The membrane structure consists of several membrane-delimited compartments, hierarchically arranged and embedded in a main compartment, whose delimiting membrane is called skin membrane. We can also define what a region is: the space of a compartment delimited by its membrane and the membranes of the other compartments (if any) placed immediately inside it. With each region of a membrane structure we can associate a set of objects, which can be symbols or strings over a

given alphabet, and a set of evolution rules, which determine the transformation of objects, as well as their communication between adjacent compartments. In this way, different regions in the membrane structure can communicate with each other by sending or receiving objects. The evolution rules are applied in a nondeterministic and maximally parallel way: all the objects which can evolve should evolve. Moreover, a strict partial order relation can be defined over the set of evolution rules present inside a compartment, indicating a priority relation among them. Due to the application of evolution rules, a membrane can also be dissolved: it disappears from the membrane structure, its set of rules is removed but its set of objects still remains. Hence, the evolution rules do not only modify the objects and their placing, but also intervene in changing the membrane structure. A computing device is thus obtained, starting from an initial configuration (described by a fixed membrane structure containing a certain number of objects and rules) and letting the system evolve. A universal clock is

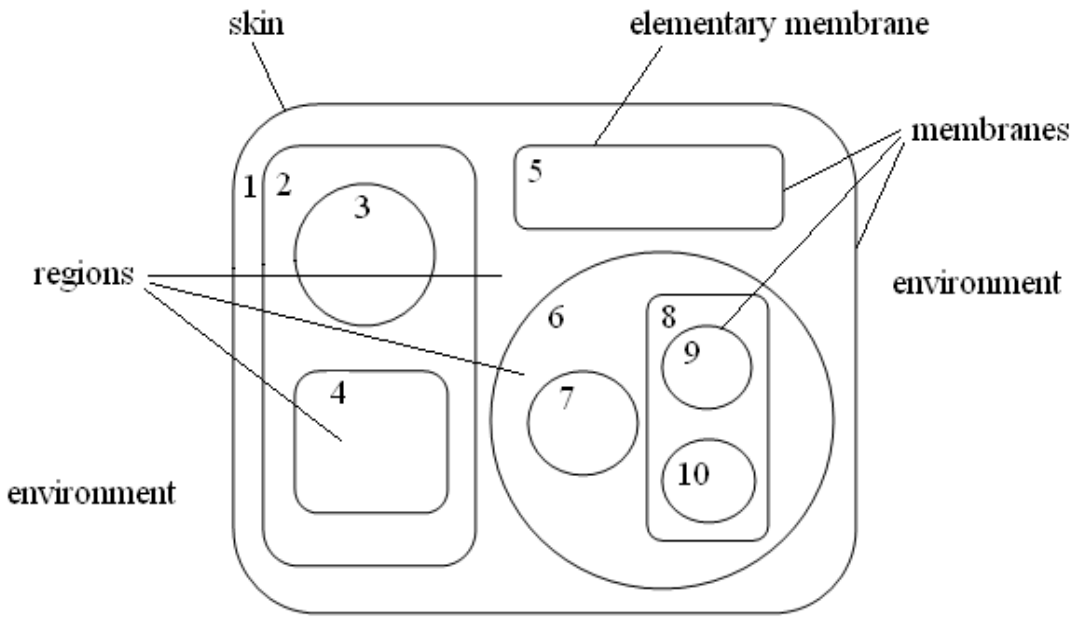


Figure 2: A membrane structure

assumed to exist: at each step, all rules from all regions are simultaneously applied to all objects which can be the subject of an evolution rule. When no further rule can be applied, the computation halts and we get the result in a prescribed way. On the contrary, if there is at least one rule that can be applied forever, then the computation is unsuccessful and no output is obtained.

In the following, we will give the basic definitions of P system and we will present some variants, together with a survey of gained results. We introduce the basic notions of this theory and the different proof techniques. For further details and for more information, examples and other variants we refer to Păun (2000) and Păun (2002).

Definition 3.15

The definition of a P system is based on the notion of membrane structure, which is introduced in terms of the language MS over the alphabet $\{[,]\}$, consisting of the strings recurrently defined as follows:

i. $[\,] \in MS$;

ii. If $\mu_1, \ldots, \mu_n \in MS, n \geq 1$, then $[\mu_1 \ldots \mu_n] \in MS$;

iii. nothing else is in MS.

We assume that each pair of well matching parentheses [ ] can be labeled in a one-to-one manner by an integer in the set $\{1, \ldots, n\}$.

We define now a relation over the elements of MS. We write $x \sim y$ if and only if it is possible to write the two strings as $x = [_1 \ldots [_2 \ldots]_2 \ldots [_3 \ldots]_3 \ldots]_1$ and $y = [_1 \ldots [_2 \ldots]_2 \ldots [_3 \ldots]_3 \ldots]_1$, that is two pairs of parentheses which are neighbors at the same hierarchical level can be interchanged, together with their contents. Thus, the order of pairs of parentheses is irrelevant, what matters is their respective relationship.

We denote by $\sim^*$ the reflexive and transitive closure of $\sim$. The set of equivalence classes with respect to the equivalence relation $\sim^*$ is denoted by $\overline{MS}$. Any element in MS is called a membrane structure. Each pair of matching parentheses [ ] appearing in a membrane structure $\overline{MS}$ is called a membrane. The external membrane of a membrane structure is called the skin membrane, while any membrane without any other membrane inside (appearing in a membrane structure in the form [ ]) is

called an elementary membrane. The number of membranes in μ is called the degree of μ and it is denoted by $deg(\mu)$. The depth of a membrane structure μ, $dep(\mu)$, is recurrently defined as:

i. If $\mu = [\,]$, then $dep(\mu) = 1$

ii. If $\mu = [\mu_1 \dots \mu_n]$, for some $\mu_1, \dots, \mu_n \in MS$, then $dep(\mu) = 1 + \max\{dep(\mu_i) \mid \leq i \leq n\}$.

A membrane structure can be represented as in Figure 2, which also makes clear the notion of region, the closed space delimited by a membrane and by the membranes immediately inside it. Note that a membrane structure of degree n contains exactly n regions, each one associated with a membrane and labeled with the same integer of that membrane. The whole space outside the skin membrane is called the outer region or environment (no labels are associated with it). If, given a finite alphabet V, we provide a membrane structure μ with some multisets $M_i : V \to N$, one for each membrane $i \in \{1, \dots, n\}$, then we obtain a super-cell, which is a construction of the form $SC = \{(\mu, M_1, \dots, M_n) \mid \mu$ is a membrane structure of degree n and $M_i : V \to N, i = 1, \dots, n$, is the multiset associated with the region $i\}$

Any multiset M_i can be empty; in particular, all of them could be empty, thus a membrane structure is a super-cell. The elements of a multiset are called objects, the multiset present in a region is the contents of this region. The total number of objects in an elementary membrane i is called the size of i, denoted by $size(i)$. If a

membrane j is placed inside a membrane i and they contribute to delimiting the same region (namely, the one delimited by the membrane i), then the objects inside the region associated with i are said to be adjacent to membrane j. The graphical representation and several of the notions defined for membrane structure, such as degree, depth, etc., can be naturally extended to super-cells. The last additional feature before giving the formal definition of a P system is the possibility of objects to evolve, according to certain specified rules. An evolution rule over an alphabet V is a pair (u, v), which will be written as $u \rightarrow v$, where u is a (non empty) string over the alphabet V and $v = v'$ or $v = v'\delta$, where v' is a string over $V \times \left(\{here, out\} \cup \left\{in_j \mid 1 \leq j \leq n\right\}\right)$, and δ is a special symbol not in V, in a natural sense.

Once we have provided a super-cell with multisets and evolution rules, we can give the formal definition as in Păun (2000) of a transition P system of degree $n, n \geq 1$, which is a construction

$$\Pi = (V, T, C, \mu, M_1, \ldots, M_n, (R_1, \rho_1), \ldots, (R_n, \rho_n), i_0)$$

where:

i. V is the alphabet of the system;

ii. $T \subseteq V$ is the terminal (or output) alphabet;

iii. $C \subseteq V, C \cap T = \emptyset$ is the alphabet of catalysts;

iv. μ is a membrane structure consisting of n membranes, which are injectively labelled by numbers in the set $\{1, \ldots, n\}$;

v. $M_1, \dots, M_n$ are multisets over V , representing the objects initially present in the regions $1, \dots, n$ of the system;

vi. $R_1, \dots, R_n$ are finite sets of evolution rules associated with the regions $1, \dots, n$ of μ;

vii. $\rho_1, \dots, \rho_n$ are strict partial order relations defined over $R_1, \dots, R_n$ respectively, specifying a priority relation among the evolution rules;

viii. i_0 is a number in the set $\{1, \dots, n\} \cup \{\infty\}$, indicating the output region.

Some more remarks about the definition of a P system are to be given, before explaining its functioning as a computing device.

Any of the multisets $M_1, \dots, M_n$, can be empty, as well as any of the sets of evolution rules $R_1, \dots, R_n$, or any of the strict partial order relations $\rho_1, \dots, \rho_n$.

If Π contains evolution rules whose *radius* is greater than 1, then it is said to be *cooperative*, otherwise it is a *non-cooperative* system. A particular class of cooperative systems is that of catalytic systems: whenever the radius of an evolution rule is strictly greater than 1, then such rule is of the only form $ca \to cv$, where $c \in C, a \in V \backslash C, v \in (V \backslash C)^*$; moreover, no other rule contains any catalytic symbols. If $C = \emptyset$, then the system is *non-catalytic*.

According to the strict partial order relation $\rho_i, 1 \leq i \leq n$, a rule $r \in R_i$ is said to be with priority if there exists another rule $r' \in R_i$ such that $r > r'$ or $r' > r$. If r is a rule with priority and there is no other rule r' such that $r' > r$, then r is a rule of

highest priority. Note that, as ρ_i is a (strict) partial order relation, there can be more than one rule with highest priority. A rule r is said to be without priority if it does not compare in ρ_i.

The $(n+1)$-tuple $(\mu, M_1, \dots, M_n)$ constitutes the initial configuration of the system. In general, any sequence of the form $(\mu', M'_{i_1}, \dots, M'_{i_k})$ constitutes a configuration of Π where $M'_{i1}, \dots, M'_{ik}$ are multisets over V and μ' is a membrane structure obtained by removing from the original membrane structure μ all membranes different from $i_1, \dots, i_k$ (and not equal to the skin membrane), according to the dissolving action which will be explained in the following. The labels $i_1, \dots, i_k$ of the membranes in μ' are always numbers in the set $\{1, \dots, n\}$, corresponding to the initial labels of the membranes in μ, that is in successive configurations the initial labeling of the membranes (which have not been removed from μ) is never modified. Given two configurations $C_1 = (\mu', M'_{i_1}, \dots, M'_{i_k}), C_2 = (\mu'', M''_{j_1}, \dots, M''_{j_l})$ of Π, we write $C_1 \Longrightarrow C_2$ and we say that there is a transition from C_1 to C_2 if we can pass from C_1 to C_2 by using the evolution rules in $R'_{i_1}, \dots, R'_{i_k}$ using the following prescriptions.

Consider a rule $r : u \to v$ in R'_{i_s} and look at the region i_s in μ'. If all the objects in the multiset u appear in the region i_s (formally, $u \subseteq M'_{i_s}$), then r could be applied over r according to the priority relation defined by ρ_{i_s}. All rules are simultaneously examined, in the decreasing order of their priority (if any), and objects are accordingly assigned to rules. A rule r can be applied when there are enough

copies of the objects whose evolution it describes and which were not “consumed" by rules of higher priority; that is, objects are assigned to the rules which have the highest priority and, in the case that two or more rules of this type compete for the same objects, then the rule to be applied is nondeterministically chosen among them. Then, when no more rules with the highest priority can be applied (irrespectively of the objects they involve), the rules with a lower priority are considered. Again, if there are many rules with the same priority which compete for some objects, then the rule to be applied is non-deterministically chosen among them.

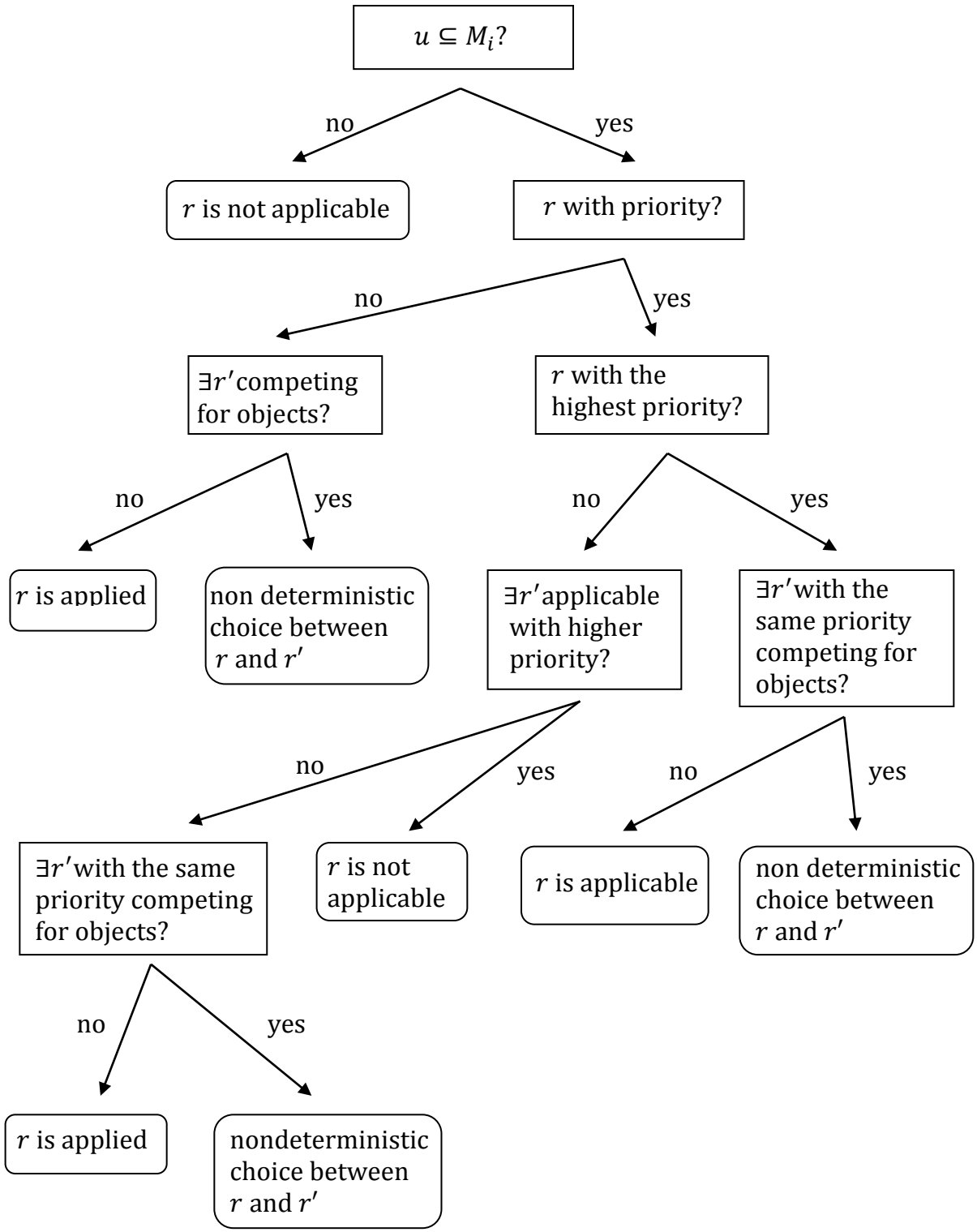


Figure 3: Application of an evolution rule $r : u \to v$ according to some priority

All the objects which can be the subject of evolution rules must be involved in the rule application. This use of the priority relations corresponds to the strong interpretation: if a rule with a higher priority is used, then no other rules with less priority can be applied, even if they do not compete for the same objects. Nonetheless, also a weak interpretation of the priority rules can be considered: a rule with less priority can be applied at the same time with the rule of higher priority if there are "free" objects which are not involved by the rule with the higher priority. The rules without priorities can be applied at any time, even together with rules with priorities.

The result of the application of the rule $(r : u \to v) \in R'_{i_s}$ is determined by v:

i. if an object appears in v in the form $(a, here)$, then it remains in the same region is (the indication here will often be omitted);

ii. if an object appears in v in the form $(a;\ out)$, then i_t exits the membrane i_s and becomes an element of the region immediately outside it. If the membrane is i_s the skin membrane, then the object leaves the system and it will never come back;

iii. if an object appears in v in the form (a, in_t) and i_t is a membrane immediately inside i_s, then the object i_s added to the multiset M_{i_t} . If i_t is not a membrane immediately inside i_s, then the application of the rule is not allowed;

iv. if the symbol δ appears in v, then membrane i_s is removed (we say it is dissolved) from μ' and, at the same time, the couple (R'_{i_s}, ρ_{i_s}) is erased, the multiset M'_{i_s} is added to the region immediately outside membrane i_s. The dissolving of the skin membrane is not allowed, because this would mean to lose the super-cell.

All these operations are performed in a maximal parallel mode: in one step, inside each region all applicable rules $u \to v$ are used over all occurrences of multisets u, and all regions are processed at the same time. Hence, the parallelism is maximal at both a global and a local level: the global level involves all membranes at the same time, the local level involves all evolution rules and all objects inside each region.

In the following, when referring to the maximal parallel application of rules, we will mean the working prescriptions just described in the previous paragraph. No contradictions arise because of multiple membrane dissolving or when both type of symbols (a, out) and δ appear simultaneously in v. No contradictions arise also when at the same time we have a symbol (a, in_t) outside a membrane t and δ inside this membrane: we assume that a is introduced in membrane t simultaneously with the dissolving of such membrane, thus a will remain in the region placed outside membrane t (that is, it remains in the membrane where it was originally placed before the application of the rule).

The functioning of the system determines its nature of a *computing device*. Starting from the initial configuration and applying the rules in the way described above, we obtain a sequence of transition among configurations: such a sequence is called a computation with respect to Π. A computation is successful if and only if it halts, that is no rule is further applicable to the objects appearing in any membrane of the system and, if an internal membrane i_0 is specified as the output membrane, then it must appear in the halting configuration as an elementary membrane. If $i_0 \in \{1, \dots, n\}$, then the system is said to work in the internal mode, and the result of a successful computation can be read in one of the following ways:

i. Consider the multiset which is present in the output membrane, the system is said to *generate multisets*;
ii. Consider the size of the output membrane, the system is said to *generate numbers*;
iii. If certain initial objects $a_{i_1}, \dots, a_{i_k}$ are specified in advance, then consider the occurrences $n_1, \dots, n_k$ which might be present in the output membrane at the end of the computation. Then the system is said to *generate relations*;
iv. If a fixed elementary membrane is specified as the input membrane, the system is said to *recognize a multiset* (the initial contents of the input membrane), a number (the size of the input membrane) or a relation (the occurrences of certain symbols in the input membrane).

The indication i_0 for the output membrane can be omitted if $i_0 = \infty$, in such a case the P system is said to be with external output, the objects which constitute the output of the system are collected outside the skin membrane, in the order they are ejected. Using these objects, a string is formed. When several objects are ejected at the same time, any permutation of them is considered. In this way, a language $L(\Pi)$ is generated by the system.

CHAPTER FOUR

PARTIALLY ORDERED MULTISETS FOR THE PROPOSED ALTERNATIVE P SYSTEM

4.0 Introduction

Since the appearance of a fundamental work of Dershowitz and Manna in Dershowitz and Manna (1979), various ramifications of multiset orderings have appeared (see Huet and Oppen (1980) and Jouannaud and Lescanne (1982), just to mention a few). The main objective of this section is to present an extension of set-based multiset ordering in Jouannaud and Lescanne (1982). Intuitively, similar to the manner in which *reference lines* on a *map* are used in finding specific points, the elements of a multiset can be *referenced* by a *grid*. A grid can be viewed as representing a multiset partition. Each element (in this case, set or submultiset as we shall see later) of the partition represents a *reference* on the grid. We introduce the concept of *difference grid* of two multisets in which some references are allowed to be empty. We thereafter define multiset ordering based on difference grid. As some references may be empty, the approach can be seen as an extension of the method used by Jouannaud and Lescanne (1982) in which each set in the partition contains at least one element of the partitioned multiset. The flexibility potential of the approach is demonstrated in the proofs of a number of results. In particular, the *pair-wise equality theorems for multisets* are proved (Peter and Singh, 2013). We begin our analysis with a lemma which reveals a consequence of incomparability between two sets.

Lemma 4.1

Let the relation S # T hold for sets S and T. There exist incomparable elements x and y such that $x \in S\backslash T, y \in T$ *and there exist incomparable elements v and w such that* $v \in T\backslash S$ *and* $w \in S$.

Proof

Given that S and T are sets such that $S \# T$ holds. By Definition 3.9, $S \neg\ll T$, $T \neg\ll S$ and $S \neq T$ hold. Since $S \neg\ll T$ holds, by Definition 3.7 $S \not\subseteq T$ and $\nexists y \in T$ such that $y \succ x \; \forall x \in S\backslash T$. It follows that $\exists x \in S\backslash T$ such that $x \succ y, x = y$ or $x \# y$ for some $y \in T$. If $x \succ y \; \forall x \in S\backslash T$, then $T \ll S\backslash T$. This implies $T \cup (S \cap T) \ll S\backslash T \cup (S \cap T)$, which implies $T \ll S$; and this is a contradiction of $S \# T$. Therefore, the statement $\exists x \in S\backslash T$ such that $x = y$ for some $y \in T$ or the statement $\exists x \in S\backslash T$ such that $x \# y$ for some $y \in T$, holds. Again, if $x = y \; \forall x \in S\backslash T$, then $S\backslash T \subseteq T$, which implies $S \subseteq T$. This contradicts $S \# T$. Therefore, $\exists x \in S\backslash T$ such that $x \# y$ holds. Similarly since $T \neg\ll S$, there exists incomparable elements v and w such that $v \in T\backslash S$ and $w \in S$. □

4.1 The Concept of Set-based Grid of a Partially Ordered Multiset

Consider the following (diagrammatic representation of a) multiset:

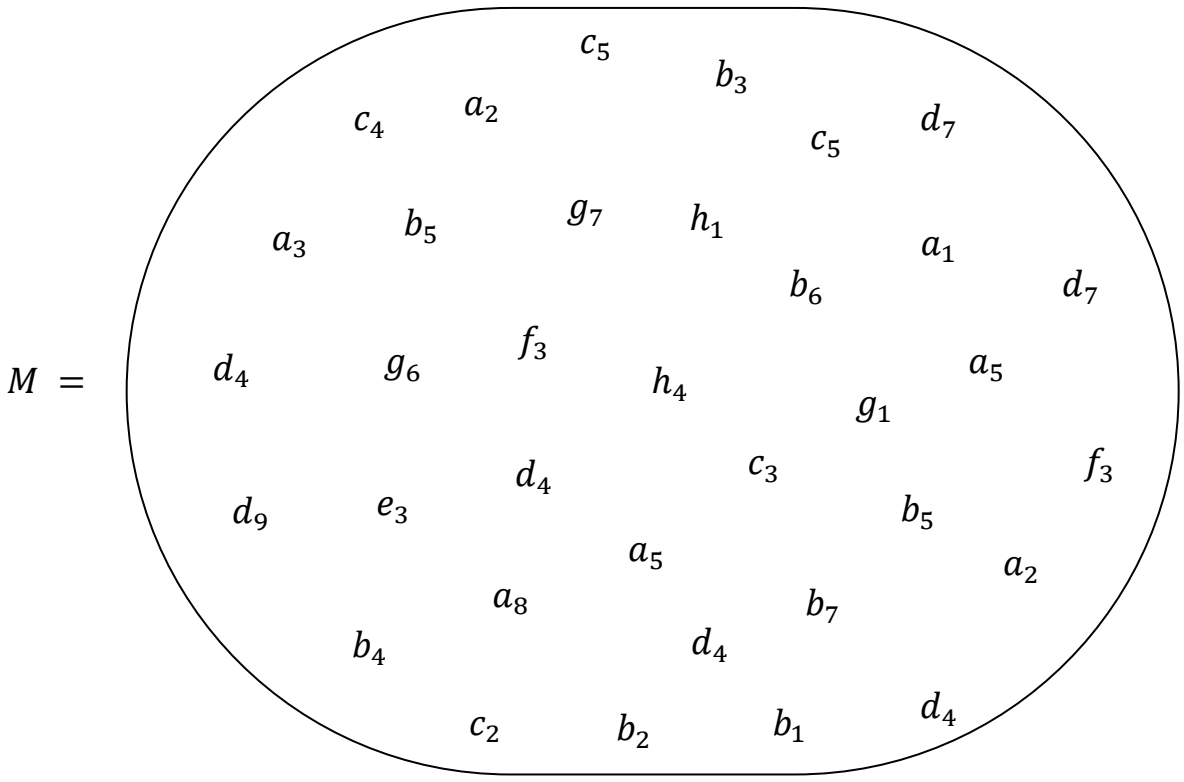


Figure 4: Diagrammatic representation of a multiset M for set-based multiset

Suppose that the elements of M are ordered as follows:

i. $a_{i+1} \prec a_i, b_{i+1} \prec b_i, c_{i+1} \prec c_i, d_{i+1} \prec d_i, e_{i+1} \prec e_i, f_{i+1} \prec f_i, g_{i+1} \prec g_i, h_{i+1} \prec h_i,$

ii. Elements with different letterings are incomparable.

iii. Elements with the same lettering and index are equal.

Consider now the following monotonic non-increasing sequences of elements of M.

$a_1 = a_1 \succ a_2 \succ a_3 \succ a_5 = a_5 \succ a_8 = a_8 = a_8 = a_8,$

$b_1 \succ b_2 \succ b_3 \succ b_4 = b_4 = b_4 \succ b_5 = b_5,$

$c_2 \succ c_3 = c_3 \succ c_5 = c_5$,

$d_4 \succ d_7 \succ d_9$,

$e_1 \succ e_3 \succ e_9$,

$f_3 = f_3$,

g_6,

h_1.

Let G_i be the set containing all the i^{th} elements from the sequences. Thus,

$G_1 = \{a_1, b_1, c_2, d_4, e_1, f_3, g_6, h_1\}$,

$G_2 = \{a_1, b_1, c_3, d_7, e_3, f_3\}$,

$G_3 = \{a_2, b_3, c_3, d_9, e_9\}$,

$G_4 = \{a_3, b_4, c_5\}$,

$G_5 = \{a_5, b_4, c_5\}$,

$G_6 = \{a_5, b_4\}$,

$G_7 = \{a_8, b_5\}$,

$G_8 = \{a_8, b_5\}$,

$G_9 = \{a_8\}$,

$G_{10} = \{a_8\}$.

One can immediately see that G_i's are submultisets of M, which are, in fact, sets of incomparable elements of M. Consequent upon the incomparability of the elements of each G_i, the number of submultisets of M in $[G_i]$ containing an element x

equals the multiplicity of x in M, henceforth denoted by $\alpha_{M(x)}$. Moreover, from the above, we have

$$[G_i] = [G_1 \gg G_2 \gg G_3 \gg G_4 \gg G_5 \gg G_6 \gg G_7 = G_8 \gg G_9 = G_{10}].$$

That is, $[G_i]$ is a non-increasing sequence. A permutation (an ordered sequence of elements with repetition allowed) is usually enclosed in square brackets. We call $[G_i]$ the *set-based grid* of M and each G_i, a *set-based grid reference* of M (or, simply a *reference* of M or *M-reference*). We give below, a formal definition of the concept. Viz: *set-based grid of a partially ordered multiset*.

Definition 4.1

Let $\preccurlyeq$ be a partial ordering defined on a set S and let M be a multiset of cardinality n over S. The permutation $[M_i]$ of subsets $M_1, M_2, \ldots M_m$ of $M, m \leq n$, is called the *set-based grid* of M if the following properties are satisfied:

i. $\forall(x, y, i)[x \in M_i, y \in M_i] \Longrightarrow [x \# y]$. (*Incomparability property*).

From (1) it follows that M_i's are sets.

ii. $\forall(i < j)[x \in M_j] \Leftrightarrow \exists y(y \in M_i)\ [x \preccurlyeq y]$. (*Order property*).

If for all x in M_j we have $x \prec y$ (in the strict sense of the ordering) for some y in M_i we say that M_j is strictly dominated by M_i, denoted by $M_j \lll M_i$). Obviously, $M_j = M_i$ if both contain the same elements. If $\prec$ is a total ordering on S, then any two elements of M are comparable (either equal or one greater than the other), and

each M_i would contain a single element. In such a case, $m = n$. If $\prec$ is a strict partial ordering on S then $\exists i$ for which M_i contains at least two incomparable elements of M, and $m < n$. If no two elements of M are comparable, then all the elements of M belong to the only reference available, which is M itself. The references of M are ordered by $\ll$. Thus, the set-based grid of M is the monotonic non-increasing (with respect to the set-based ordering) sequence of subsets of incomparable elements of M, in which the i^{th} subset contains the i^{th} elements from each of all the longest possible monotonic non-increasing sequences of comparable elements of M. This construct is a variant of Jouannaud and Lescanne (1982).

We next show that the second property in the above definition is equivalent to Definition 3.7 in the lemma that follows.

Lemma 4.2

Let S and T be finite sets. The following statements are equivalent.

(1) $S \gg T$ *iff* $S \neq T$ *and* $\forall x \in T\ \exists y \in S$ *such that* $y \succcurlyeq x$*;*

(2) $S \gg T$ *iff* $(x \in T \Longrightarrow x \notin S) \Longrightarrow (\exists y \in S \backslash T, y \succ x)$.

Proof

(1) $\Longrightarrow$ (2).

Let $x \in T$ and let $y \in S$ such that $y \succcurlyeq x$ hold. Thus, we have the following union of sets: $\{x \in T : y \succ x, y \in S\} \cup \{x \in T : y = x, y \in S\}$. By the additional constraint $x \notin S$ from (2), we have $\{x \in T : y \succ x, y \in S\} \cup \{x \in T : y = x, y \in S, x \notin S\}$. This

reduces the set to $\{x \in T: y \succ x, y \in S\} \cup \emptyset$. Hence, $y \succ x$. We claim $y \notin T$. Suppose the contrary and assume $y \in T$. Since $S \gg T$, by the hypothesis, $\exists y_1 \in S$ such that $y_1 \succcurlyeq y$, and by the result obtained $y \notin S$. This contradicts the hypothesis. Hence, $y \notin T$ holds.

(2) $\Longrightarrow$ (1).

It is easy to see that $T \neq S$. Let $x \in T$: If $x \notin S$, then from (2) $\exists y \in S \backslash T$ such that $y \succ x$. Also, if $x \in S$, then $\exists y \in S$ such that $y = x$ holds. The two statements imply $\exists y \in S$ such that $y \succcurlyeq x$. □

4.1.1 Set-based difference grid

Definition 4.2

Let $[M_i]$ and $[N_j]$ be the set-based grids of multisets M and N for $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$, respectively. Let $q = \max\{m, n\}$ and let $k = 1, 2, \ldots, q$. We construct the *set-based difference grid* $[M_k, N_k]$ of M and N as follows:

i. $M_q \neq \emptyset$ or $N_q \neq \emptyset$ (note the inclusive "or"),

ii. If $M_k \neq \emptyset$ and $N_k \neq \emptyset$ for a given k then $M_{k-1} \neq \emptyset$ and $N_{k-1} \neq \emptyset$.

M_k are the references of M in the grid $[M_k, N_k]$ while N_k are the references of N in the grid $[M_k, N_k]$. It is easy to see from (i) that $M_q \neq \emptyset$ if $n < m$, $N_q \neq \emptyset$ if $m < n$ and both are nonempty if $m = n$. In other words, if $m < n$ then $q = n$ and M_k is empty for $k = m + 1, m + 2, \ldots, n$, and if $n < m$ then $q = m$ and N_k is empty for $k = n + 1, n + 2, \ldots, m$. Unlike the references in the grid of a multiset, the references

in a difference grid of two multisets are empty up to the number of references with which the grid with more references exceeds the grid with fewer references. Also, the difference grid of two multisets is a collection of all the references from the individual grids of the multisets.

By way of illustration, consider two multisets $M = \{5,4,2,2,1, a, a, a, a, b, b\}$ and $N = \{6,3,3, a, a, a, b, b, c\}$. The difference grid of *M* and *N* is given in Table 1.

Table 1: The difference grid of $\{5,4,2,2,1, a, a, a, a, b, b\}$ and $\{3,3,3, a, a, a, b, b, c\}$

$\{5, a, b\}$	$\{4, a, b\}$	$\{2, a\}$	$\{2, a\}$	$\{1\}$
$\{6, a, b, c\}$	$\{3, a, b\}$	$\{3, a\}$	$\emptyset$	$\emptyset$

where the following two tables are tabular forms of the grids of M and N, respectively:

Table 2: The grid of $\{5,4,2,2,1, a, a, a, a, b, b\}$

$\{5, a, b\}$	$\{4, a, b\}$	$\{2, a\}$	$\{2, a\}$	$\{1\}$

Table 3: The grid of $\{6,3,3, a, a, a, b, b, c\}$

$\{6, a, b, c\}$	$\{3, a, b\}$	$\{3, a\}$

Definition 4.3

Let $[M_k, N_k]$ be the difference grid of two multisets M and N. A property p of references is said to be *pair-wise* iff whenever p is attributed to a reference M_i of $[M_k, N_k]$, then p is also attributed to the corresponding reference N_i of $[M_k, N_k]$. For instance, if p stands for 'nonempty', and M_i and N_j are nonempty for $i = j$ then M_i and N_j are *pair-wise nonempty*; if p stands for disjoint, and M_i and N_j are disjoint for $i = j$ then M_i and N_j are *pair-wise disjoint*. We now prove the following theorem.

Theorem 4.3 (*Set-based pair-wise equality theorem for multiset*)

Let M and N be multisets. Then $M = N$ iff $M_i = N_j$ for all $i = j$ where M_i and N_j are the respective references of M and N in the difference grid $[M_k, N_k]$ of M and N.

Proof

Given than M and N are multisets over a domain set S and suppose $M_i = N_j$ for all $i = j$. Since M_i are the references of M in the difference grid $[M_k, N_k]$, an element x of M belongs to M_i for some i. Since only incomparable elements belong to any set-based reference of a multiset, only one occurrence of x belongs to a reference containing x in the grid of M. Thus, we have $M(x) = \alpha_{M(x)}$. By hypothesis, N_j contains x for some j and only one occurrence of x belongs to N_j. Thus, again, $N(x) = \alpha_{N(x)}$. Hence, $\alpha_{M(x)} = \alpha_{N(x)}$, and it follows that $M(x) = N(x)\ \forall x \in S$. Therefore, $M = N$.

Conversely, let $M = N$. We claim $M_i = N_j$ for all $i = j$. Suppose the contrary holds and let l_0 be the smallest positive integer for which the claim is not true. Then either $M_{l_0} \gg N_{l_0}$ or $M_{l_0} \ll N_{l_0}$ or $M_{l_0} \# N_{l_0}$. Suppose $M_{l_0} \gg N_{l_0}$. By Definition 3.7 and Lemma 4.2, $\left(x \in N_{l_0} \Longrightarrow x \notin M_{l_0}\right) \Longrightarrow \left(\exists y \in M_{l_0} \backslash N_{l_0}, y \succ x\right)$. By Property (ii) of Definition 4.1, $y \notin N_l \; \forall l > l_0$. Hence, $M(y) > N(y)$, thereby contradicting the equality of M and N. Again, by Definition 3.7 and Lemma 4.2, if on the other hand $\left(x \in N_{l_0} \Longrightarrow x \in M_{l_0} \forall x\right)$, then $N_{l_0} \subseteq M_{l_0}$. Since $M_{l_0} \neq N_{l_0}$, let $u \in M_{l_0} \backslash N_{l_0}$. If $u \notin N_l$ for all $l > l_0$ then $M(u) > N(u)$. This is a contradiction to the equality of M and N. If on the other hand there exists $l > l_0$ such that $u \in N_l$, then by Property (ii) of Definition 4.1, there exists a positive integer k_1 less than l and an element z_1 of N_{k_1} such that $z_1 \succcurlyeq u$. Again, there exists a positive integer k_2 less than k_1 and an element z_2 of N_{k_2} such that $z_2 \succcurlyeq z_1 \succcurlyeq u$. Continuing this way, it can be seen that there exists an element z of N_{l_0} such that $z \succcurlyeq \cdots z_2 \succcurlyeq z_1 \succcurlyeq u$. This implies $z \succcurlyeq u$. Since $N_{l_0} \subset M_{l_0}$ then $z \in M_{l_0}$. This contradicts the incomparability of u and z in M_{l_0}. Similar argument holds if $M_{l_0} \ll N_{l_0}$. If $M_{l_0} \# N_{l_0}$ then by Lemma 4.1, there exists x and y where $x \# y$, such that $x \in M_{l_0} \backslash N_{l_0}$ and $y \in N_{l_0}$. If $x \notin N_l \; \forall l > l_0$, then $M(x) > N(x)$, and this is a contradiction of the equality of *M* and *N*. If on the other hand, there exists *l* where $l > l_0$ such that $x \in N_l$, then by Property (ii) of Definition 4.1, $z \in N_{l_0}$ such that $z \succ x$. Also, $z \notin M_{l_0} \; \forall l \geq l_0$*:* Furthermore, $M_i = N_i \; \forall i < l_0$. It

follows that $N(z) > M(z)$. This is a contradiction of the equality of M and N. Therefore, $M_i = N_j$ for all $i = j$. □

Theorem 4.4

The set-based grid of a partially ordered multiset is unique.

Proof

Assume $[M_i]$ and $\left[N_j\right]$ are two set-based grids of a partially ordered multiset M. By the set-based pairwise equality theorem $M_i = N_j$ for all $i = j$. Thus, $[M_i] = \left[N_j\right]$. Therefore, M has a unique set-based grid. □

4.1.2 Grid approach to Jouannaud-Lescanne's set-based multiset ordering

Definition 4.4

Let M and N be multisets and let $\ll$ be the set ordering in Definition 3.7. Then M precedes N in the sense of set-based grid (denoted $M \ll_{PS} N$) if and only if the following property is satisfied:

If $M_i = N_i$ for all i such that $M_i \neq \emptyset$ and $N_i \neq \emptyset$ then $M_i = \emptyset$ for the remaining i; otherwise if $\exists i$ such that $M_i \neg\ll N_i$ then $\exists j$ with $j < i$ such that $M_j \ll N_j$, for all set-based references M_i of M and N_i and N_j of N in the difference grid $[M_k, N_k]$ of M and N.

By this definition, the empty multiset $\emptyset$ can be seen to be smaller than every arbitrary nonempty multiset N. This is clear since as all the references of the grid of $\emptyset$

are empty, Properties (i) and (ii) of Definition 4.2 imply all the references of the grid of N are nonempty. It can also be seen that no element of $\emptyset$ is greater than or equal to any element of N. Also, the relation $\emptyset \lll_{PS} \emptyset$ and $N \lll_{PS} N$ do not exist considering the definition. This variant of set-based multiset ordering is equivalent to the variant presented in Jouannaud and Lascanne (1982) as we shall see in the theorem that follows. However, the grid variant proves to be more elegant at implementation.

Theorem 4.5

The ordering $\lll_{PS}$ is equivalent to the ordering $\lll_{JL}$.

Proof

Let $\lll_{JL}$ be the Jouannaud-Lescanne set-based (defined using *set-based partition*) multiset ordering and let $\lll_{PS}$ be the multiset ordering in Definition 4.4. Suppose M and N are multisets such that $M \lll_{PS} N$. We first show that $M \neq N$. Let $[M_k, N_k]$ be the difference grid of M and N for $k = 1,2, \ldots, p$. By the first property of Definition 4.2, $M_p \neq \emptyset$ or $N_p \neq \emptyset$. If either M_p or N_p is nonempty then $M \neq N$. If both M_p and N_p are nonempty then by the second part of Definition 4.4 or by the contrapositive of the first part of Definition 4.4, $M_k \neq N_k$ for some k. Therefore, by Theorem 4.3, $M \neq N$. The monotonic non-increasing sequence of all the nonempty references of the grid of M from $[M_k, N_k]$ is a set-based partition $\overline{M}$ of M in lexicographic order, each set containing incomparable elements (Properties (i) and (ii) of $\lll_{JL}$). Also, Property (iii) of $\lll_{JL}$ is embedded in Property (ii) of Definition 4.1. Thus, Properties

(i), (ii) and (iii) of $\ll_{JL}$ hold. From the first part of Definition 4.4, $M_i = N_i$ for all i such that $M_i \neq \emptyset$ and $N_i \neq \emptyset$ implies $M_i = \emptyset$ for the remaining i, then the number of sets in the partition of N is greater than the number of sets in the partition of M. Thus N is greater than M by lexicographic ordering. From the second part of Definition 4.4, $M_i \neg\ll_{PS} N_i$ implies $\exists j$ with $j < i$ such that $M_j \ll_{JL} N_j$, then the partition of N is greater than the partition of M by lexicographic ordering. Therefore, $M \ll_{JL} N$. Conversely, suppose $M \ll_{JL} N$. Let p and q be the number of sets in the partitions of M and N, respectively. Let $\ll$ be the set ordering in Definiton 3.7. Lexicographical extension of $\ll_{JL}$ entails the following:

i. $M_i = N_i$ for all $i = j$ implies $p < q$.

ii. $\exists i$ such that $M_i \neg\ll N_i$ implies $\exists j < i$ such that $M_j \ll N_j$.

Given (ii), either of the permutations $[M_i, \emptyset, \emptyset, \dots, \emptyset, N_j]$ where $\emptyset$ appears $q - p$ times or, $[M_i, N_j, \emptyset, \emptyset, \dots, \emptyset\,]$ where $\emptyset$ appears $p - q$ times is the difference grid of M and N. However, only the former is the difference grid of M and N for (i). □

Theorem 4.6

The ordering $\ll_{PS}$ is stronger than the ordering $\ll_{HO}$.

Proof

Consider the orderings $\ll_{HO}$ and $\ll_{PS}$. Suppose $M \ll_{HO} N$ holds. Let i_0 be such that $[M_{i_0}, N_{i_0}]$ is the first pairwise unequal references of the difference grid of M and

N with respect to $\ll_{PS}$. We claim $N_{i_0} \gg M_{i_0}$. Suppose the contrary, that is $N_{i_0} \neg \gg M_{i_0}$. Consider the contrapositive of Definition 3.7 on M_{i_0} and N_{i_0}. Thus, $x \in M_{i_0} \setminus N_{i_0}$ implies there exist w and x such that either $w \in N_{i_0} \setminus M_{i_0}$ and $w \not\succ x$ or $w \succ x$ and $w \notin N_{i_0} \setminus M_{i_0}$. Either case implies $w \notin N_l \; \forall l > i_0$ by Property (ii) of Definition 4.1. Furthermore, $M_i = N_i \; \forall i < i_0$ holds. This implies $N(w) \not\succ M(w)$. This contradicts the relation $M \ll_{HO} N$. Next, we show that the converse is not always true by the following example. Let $M = \{2,1\}$ and $N = \{2, a\}$ be such that $a \# 1$ and $a \# 2$. Then $M \ll_{PS} N$ since the reference $\{2, a\}$ of N is greater than either of the two references $\{2\}$ and $\{a\}$ of M. However, M and N are incomparable under $\ll_{HO}$. □

Theorem 4.7

The ordering $\ll_{PS}$ is stronger than the ordering $\ll_{DM}$.

Proof

The article Jouannaud and Lescanne (1982) contains a proof of the Dershowitz-Manna definition being equivalent to the Huet-Oppen definition. Since by Theorem 4.6 Definition 4.4 is stronger than the Huet-Oppen definition, it follows that $\ll_{PS}$ is stronger than $\ll_{DM}$. □

The above theorem shows that the Jouannaud-Lescanne set-based multiset ordering and the grid-based multiset ordering are not one stronger than the other. The two definitions are, in fact, the same. We emphasize here, what we mean by the strength

of an ordering over another. Generally, an ordering $\prec_1$ is *stronger than* or is *more powerful than* an ordering $\prec_2$ if and only if $\prec_2$ implies $\prec_1$ but not $\prec_1$ implying $\prec_2$. Though, the grid-based set-based ordering in Definition 4.4 and the conventional partition-based set-based ordering are equivalent, the choice of the former over the latter lies in its flexibility.

4.2 Redefined Submultiset-based Multiset Ordering via Grid

Like we have already seen, the idea of *set-based* grid of a partially ordered multiset is a grid where the references of the multisets are *sets* (subsets of the multiset in question). We modify the model a little by presenting a variant in which the references in the grid are submultisets of the multiset, hence the term *submultiset-based reference* of the grid in question. The approach is an extension of the method used by Joannaud and Lescanne. This thesis verifies that the approach is flexible. Such flexibility potential is demonstrated in the proofs of some important results, the *submultiset-based pair-wise equality theorem* in particular (Peter and Singh, 2015).

We start with the following lemma which establishes a consequence of incomparability between two multisets.

Lemma 4.8

Let E be a base set of multisets M and N. If $M \# N$ holds then there exist incomparable objects x and y of E such that $M(x) > N(x)$ and $N(y) > M(y)$.

Proof

Suppose the relation $M \# N$ holds. We have the following properties by the definition of incomparability on multisets: $M \neg\ll N,\ N \neg\ll M$ and $M \neq N$. Since $M \neg\ll N$, by the Huet-Open definition of multiset ordering, for all x such that $M(x) > N(x)$, $\nexists y \succcurlyeq x$ such that $N(y) > M(y)$. Either $y \prec x$ or $y \# x$ for such x and y. If $y \prec x$ for all of such x and y, then $M \gg N$; and this is a contradiction of $M \# N$. Therefore, there exists incomparable objects x and y such that $M(x) > N(x)$ and $N(y) > M(y)$.

□

Next, we construct below a submultiset-based grid for the multiset orderings.

4.2.1 The concept of submultiset-based grid of a partially ordered multiset

Consider the following (diagrammatic representation of a) multiset:

$M =$

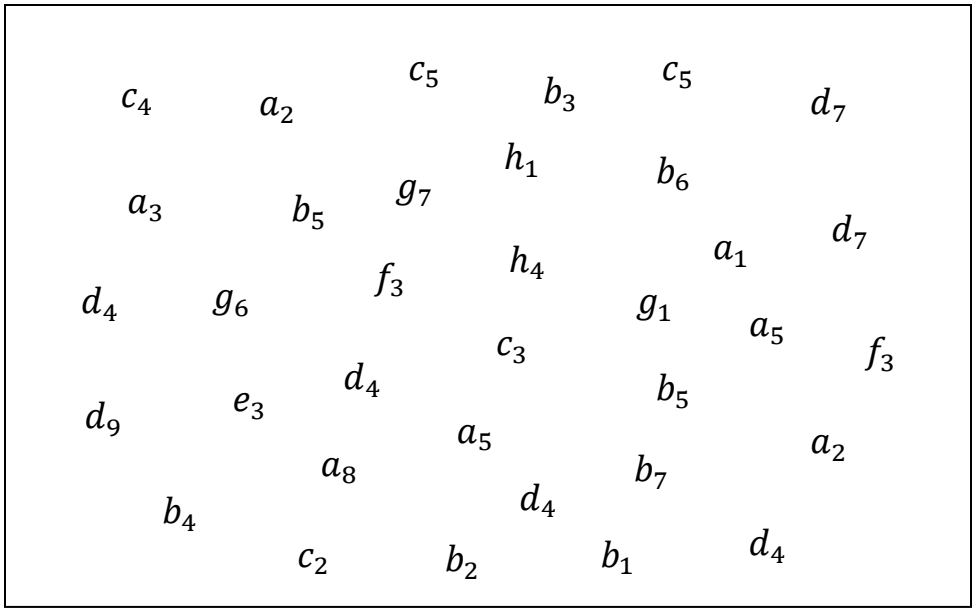


Figure 5: Diagrammatic Representation of a Multiset M for Submultiset-based

Assume the elements of M to satisfy the following properties:

i. $\omega_i \prec \omega_{i+1}$, where ω represents each of the alphabets in M.

ii. Elements with different letterings are incomparable.

iii. Elements with same lettering and index are equal.

Consider the following monotonic decreasing sequences of elements of M:

$a_1 \succ a_2 = a_2 \succ a_3 \succ a_5 = a_5 \succ a_8$,

$b_1 \succ b_2 \succ b_3 \succ b_4 \succ b_5 = b_5 \succ b_6 \succ b_7$,

$c_2 \succ c_3 \succ c_4 \succ c_5 = c_5$,

$d_4 = d_4 = d_4 = d_4 \succ d_7 = d_7 \succ d_9$,

e_3,

$f_3 = f_3$,

$g_1 \succ g_6 \succ g_7$,

$h_1 \succ h_4$.

Let F_i be the set containing all occurrences of the i^{th} object from each monotonic decreasing sequence of elements of M as follows:

$F_1 = [a_1, b_1, c_2, d_4, d_4, d_4, d_4, e_3, f_3, f_3, g_1, h_1]$,

$F_2 = [a_2, a_2, b_2, c_3, d_7, d_7, g_6, h_4]$,

$F_3 = [a_3, b_3, c_4, d_9, g_7]$,

$F_4 = [a_5, a_5, b_4, c_5, c_5]$,

$F_5 = [a_8, b_5, b_5]$,

$F_6 = [b_6]$,

$F_7 = [b_7]$.

The elements of F_i for each i are unordered except for equality (a trivial case of any ordering). Consider now, the following basic definition of ordering on $[F_i]$, a class of multisets with unordered elements.

Definition 4.5

Multiset ordering $\gg_R$ defined on the submultiset-based grid of a multiset.

Let E be a base set of the unordered multisets G and H, then $G \gg_R H$ if and only if $G \neq H$ and either or both of the following properties are satisfied:

i. $G(x) \not\prec H(x)$ for any x in E

ii. $x \prec y$ for any $x \in G$ and any $y \in H$ such that $x \neg\# y$.

Lemma 4.9

The collection $[F_i]$ is a decreasing sequence of submultiset of M with respect to $\gg_R$.

Proof

In fact, only the second property of Definition 4.5 suffices. This is obvious since to each object in F_i there exists a larger object in F_{i-1} with respect to the basic ordering $\succ$. □

Lemma 4.10

The ordering $\gg_{HO}$ is stronger than the ordering $\gg_R$.

Proof

Suppose $G \gg_R H$ holds where G and H are multisets whose elements are from a base set E. Consider the properties of $\gg_R$ on G and H. Property (i) implies $G(x) \geq H(x)\ \forall x \in E$. Since $G \neq H$, then the relation $G(x) = H(x)\ \forall x \in E$ does not hold, which implies $G \gg_{HO} H$. Property (ii) implies $x \geqslant y$ for any $x \in G$ and for any $y \in H$ such that $x \neg\# y$, thus $G(x) \geq H(x)$. Again, since $G \neq H$ holds then $G \gg_{HO} H$. Next, we show that multisets exist which are ordered by $\gg_{HO}$ but not by $\gg_R$. Consider the simple multisets [1,2,3] and [2,2,2]. Now the relation [1,2,3] $\gg_{HO}$ [2,2,2] is obvious, whereas the two multisets are incomparable under $\gg_R$. □

The above theorem makes it clear that $\gg_{HO}$ is stronger than $\gg_R$. However, $\gg_R$ sufficiently orders $[F_i]$ and will be used as the reference ordering for a given multiset. The ordering $\gg_{HO}$ will be used when the references involve at least two multisets. Note that $[F_i]$ is a permutation (an ordered sequence of elements with repetition allowed). The number of submultisets of *M* in $[F_i]$ equals the highest number of comparably unequal elements of *M*. In this illustration, the highest number of comparably unequal elements of M is the number of elements of $[b_i]$. This sequence may not be unique. We denote this number by α_M. The collection $[F_i]$ is

called *the submultiset-based grid* of M and each member of $[F_i]$ is called a *submultiset-based grid reference of M* (or simply, a *reference* of *M* or *M-reference*). Below is a formal definition of the concept.

Definition 4.6

Let $\preccurlyeq$ be a strict partial order defined on a set E and let M be a multiset of cardinality n over E. The permutation $[M_i]$ of submultisets $M_1, M_2, \ldots M_m$ of M is called the *submultiset-based grid* of M if the following properties are satisfied:

i. $M(x) = M_i(x)\ \forall x \in E, \forall i$ (*Whole submultiset property*. See Singh *et al.* (2007), for the definition of a 'whole submultiset' of a multiset).

ii. $\forall(x, y, i)[x \in M_i, y \in M_i] \Longrightarrow [(x = y)$ or $(x \# y)]$ (*Inordering property*). In order words, the elements of M_i are unordered.

iii. $\forall(i < j)[x \in M_j] \Leftrightarrow \exists y(y \in M_i)\ [x \prec y]$ (*Strict order property*).

If such properties exist, we say that M_j *precedes* M_i, and we denote this by $M_j \gg_R M_j$. Also, the *set* M_i is nonempty for all i. If no two elements of M are one strictly less than the other, then all the elements are contained in the only reference available, which is M.

The ordering $\gg_R$ is a *submultiset-based reference ordering* on the grid of the multiset M over E. Thus, the submultiset-based grid of M is the monotonic decreasing (with respect to the ordering $\gg_R$) sequences of submultisets of elements of M, in

which the i^{th} submultiset contains the i^{th} elements from each of all the longest possible monotonic decreasing sequences of comparably unequal objects of M. See submultiset-based partition and multiset ordering in Jouannaud and Lescanne (1982), for further details on the original variant of the construct.

4.2.2 Submultiset-based difference grid

Definition 4.7

Let $[M_i]$ and $[N_j]$ be the submultiset-based grids of the multisets M and N for $i = 1, 2, \dots, m$ and $j = 1, 2, \dots, n$, respectively. Let $p = \max\{m, n\}$ and let $k = 1, 2, \dots, p$. Similar to its set-based counterpart earlier presented, we construct the *submultiset-based difference grid* $[M_k, N_k]$ of M and N as follows:

i. $M_p \neq \emptyset$ or $N_p \neq \emptyset$,

ii. If $M_k \neq \emptyset$ and $N_k \neq \emptyset$ for a given k then $M_{k-1} \neq \emptyset$ and $N_{k-1} \neq \emptyset$.

The submultisets M_k are the submultiset-based references of M in the grid $[M_k, N_k]$ while N_k are the submultiset-based references of N in the grid $[M_k, N_k]$. If $m < n$ then $p = n$ and M_k is empty for $k = m + 1, m + 2, \dots, n$, and if $n < m$ then $p = m$ and N_k is empty for $k = n + 1, n + 2, \dots, m$. Unlike the references in the grid of a multiset, the references in a difference grid of two multisets are empty up to the number of references with which the grid with more references exceeds the grid with fewer references. Thus, the difference grid of two multisets is a collection of all the references from the individual grids of the multisets, where some references are

empty in the difference grid up to the number of references with which the grid with more references exceeds the grid with fewer references. We now prove the submultiset-based grid version of the pairwise equality theorem.

Theorem 4.11 (Submultiset-based pair-wise equality theorem for multiset)

Let M and N be multisets. $M = N$ iff $M_i = N_j$ for all $i = j$ where M_i and N_j are the respective submultiset-based references of M and N in the difference grid $[M_k, N_k]$ of M and N.

Proof

Let M and N be multisets over a domain set E and suppose $M_i = N_j$ for all $i = j$. By Definition 3.3, $M_i(x) = N_j(x)$ for all $x \in E$. By Property i of Definition 4.6 (whole-submultiset property), M_i is the only reference of M containing all occurrences of x in M. Similarly, N_j is the only reference of N containing all occurrences of x in N. Thus, $M_i(x) = M(x)$ and $N_j(x) = N(x)$. It follows that $M_i(x) = M(x)$ and $N_j(x) = N(x)$ for all $i = j$. Therefore, $M = N$. Conversely, let $M = N$ where M and N are multisets over a domain set E. Let M_i and N_j be the references of M and N, respectively. By Definition 3.3, $M(x) = N(x)$ for all $x \in E$. We claim $M_i = N_j \ \ \forall i = j$. Suppose the contrary, and let l_0 be the smallest positive integer for which the claim is not true. Either $M_{l_0} \lll_{HO} N_{l_0}$ or $N_{l_0} \lll_{HO} M_{l_0}$ or $M_{l_0} \ \# \ N_{l_0}$. If $M_{l_0} \lll_{HO} N_{l_0}$ holds, then by Definition 3.12, $M_{l_0} \neq N_{l_0}$ holds. Thus, $\exists a \in E$ such that $M_{l_0}(a) \neq N_{l_0}(a)$. By the whole-submultiset property, $M(a) \neq N(a)$. This contradicts the relation $M = N$.

Similarly, if $M_{l_0} \gg N_{l_0}$, we get a contradiction of the relation $M = N$. If $M_{l_0} \# N_{l_0}$ holds, then by Lemma 4.8 there exist incomparable objects u and v of E such that $M_{l_0}(u) > N_{l_0}(u)$ and $N_{l_0}(v) > M_{l_0}(v)$. It follows by the "Whole submultiset" property that $N \neq M$. Again, this is a contradiction of the relation $M = N$. Therefore, $M_i = N_j$ for all $i = j$. □

4.2.3 Grid approach to the Jouannaud-Lescanne's submultiset-based multiset ordering

Definition 4.8

Let M and N be multisets, then M precedes N in the sense of submultiset-based grid (written $M \ll_{PSM} N$ if and only if the following property is satisfied:

If $M_i = N_i$ for all i such that $M_i \neq \emptyset$ and $N_i \neq \emptyset$ then $M_j = \emptyset \ \forall j > i$; otherwise if $\exists i$ such that $M_i \neg\ll_{HO} N_i$ then $\exists j$ with $j < i$ such that $M_j \ll_{HO} N_j$, for all nonempty references M_i, N_i and N_j of the submultiset difference grid $[M_k, N_k]$ of M and N.

Theorem 4.12

The ordering $\ll_{PSM}$ is equivalent to the Jouannaud-Lescanne submultiset-based multiset ordering.

Proof

Let $\ll_{JL}$ be the Jouannaud-Lescanne submultiset-based multiset ordering (defined using *submultiset-based partition*) and let $\ll_{PSM}$ be the multiset ordering in

Definition 4.8. Suppose M and N are multisets such that $M \lll_{PSM} N$. We first show that $M \neq N$. By Property (i) of Definition 4.7, we have $M_p \neq \emptyset$ or $N_p \neq \emptyset$. If either M_p or N_p is nonempty then $M \neq N$. If both M_p and N_p are nonempty then by the 'if' part of Definition 4.8 or by the contrapositive of the 'if otherwise' part of Definition 4.8, $M_k \neq N_k$ for some k. Therefore, by Theorem 4.11, $M \neq N$. The monotonic non-increasing sequence of all the nonempty references of the grid of M from the difference grid $[M_k, N_k]$ is a submultiset-based partition $\overline{M}$ of M in lexicographic order, each submultiset containing only unordered elements. Conditions i, ii and iii of the submultiset-based multiset ordering are the same as those of $\lll_{JL}$. Let $M_i = N_i$ for all i such that $M_i \neq \emptyset$ and $N_i \neq \emptyset$ imply $M_i = \emptyset$. Then the number of sets in the partition of N is greater than the number of sets in the partition of M. Thus N is greater than M by the lexicographic ordering of $\lll_{JL}$. If $N_i \lll_{HO} M_i$ imply that $\exists j$ with $j < i$ such that $M_j \lll_{HO} N_j$, then again the partition of N is greater than the partition of M by the lexicographical ordering of $\lll_{JL}$. Conversely, suppose $M \lll_{JL} N$. Let p and q be the number of sets in the partitions of M and N respectively. Lexicographical extension of the ordering entails the following:

i. $M_i = N_i$ for all $i = j$ implies $p < q$.

ii. $\exists i$ such that $N_i \lll_{HO} M_i$ implies $\exists j < i$ such that $M_j \lll_{HO} N_j$.

In both cases, if we introduce difference grid then M_i and N_i are the references of M and N. Consider the repeated empty set $\emptyset, \emptyset, \dots, \emptyset$ ($j - i$ many times). Then $[M_i, \emptyset, \emptyset, \dots, \emptyset, N_j]$ is the difference grid of M and N. □

Theorem 4.13

The ordering $\lll_{PSM}$ is stronger than the Huet-Oppen multiset ordering

Proof

Let $M \lll_{HO} N$ hold. Suppose $\exists j$ such that $M_j \ggg N_j$. Then $\exists x$ such that $M_j(x) > N_j(x)$. By the whole submultiset property of Definition 4.6 $M(x) > N(x)$. By Huet-Oppen ordering $\exists y \succ x$ such that $N(y) > M(y)$. By Properties (ii) and (iii) of Definition 4.6 $\exists i < j$ such that $N_i(y) > M_i(y)$. Without loss of generality, we may assume y largest such elements. Hence, $N_i \ggg M_i$ holds, which implies $N \ggg_{PS} M$. Next, we show that the converse is not always true. Consider the multisets $M = \{2,1,1\}$ and $N = \{2, a\}$. It is easy to see that $M \lll_{PS} N$ holds since the submultiset-based reference $\{2, a\}$ of the grid of N is greater than either of the two references $\{2\}$ and $\{1,1\}$ of the grid of M. However, M and N are incomparable under $\lll_{HO}$ since there exists no element in N greater than all the elements in M whose multiplicity in N is greater than its multiplicity in M and vice versa. Therefore, Definition 4.8 is stronger than the Huet-Oppen multiset ordering. □

Theorem 4.14

The ordering $\ll_{PSM}$ is stronger than the Dershowitz-Manna definition.

Proof

Like their set counterparts, the proof of equivalence between Dershowitz-Manna submultiset-based and Huet-Oppen submultiset-based orderings can be found in Jouannaud and Lescanne (1982). Since by Theorem 4.13 Definition 4.8 is stronger than the Huet-Oppen definition, it follows by transitivity that Definition 4.8 is stronger than the Dershowitz-Manna definition. □

4.3 Some Results Holding for Multiset Ordering

In the results that follow, the order symbol $\ll$ is the Dershowitz-Manna multiset ordering. In many of the proofs, we shall use the Huett-Oppen equivalence of the ordering which is handier in the implementation of the ordering. As usual, the symbol $\prec$ represents the ordering that exists among the elements of a set or a multiset in general, while the symbol < is the usual ordering on the set of real numbers.

Lemma 4.15

Let E be a based set of multisets M, N and T such that $N(x) > M(x)$ where $x \in E$, then $[N \uplus T](x) > [M \uplus T](x)$.

Proof

Since $N(x)$ denotes the multiplicity of the object x in N, then $N(x)$ is a real number. Similarly, $M(x)$ is a real number. It follows that $N(x) + T(x)$ and $M(x) + T(x)$ are

real numbers since $T(x)$ is a real number for a multiset T. Since $N(x) > M(x)$ holds by the hypothesis, then $N(x) + T(x) > M(x) + T(x)$ holds by the properties of real numbers. By the definition of additive union, we have $[N \uplus T](x) > [M \uplus T](x)$. □

Lemma 4.16

Let E be a based set of multisets M, N and T such that $N(x) > M(x)$ where $x \in E$ and $T \subseteq M$, then $[N \backslash T](x) > [M \backslash T](x)$.

Proof

Suppose $T \subseteq M$. By the definition of submultiset $M(x) \geq T(x)$ for all x in E. By hypothesis, $N(x) > M(x)$ holds and $M(x)$, $N(x)$ and $T(x)$ are real numbers. By the transitivity property of real numbers, $N(x) > M(x) \geq T(x)$. It therefore follows from the properties of real numbers that $N(x) - T(x) > M(x) - T(x)$. Therefore, $[N \backslash T](x) > [M \backslash T](x)$ holds by the definition of multiset difference. □

Lemma 4.17

Let E be a based set of multisets M and N such that $N(x) > M(x)$ where $x \in E$, then $[N \backslash M](x) > [M \backslash N](x) = 0$.

Proof

Since by hypothesis $N(x) > M(x)$, then $N(x) + N(x) > M(x) + M(x)$. This implies $N(x) - M(x) > M(x) - N(x)$ by the properties of real numbers. In particular, $M(x) - N(x) \leq 0$. By the definition of multiset difference, $[M \backslash N](x) =$

$\max(M(x) - N(x), 0)$ which is 0 for the object $x \in E$. Hence, $[N \backslash M](x) > [M \backslash N](x) = 0$. □

Lemma 4.18

Let M, N and T be multisets such that $N \gg M$, where $\gg$ is the Huet-Oppen multiset ordering, then $(N \uplus T) \gg (M \uplus T)$.

Proof

Let E be a base set of M, N and T. Suppose $[M \uplus T](x) > [N \uplus T](x)$ for $x \in E$, then by definition of additive union $M(x) + T(x) > N(x) + T(x)$. That is $[M(x) + T(x)] - T(x) > [N(x) + T(x)] - T(x)$. It follows that $M(x) > N(x)$. By the definition of Huet-Oppen ordering, there exists y in E where $y \succ x$ such that $N(y) > M(y)$. This implies $[N(y) + T(y)] > [M(y) + T(y)]$. Again by the definition of additive union $[N \uplus T](y) > [M \uplus T](y)$. Hence, $[N \uplus T] \gg [M \uplus T]$. □

Lemma 4.19

Let M, N and T be multisets such that $N \gg M$ and $T \subseteq M$, where $\gg$ is the Huet-Oppen multiset ordering, then $(N \backslash T) \gg (M \backslash T)$.

Proof

Let E be a base set of M, N and T. Suppose $[M \backslash T](x) > [N \backslash T](x)$ holds for an object $x \in E$, then by the definition of multiset difference, $M(x) - T(x) > N(x) - T(x)$. Thus, $[M(x) - T(x)] + T(x) > [N(x) - T(x)] + T(x)$. This implies $M(x) >$

$N(x)$. By the definition of Huet-Oppen ordering, there exists $y \in E$ where $y \succ x$ such that $N(y) > M(y)$. This implies $[N(y) - T(y)] > [M(y) - T(y)]$, and by the definition of multiset difference $[N \backslash T](y) > [M \backslash T](y)$. Hence, $[N \backslash T] \gg [M \backslash T]$.

□

Lemma 4.20

Let M, N and T be multisets such that $N \gg M$ and $N \subseteq T$, where $\gg$ is the Huet-Oppen multiset ordering, then $(T \backslash M) \gg (T \backslash N)$.

Proof

Let E be a base set of M, N and T. Suppose $[T \backslash N](x) > [T \backslash M](x)$ holds for an object $x \in E$, then $T(x) - N(x) > T(x) - M(x)$ holds by the definition of multiset difference. This implies $[T(x) - N(x)] + T(x) > [T(x) - M(x)] + T(x)$. Hence, $M(x) > N(x)$. By the definition of Huet-Oppen ordering, there exists $y \in E$ where $y \succ x$ such that $N(y) > M(y)$. This implies $-M(y) > -N(y)$, which implies $[T(y) - M(y)] > [T(y) - N(y)]$ by the properties of real numbers. Subsequently, we get $[T \backslash M](y) > [T \backslash N](y)$ by multiset difference. Therefore, $[T \backslash M] \gg [T \backslash N]$.

□

Lemma 4.21

Let M, N and T be multisets such that $N \gg M$, where $\gg$ is the Huet-Oppen multiset ordering, then $[N \backslash M] \gg [M \backslash N]$.

Proof

Let E be a base set of M, N and T. Suppose $[M\backslash N](x) > [N\backslash M](x)$ for $x \in E$, then x is necessarily in $[M\backslash N]$. This implies $M(x) > N(x)$. Since $N \gg M$ where $\gg$ is the Huet-Oppen ordering then there exists $y \in E$ where $y \succ x$ such that $N(y) > M(y)$ and $M \neq N$. By Lemma 4.17, $[N\backslash M](y) > [M\backslash N](y)$, hence $[N\backslash M] \gg [M\backslash N]$. □

Lemma 4.22

Let M, N, S and T be multisets such that $N \gg M$ and $T \gg S$, then $N \uplus T \gg M \uplus S$.

Proof

By Lemma 4.18, $N \uplus T \gg M \uplus T$ and $M \uplus T \gg M \uplus S$. It follows by transitivity of the ordering that $N \uplus T \gg M \uplus S$. □

Lemma 4.23

Let M, N, S and T be multisets such that $N \gg M$ and $T \gg S$. If $S \subseteq M$ and $T \subseteq N$, then $N\backslash S \gg M\backslash T$.

Proof

By Lemma 4.19, $N\backslash S \gg M\backslash S$ and by Lemma 4.20, $M\backslash S \gg M\backslash T$. It follows by transitivity of the ordering that $N\backslash S \gg M\backslash T$. □

Next, we present an alternative proof of the equivalence between the Dershowitz-Manna ordering and the Huet-Oppen ordering (Peter and Singh, 2017a).

Theorem 4.24

The Dershowitz-Manna multiset ordering and the Huet-Oppen multiset ordering are equivalent.

Proof

Let $\ll_{DM}$ and $\ll_{HO}$ be the Dershowitz-Manna and Huet-Oppen orderings, respectively. Assume $M \ll_{HO} N$ holds and define X and Y as follows: $X = M\backslash N$ and $Y = N\backslash M$, thus $Y \subseteq N$ is clear. We claim $Y \neq \emptyset$. Suppose the contrary, that is $N\backslash M = \emptyset$. Either $N \subset M$ or $x \in M\backslash N$ for some $x \in E$. The former is a contradiction of $M \ll_{HO} N$ since multiset inclusion satisfies the ordering as well. If the latter is the case then $M(x) > N(x)$ and by $\ll_{HO}$ $\exists y \in E$ such that $N(y) > M(y)$. This implies $y \in N\backslash M$ – a contradiction of the assumption $Y = \emptyset$. To prove (ii), we consider that $M\backslash X = M\backslash[M\backslash N]$ and $N\backslash Y = N\backslash[N\backslash M]$. But $M\backslash[M\backslash N] = M \cap N = N\backslash[N\backslash M]$. Thus, we have $M\backslash X = N\backslash Y$. By definition $X \subseteq M$, hence $M = [N\backslash Y] \uplus X$. To prove (iii), let $x \in X$, thus $x \in M$. If $x \in N$ $\forall x$ then $M \subseteq N$ and the result follows. If $x \notin N$ then $x \in [M\backslash N]$ implies $M(x) > N(x)$. By hypothesis, $\exists y \succ x$ such that $N(y) > M(y)$. Hence, $y \in [N\backslash M]$ this implies $y \in Y$. Conversely assume $M \ll_{DM} N$. From (ii) $M\backslash X = N\backslash Y$, and from (iii), $X \neq Y$ implies $M \neq N$. Assume $M(x) > N(x)$. From (ii), $M(x) = N(x) - Y(x) + X(x)$. We have $X(x) - Y(x) > 0$, thus $x \in X$. From (iii) $\exists y \succ x$ such that $y \in Y$, thus $Y(y) > X(y)$. Moreover, $M(y) = N(y) - Y(y) + X(y)$ from (ii) implies $N(y) - M(y) > 0$. Hence, $N(y) > M(y)$ □

4.4 Some Extensions of Set-based Ordering to Simple Multisets of Incomparable Objects

It seems convincing that there exists an ordering on simple multisets of incomparable objects by means of the Jouannaud-Lascanne set-based multiset ordering. In this thesis, an attempt to show that a singleton multiset is dominated by a simple pair multiset in spite of the incomparability of their objects is made with respect to the ordering. Furthermore, an existence of the ordering among simple multisets of higher cardinalities where the cardinality of the preceding multiset is a unit or more less than that of the succeeding multiset is observed. Thus, we obtain an extension of the ordering to simple multisets of incomparable objects (Peter and Singh, 2017c). We begin with a simple corollary, which establishes a relationship between pair sets and simple multisets of finite cardinalities with respect to set-based multiset ordering.

Lemma 4.25

Every pair set dominates a finite simple multiset with respect to set-based ordering where the set and the multiset have a common object.

In other words, if S is a pair set and M is a simple multiset such that an object a belongs to $S \cap M$, then S dominates M with respect to set-based ordering.

Proof

Let S be the set $\{a, b\}$ and let M be a simple multiset of an object a. Since, $a \in S \cap M$, then a belongs to each of the first set-based references of S and M as illustrated in the following table:

Table 4: The set-based difference grid of the set $\{a, b\}$ and the simple multiset $\{a, a, \dots, a\}$

$\{a, b\}$	$\emptyset$	…	$\emptyset$
$\{a\}$	$\{a\}$	…	$\{a\}$

Since the first and only nonempty reference of S dominates the first reference of M with respect to the Huet-Oppen ordering, then S dominates M by the set-based ordering defined above (or by set inclusion). □

The two multisets $\{e, e\}$ and $\{a\}$ cannot be shown to be comparable using the conventional set-based approach. In fact their difference grid has no basis for comparability. To see this, consider placing them in a difference grid in the following table:

Table 5: The supposed set-based difference grid of $\{e, e\}$ and $\{a\}$

$\{e\}$	$\{e\}$
$\{a\}$	$\emptyset$

For two multisets to be comparable, one reference must be greater than the other in the last unequal pair of references in their difference grid. Such is not the case here. Thus, the method fails to determine the dominance between the multisets. The ordering between such multisets is better established in the following theorem.

Theorem 4.26 (The harsh–simple theorem on multiset ordering)

Every simple pair multiset dominates a singleton multiset with respect to set-based ordering where both multisets have incomparable objects.

In other words, if e and a are incomparable objects with respect to a basic ordering $\succ$, then the relation $\{e, e\} \gg \{a\}$ holds where $\gg$ is the set-based multiset ordering.

Proof

Choose multisets $M = \{a, a, e\}$, $N = \{a, a, e, e\}$ and $X = \{a, a, a, e\}$. Now N is greater than M in the sense of set-based ordering. To see this, consider the difference grid of M and N in the following table.

Table 6: The difference grid of $\{a, a, e\}$ and $\{a, a, e, e\}$

$\boldsymbol{N}$	$\{a, e\}$	$\{a, e\}$
$\boldsymbol{M}$	$\{a, e\}$	$\{a\}$

Thus, the relation $N \gg M$ holds since $\{a, e\} \gg_{HO} \{a\}$ holds from the second pair of references of M and N. Similarly, the relation $N \gg X$ holds. Consider the following multiset operations:

$$N \gg M$$

$$N \uplus \{a, e\} \gg M \uplus \{a, e\}$$

$$N \uplus \{a, e\} - \mathrm{X} \gg M \uplus \{a, e\} - N$$

$$\{e, e\} \gg \{a\}$$

hence, the proof. □

It follows that the set-based ordering has a possible extension and could be more powerful than one thought. Similar to the multisets in Theorem 4.26, the set-based ordering between the multisets $\{e, e, e, e\}$ and $\{a, a, a\}$ cannot be determined using the conventional difference grid approach. Consider then the following theorem, which is an extension of Theorem 4.26.

Theorem 4.27 (The first extended harsh-simple theorem)

Every simple multiset dominates another simple multiset with respect to set-based ordering where both multisets have incomparable objects and the cardinality of the preceding multiset is a unit less than that of the succeeding multiset.

In other words, if e and a are incomparable objects with respect to a basic ordering $\succ$, then the relation $\{e, e, e, \dots, e\}$ ($n + 1$ times) $\gg \{a, a, a, \dots, a\}$ (n times) holds, where $\gg$ is the set-based multiset ordering.

Proof

The proof is by induction on n. The statement is true for $n = 1$, since $\{e, e\} \gg \{a\}$ by Theorem 4.26. Suppose it is true for k. That is,

$$\{e, e, e, \dots, e\}(k + 1 \text{ times}) \gg \{a, a, a, \dots, a\}(k \text{ times}) \qquad (2)$$

We need to show from (2) that it is true for $n = k + 1$. Now the relation $\{a, e\} \gg \{a, a\}$ holds by Lemma 4.25. Taking the additive union of $\{a, e\}$ with the L.H.S of (2) and that of $\{a, a\}$ with the R.H.S of (2) we obtain:

$$\{a, e\} \uplus \{e, e, e, \dots, e\}\ (k + 1 \text{ times}) \gg \{a, a, a, \dots, a\}\ (k \text{ times}) \uplus \{a, a\}.$$

This implies

$$\{e, e, e, \dots, e\}(k + 2 \text{ times}) \uplus \{a\} \gg \{a, a, a, \dots, a\}\ (k + 2 \text{ times}).$$

Subtract $\{a\}$ from both sides to get

$$\{e, e, e, \dots, e\}\ (k + 2 \text{ times}) \gg \{a, a, a, \dots, a\}\ (k + 1 \text{ times}).$$

That is,

$$\{e, e, e, \dots, e\}\ ((k + 1) + 1 \text{ times}) \gg \{a, a, a, \dots, a\}\ (k + 1 \text{ times}).$$

It follows that

$$\{e, e, e, \dots, e\}\ (n + 1 \text{ times}) \gg \{a, a, a, \dots, a\}\ (n \text{ times}).$$

Since being true for $n = k$ implies being true for $n = k + 1$, then the statement is true for all $n \in \mathbb{N}$. □

We take the extension further to multisets where the cardinality of the succeeding multiset is more than a unit greater than the cardinality of the preceding multiset in Theorem 4.28.

Theorem 4.28 (The second extended harsh-simple theorem)

Every simple multiset dominates another simple multiset with respect to set-based ordering where both multisets have incomparable objects and the cardinality of the preceding multiset is greater than that of the succeeding multiset.

In other words, if e and a are incomparable objects with respect to a basic ordering $\succ$, and m and n are positive integers, then the relation $\{e, e, e, \dots, e\}$ ($m + n$ times) $\gg$ $\{a, a, a, \dots, a\}$ (m times) holds, where $\gg$ is the set-based multiset ordering.

Proof

The proof is by induction on n, where n is the number by which the multiplicity of the elements of $\{e, e, e, \dots, e\}$ exceeds the multiplicity of the elements of $\{a, a, a, \dots, a\}$. The statement is true for $n = 1$ since $\{e, e, e, \dots, e\}$ ($m + 1$ times) is greater than $\{a, a, a, \dots, a\}$ (m times) by Proposition 4.3. Suppose the statement is true for k. That is,

$$\{e, e, e, \dots, e\} \ (m + k \text{ times}) \gg \{a, a, a, \dots, a\} \ (m \text{ times}). \tag{3}$$

Consider the relation $\{e\} \gg \emptyset$ and add $\{e\}$ to the L.H.S. and $\emptyset$ to the R.H.S. of (3).

$$\{e\} \uplus \{e, e, e, \dots, e\} \ (m + k \text{ times}) \gg \emptyset \uplus \{a, a, a, \dots, a\} \ (m \text{ times}).$$

This implies

$$\{e, e, e, \dots, e\} \ (m + k + 1 \text{ times}) \gg \{a, a, a, \dots, a\} \ (m \text{ times}).$$

It follows that

$$\{e, e, e, \dots, e\} \ (m+n \text{ times}) \gg \{a, a, a, \dots, a\} \ (m \text{ times}).$$

where n is the number by which the multiplicity of the elements of $\{e, e, e, \dots, e\}$ exceeds the multiplicity of the elements of $\{a, a, a, \dots, a\}$. □

For the purpose of this thesis and possible future use, we shall denote the harsh-simple ordering by $\gg_{\#}$ and denote the extended harsh-simple orderings by $\gg_{1\#}$ and $\gg_{2\#}$, accordingly.

4.5 The Concept of 'Ideal' Ordering

4.5.1 Ideal ordering (a consequence of the properties of partial ordering)

It is a natural property of comparability relations that objects to be compared be of the same category as far as such comparability is concerned. For example, there exists no comparability between weight measure and length measure. That is, although the pure number 35 is less than the pure number 50 we cannot conclude that the quantity 35 grams is less or equal or even greater than the quantity 50 metres. This is because grams and meters are different "categories" of measurement. We say that 35 grams and 50 meters are incomparable with respect to the usual ordering < on the set of real numbers, symbolically written as "*35 grams # 50 metres*". In general, if two quantities A and B satisfy similar behaviour as 35 grams and 50 metres above (that is A and B are incomparable with respect to the ordering $\prec$), we denote such phenomenon by $A \# B$ following the definition of incomparability relation. However,

it is possible to find a third quantity C for which both $C \succ A$ and $C \succ B$ hold or both $C \prec A$ and $C \prec B$ hold. The ordering is a restriction of the properties of partial ordering to include only the aforementioned ones in addition to its restriction to only three distinct elements. Hence, it has attracted the broader name of *property-restricted partial ordering* or *ideal ordering* as is used in this thesis. See Peter *et al.* (2017d) for further details. We present a formal definition of the concept in the definition that follows.

It may be emphasized that there could be the following possible order relations between two quantities a and b, viz: $a \prec b,\ a \succ b,\ a \# b\ a = b$ and $a\ ?\ b$, where $\prec$ or $\succ$ refers to the ordering, # is incomparability, and "?" stands for *unknown*.

Definition 4.9

A partial order $\preccurlyeq$, defined on a set S is called an *ideal partial order* (or simply an *ideal order*) on S if and only if there exists a pair of incomparable elements x and y in S and an element $a \in S$ such that either $a \prec x$ and $a \prec y$ or $x \prec a$ and $y \prec a$.

In other words, there exist at least two incomparable elements of S that are both less than or greater than at least one element of S. If such a set S exists, we say that $\preccurlyeq$ is a *property-restricted partial order on* S, and S is a *property-restricted set* with respect to $\preccurlyeq$. For the purpose of clarity, we shall call a partially ordered set that is not an ideal set an *ordinary partially ordered set*.

Definition 4.10

An ideal set S over the ordering $\preccurlyeq$ is called an *isosceles set* if and only if for any two elements x and y of S such that $x \# y$ holds, there exists an element $a \in S$ such that $a \prec x$ and $a \prec y$ or $x \prec a$ and $y \prec a$ exist.

In other words, any two incomparable elements of S are both less than or greater than at least one other element in S. If such a relation exists, we say that S is *isoscelesly ordered* by $\preccurlyeq$, and we call S an *isosceles set* with respect to $\preccurlyeq$. The smallest subset T of S satisfying this property is an *isosceles subset* of S consisting of three elements. We call it an *isosceles triple*.

4.5.2 The structure of an isosceles triple

An *isosceles triple* is best pictured in a diagram consisting of two eccentric isosceles triangles. Such a diagram contains elements between the apexes of the triangles. The order symbol is written between the edges on both sides of the triangles, while the incomparability symbol is written between the bases of the triangles. Consider the isosceles triple $A = \{a, b, c\}$ where $b \# c$ is such that $a \succ b$ and $a \succ c$ and the isosceles triple $X = \{x, y, z\}$ where $y \# z$ is such that $x \prec y$ and $x \prec z$. Figure 6 illustrates the isosceles triple of A (with a *peak*) while Figure 7 illustrates the isosceles triple of B (with a *trough*). In other words, a is the peak of the isosceles triple illustrated in Figure 6 while x is the trough of the isosceles triple illustrated in Figure 7.

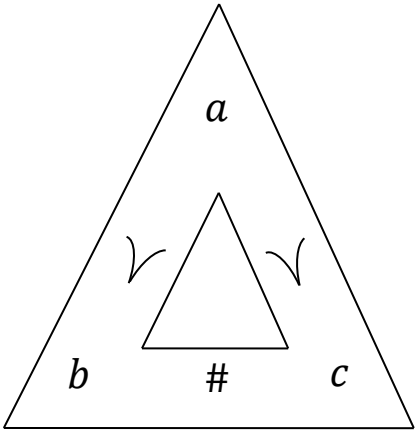


Figure 6: The Structure of the Isosceles Triple of A (with a Peak)

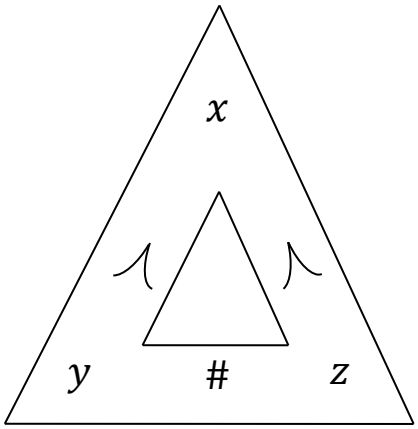


Figure 7: The structure of the isosceles triple of X (with a trough)

The elements b, c, y and z are called *base elements* of A.

Theorem 4.29

The class of total ordering and ideal ordering are disjoint.

Proof

Let $\prec_T$ be a total order defined on a set T. By definition $\prec_T$ is a partial order on T and for all element u and v of T, either $u \prec_T v$ or $v \prec_T u$. It follows that there exists no relation of the form $x \#_T y$ where $\#_T$ is incomparability with respect to $\prec_T$. Therefore, no three elements of T satisfy the ideal order property. Consequently, $\prec_T$ is not an ideal order. On the other hand, let $\prec_I$ be an ideal order on a set S. By definition, $\prec_I$ is a partial order on S and there exists a pair of elements x and y where $x \# y$ such that either $a \prec_I x$ and $a \prec_I y$ or $x \prec_I a$ and $y \prec_I a$ in the strict sense of the order, where a is an element in S. Thus, statements equivalent to 'either $x \prec_I y$ or $y \prec_I x$ for all x and y in S' do not hold. Therefore $\prec_I$ is not a total order on S. □

Corollary 4.30

Total ordering and ideal ordering are incomparable.

Proof

It is sufficient to present two examples, each satisfying the characteristics of either type of multiset orderings. The set $\mathbb{N}$ of natural numbers is totally ordered by the usual ordering, but not *ideally* ordered since no two natural numbers are incomparable with respect to the usual ordering. On the other hand, the set $\{\{n\}: n \in \mathbb{N}\} \cup \{\{m, n\}: m, n \in \mathbb{N}\}$ of singletons and pairs of natural numbers defined by the containment ordering $\subset$ is ideally ordered but not totally ordered. □

4.5.3 Examples of ideal ordering/set

Example 4.1

An obvious example of an ideal ordering in Mathematics (specifically in set theory) is the ordering defined by the strict containment relation ($\subset$) on multisets where # is interpreted to mean $\neg\{\subset, \supset, =\}$. For instance, let $A = \{a\}$, $B = \{a, b\}$, and $C = \{a, c\}$. Now $A \subset B, A \subset C$ and $B \# C$. We represent this with a Venn diagram as follows:

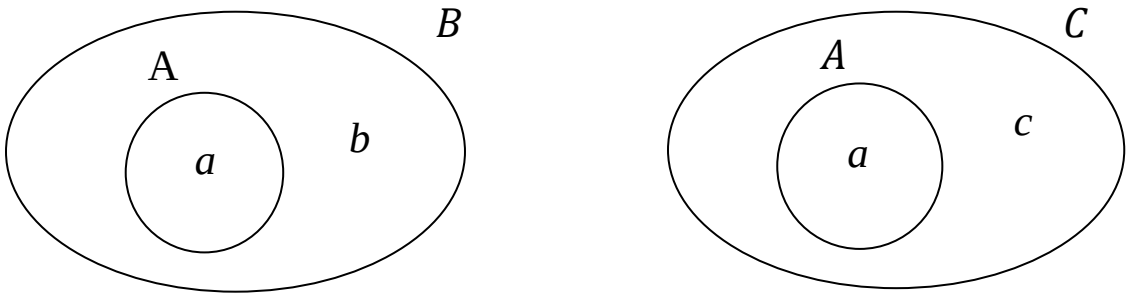


Figure 8: Venn diagrams illustrating set examples of ideal ordering with a trough

On the other hand, let $A = \{b, c\}, B = \{b\}$ and $C = \{c\}$. Now, $B \subset A, C \subset A$ and $B \# C$.

The diagram is as follows:

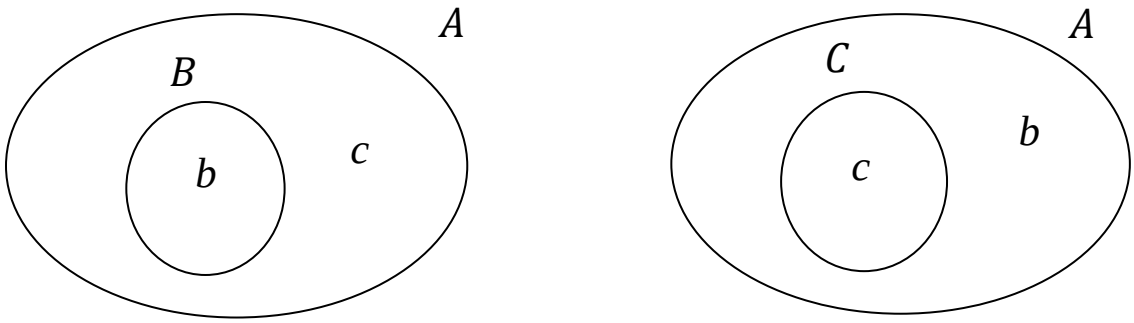


Figure 9: Venn diagrams illustrating set examples of ideal ordering with a peak

This example extends to sets of higher cardinalities as well.

Example 4.2

It has been shown that Darshowitz-Manna multiset ordering implies set-based multiset ordering and submultiset-based multiset ordering. It has also been shown that set-based multiset ordering and submultiset-based multiset ordering are incomparable. Consider therefore the ordering defined by 'implication' ($\Longrightarrow$) on the set $O' = \{\lll_{\mathcal{S}}, \lll_{\mathcal{M}} \lll_{DM}\}$ of types of orderings on multisets where $\lll_{\mathcal{S}}$, $\lll_{\mathcal{M}}$ and $\lll_{DM}$ are the set-based, submultiset-based and Darshowitz-Manna multiset orderings, respectively and # is defined by $\neg\{\Longrightarrow, \Longleftarrow, \equiv\}$. Thus, the ordering defined by $\Longrightarrow$ is an ideal ordering and O' is an isosceles triple.

Example 4.3

Since total ordering is partial ordering in which any pair of elements in the set under consideration are comparable, then total ordering implies partial ordering. From the definition of ideal ordering above, every ideal ordering is a partial ordering. Moreover, total ordering and ideal ordering are incomparable from Corollary 4.30. Consider then, the ordering defined by 'implication' ($\Longrightarrow$) on the set $O'' = \{\prec_T, \prec_I \prec_P\}$ of classes of orderings on sets, where $\prec_T$, $\prec_I$ and $\prec_P$ are total order, ideal order and partial order, respectively, where # is defined by $\neg\{\Longrightarrow, \Longleftarrow, \equiv\}$. Thus, the ordering defined by $\Longrightarrow$ is an ideal ordering and the set O'' is an isosceles triple.

Example 4.4

Considering the harsh-simple multiset ordering (Theorem 4.26), given two incomparable objects say e and a, the multiset $\{e, e\}$ is greater than the multiset $\{a\}$ where the ordering in question is the set-based multiset ordering. Similarly, the multiset $\{a, a\}$ is greater than the multiset $\{e\}$. This ordering is called *#-simple* (pronounced, hash-simple) *multiset ordering*. It is easy to see that the hash-simple multiset ordering is an ideal ordering, where # is defined by $\neg\{\gg_{\#}, \ll_{\#}, =\}$. To see this, consider the simple multisets, $\{a\}$, $\{e, e\}$ and $\{f, f\}$ where $a \# e$, $a \# f$ and $e \# f$ hold. Following the definition of $\gg_{\#}$, the relations $\{e, e\} \gg_{\#} \{a\}$ and $\{f, f\} \gg_{\#} \{a\}$ hold. Moreover, the relation $\{e, e\} \# \{f, f\}$ holds. Similarly, the relations $\{e, e\} \gg_{\#} \{a\}$, $\{e, e\} \gg_{\#} \{f\}$, and $\{a\} \# \{f\}$ hold.

Example 4.5

Let U be an *unordered set* (a set which has no ordering defined on it). Let T be a set of tuples of elements of U and let S be a set of simple multisets of elements of U which have equal cardinalities. The *set-based* multiset ordering $\ll_{setM}$ defined on $T \cup S$ is an ideal ordering and $T \cup S$ is an ideal set, where # is defined by $\neg\{\ll_{setM}, \gg_{setM}, =\}$. As a simple illustration, suppose T has only one element viz, $T = \{\{a, f\}\}$ where $a \# f$ holds and let $S = \{\{a, a, a, a\}, \{f, f, f, f\}\}$. Now, $T \cup S = \{\{a, f\}, \{a, a, a, a\}, \{f, f, f, f\}\}$ and the relations $\{a, a, a, a\} \ll_{setM} \{a, f\}$ and $\{f, f, f, f\} \ll_{setM} \{a, f\}$ hold from Lemma 4.25. Also, the relation $\{a, a, a, a\} \# \{f, f, f, f\}$ holds.

Example 4.6

Consider the parenthood relation ($\ll_{parn}$) defined by *parent* as follows: the relation $A \gg_{parn} B$ holds if and only if A is parent of B. Notice that if A is a parent of B and A is a parent of C then B cannot be a parent of C and C cannot be a parent of B. Since $B \neq C$, then $B \# C$ holds where # is defined by $\neg\{\gg_{parn}, \ll_{parn}, =\}$. Therefore A, B and C form an isosceles triple and the ordering $\gg_{parn}$ is an ideal ordering.

Example 4.7

Consider the relation $\ll_{ance}$, defined by *ancestor* as in the following illustration. $X \gg_{ance} Y$ if and only if X is an ancestor of Y. Notice that if X is an ancestor of Y and X is an ancestor of Z, then Y cannot be an ancestor of Z and Z cannot be an ancestor of X. Thus, Y and Z are incomparable under the relation $\gg_{ance}$. Therefore X, Y and Z form an isosceles triple and $\gg_{ance}$ is an ideal ordering.

Example 4.8

A typical real life example of ideal ordering (typically in chemistry) is the composition ordering among chemical substances. If two chemical substances (called reactants) combine to form a new chemical substance, the resultant compound is said to be a *product* of the two reactants. One says that the product is composed of the reactants. We therefore define the ordering between reactants and their product by the 'composition' relation denoted by $\gg_{comp}$. If the two reactants are different and no reactant is composed of the other, then the two reactants are said to be incomparable.

In this case the three substances form an isosceles triple with respect to the composition order $\gg_{comp}$, where $\# = \neg\{\gg_{comp}, \ll_{comp}, =\}$.

Example 4.9

In a chemical reaction, Chemical bonds in the reacting molecules are broken; this takes in energy. New chemical bonds form to make the products; this gives out energy. In a physical reaction (physical change), molecules are rearranged but their internal structures are not affected, and no energy is given off in the process. Both types of processes can be seen to be different. In fact, the two are incomparable. A third type of reaction called the null reaction (explained later in the thesis) together with the two reactions form an isosceles triple with respect to the reaction ordering.

4.5.4 The structure of an ideal set

The following is a diagrammatic structure of a typical ideal set. In the illustration, the arrows "$\longrightarrow$" indicate $\succ$ while $\Longrightarrow$ indicate $\succcurlyeq$. Note the incomparability relation # between two objects.

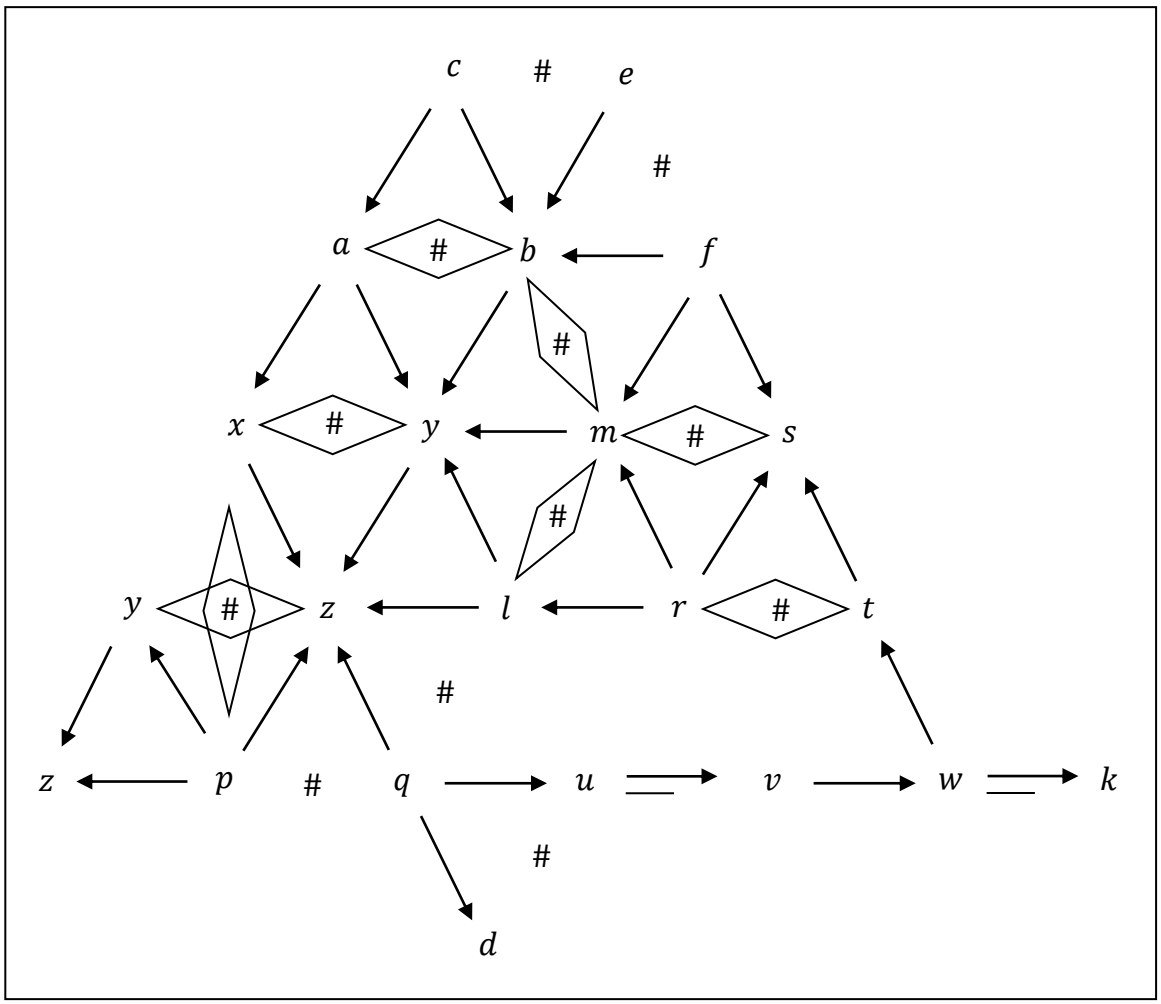


Figure 10: The structure of a typical ideal set

CHAPTER FIVE

AN APPLICATION OF IDEAL ORDERING TO ALTERNATIVE P SYSTEMS

5.0 Introduction

Membrane computing is a distributed and parallel computing model introduced in Păun (2000) whose aim is to abstract computing ideas from the structure and functions within living cells and the way they interact with one another or with their environment in a hierarchy of *membranes* called *membrane structure*. In a typical membrane structure, *multisets* of objects are processed by *rules* which act upon them both of which are present in the *regions* of the membranes together with pre-specified input and output membranes.

The ability of the membrane structure to *evolve* by the action of the rules on the objects gives rise to a computing device called P system. Of the several variants that have emerged in the literature, we shall present an improvement of the P system working in the accepting mode introduced in Freund and Păun (2003) where an input is provided in the form of a number and the system accepts this number or not, provided that a halting computation exists starting from the initial configuration. The input is introduced in the system in the initial configuration, in the form of the multiplicity of a specified object.

The P system introduced is of *specialized* type, based on a peculiar consequence of partial ordering on multisets known as *ideal ordering*. Such an

ordering is basically identified by three objects, a partial order on multisets and the accompanying incomparability relation as we shall see in subsequent sections.

An important characteristic that gave rise to the variant of P system is the way in which the functions of the rules are specialized. Much as the biochemical activities within a living cell may be classified into their respective chemical, physical and mixed forms, the rules representing these biological activities are specialized to separately carry out their functions in these forms.

In the regions delimited by the membranes are placed rules representing chemical processes, physical processes and mixtures among other rules which are said to function as background processes. The objects, which are also placed in the regions, represent chemical compounds. As is typical of most variants of P system, these objects are said to *evolve* – they can be transformed from one object to another and can combine to form another object. We advise the reader to refer to Păun (2000) and Păun (2006) for elaborate details on P systems.

It is observed that with only three rules a nondeterministic alternative P system based on ideal ordering has computational completeness, it can characterize the recursively enumerable sets of natural numbers. It is interesting to note that very simple membrane structures are enough: only one membrane suffice. The method we use to determine computational completeness is of the same type as that used in the

accepting P system provided in Freund and Păun (2003), only that this time the rules are by far fewer in number.

5.1 A Biochemical Interpretation of the Rules

The idea is to present a model of the biochemical processes taking place in the cell by means of multisets of objects. The changes in the substances involve chemical properties for chemical change as well as for physical change. The processes to consider include mixture, reversible chemical reaction, physical change, catalytic reaction, derived reaction and the null reaction. In modelling the rules, the alphabets (Greek or English) represent the chemical substances, the binary operation '+' represents the action 'reaction' or 'change' as the case may be. The arrow '→' points to the result of the reaction or change process, while the arrow '⟹' is used where such changes do not take place. Moreover, the use of the symbol '∅' is essential and is interpreted to mean that there exists a particular substance which has not reacted with another substance after they have been combined – a case of mixture, hence its use in conjunction with '⟹'.

5.1.1 Mixture

One would think of representing mixture by $\alpha + \beta \Longrightarrow \beta + \alpha$. The thought of such a representation is probably with the intention that two substances combine to yields the two substances as they were, but mixed. This however, is crude and inappropriate to our model. The idea is to model the process in reaction terms by $\alpha + \emptyset \Longrightarrow \beta$, and this presents a more logical concept of the representation of mixture. It means 'nothing'

has reacted with α and we can still obtain β. Moreover, α cannot change to β unless a second element (or energy) is involved. Hence, the symbol $\emptyset$ denotes the absence of such substance or energy. Furthermore, $\emptyset$ cannot be omitted to avoid the impression that α underwent a unilateral change or reaction to the substance β which is not a property of a biochemical process. The idea is to have the model as applicable as possible yet retain biochemical outlook.

5.1.2 Null reaction

The *null* or *trivial* reaction does not mean the absence of a reaction even though it does not model any reaction. It means the presence of a chemical environment where the potential for reactions to take place exists. In fact, the null reaction is the most essential of all reactions as it always exists. It is modeled by $\alpha + \emptyset \Longrightarrow \alpha$. It directly translates as 'only one substance is involved and again we obtain the only substance.' Furthermore, any substance which withstands a chemical or physical environment is said to have undergone a null reaction. It is easy to see that the null reaction is trivial to all other reactions. Particularly obvious (as seen from the models representing them) is its triviality to mixture. The difference between the representation of mixture and that of null reaction is that mixture involves more than one substance whereas the null reaction involves only one substance before and after a given period of time.

5.1.3 Reversible chemical reaction

Chemical reaction is modelled by $\alpha + \beta \underset{rev}{\longrightarrow} \gamma$, meaning that two substances α and β react to produce the resultant substance γ. The process can be reversed under certain conditions as indicated by the rev below the arrow.

5.1.4 Catalytic reaction

We consider a catalytic case $\alpha + \xi \rightarrow \beta + \xi$ of the chemical reaction, where ξ is a catalyst. We do not lay emphasis on the other element which reacts with α for which ξ is catalyst to the reaction of such element and α. The model is so since by the end of the reaction at least a new substance is produced – not minding the number of substances used. In other words, the state of the catalyst is the focus for the catalytic case – it is not consumed in the process.

5.1.5 Physical change

Physical change is modelled by $\alpha + \beta \rightarrow \alpha$. This captures the idea that α and β combine via the change process '+' which leaves the chemical properties of α unchanged, much as water transforms into water vapor by heating. Thus, again the change is in terms of the chemical properties of the substance – leaving the chemical properties unchanged. This process is reversible and the reverse is $\alpha \rightarrow \alpha + \beta$, similar to separating heat from water vapor to produce heat and water.

5.1.6 Incomparability with respect to reaction

If two substances cannot be combined in any of the processes above, then the substances are considered as not present in a chemical or physical environment. Such a scenario is modelled by $\alpha \# \beta$ where α and β are the substances.

5.1.7 An approach for non-determinism

It is possible to have a situation where an object α is combined with β to produce β in the sense of physical change or to produce $\gamma + \alpha$ in the sense of catalytic reaction, which indicates non-determinism in the system. Thus, both reactions cannot occur simultaneously. Same can be said about other rules.

5.1.8 Derived reactions

It is possible to derive other reaction types from a combination of some of the reactions already mentioned. A typical example of a *derived reaction* is the resultant reaction that converts the initial substance to the final substance in a series of reactions. That is, if for instance, a substance a changes to b by a reaction $a \rightarrow b$ and b changes to c by $b \rightarrow c$ then the derived reaction that changes a to c is $a \rightarrow c$. In particular, by the use of reversible chemical reactions, catalytic reactions and physical change and by employing the concept of property-restricted ordering, we derive a typical reaction which is of interest to the model of P system under investigation in the following proposition.

Proposition 5.1 *Given a finite number* $n-1$ *of catalytic reactions* $f_1 + a \rightarrow f_2 + a, \ldots, f_{n-1} + a \rightarrow f_n + a$, *a reversible chemical reaction* $f_1 + a \underset{rev}{\longrightarrow} f_n$ *and the physical change* $f_1 + a \rightarrow f_1$, *there exists a reaction of the form* $f_1 \rightarrow f_n + a$ *up to ideal ordering.*

Proof

Since the finite number $n-1$ of catalytic reactions $f_1 + a \rightarrow f_2 + a, \ldots, f_{n-1} + a \rightarrow f_n + a$ hold by the hypothesis, by transitivity $f_1 + a \rightarrow f_n + a$ is a derived reaction. Moreover, the physical change $f_1 + a \rightarrow f_1$ also holds by the hypothesis. One of four cases must hold. That is, either f_n with a is incomparable with f_1 (i.e., $f_1 \# f_n + a$) or f_n reacts with a to produce f_1 (i.e., $f_n + a \rightarrow f_1$) or f_1 produces f_n with a (i.e., $f_1 \rightarrow f_n + a$) or f_1 is not different from f_n with a (i.e., $f_1 = f_n + a$). Suppose $f_1 = f_n + a$ holds. By transitivity the catalytic chemical reaction $f_1 + a \rightarrow f_n + a$ holds, and this implies $f_1 + a \rightarrow f_1$ is not a physical change; a contradiction of the hypothesis. Since a cannot react with f_n to produce f_1 which it reacts with to produce f_n by the hypothesis, then $f_n + a \underset{rev}{\longrightarrow} f_1$ does not hold here (it is a reversible chemical reaction that may hold elsewhere, but its reverse reaction may hold here). Since every physical change is reversible, consider extending the reverse $f_1 \rightarrow f_1 + a$ of $f_1 + a \rightarrow f_1$ with the catalytic reaction $f_1 + a \rightarrow f_n + a$. Thus, we have that $f_1 \rightarrow f_n + a$ is a derived reaction. Therefore, the set $S = \{f_1,\ f_1 + a,\ f_n + a\}$ of substances is totally ordered with respect to the reaction/change ordering. By

Theorem 4.21, S is not an isosceles triple and the relation $f_1 \# f_n + a$ does not hold. Hence, only the relation $f_1 \rightarrow f_n + a$ holds and the result follows. □

5.2 Turing Computability

Classical computability can be presented in various equivalent mathematical formalisms such as *Turing machines*, *register machines*, *Chomsky type-zero grammars*, *partial recursive functions* and so on. In this thesis, we exploit the mechanism of a register machine. It is a device that consists of a given number of *registers* each of which can hold an arbitrarily large non-negative integers and a sequence of labeled instructions called *program*. The instructions determine how the integers in the registers can change and which instructions should be executed next after, l_h is the halting label and R is the set of instructions labeled by elements another. Formally, a typical *register machine* is a device $M = (m, B, l_0, l_h, R)$, where $m \geq 1$ is the number of registers, B is the set of instruction labels, l_0 is the intial label from B (R is also called the *program* of the machine). The labeled instructions are of the following forms:

$l_1 : (\mathrm{ADD}(r), l_2)$ (add 1 to register r and go to the instruction with label l_2),

$l_1 : (\mathrm{SUB}(r), l_2, l_3)$ (if register r is not empty, then subtract 1 from it and go to the instruction with label l_2, otherwise go to the instruction with label l_3),

$l_h : \mathrm{HALT}$ (the halt instruction which can only have the label l_h).

A register machine is used to recognize a number in the following manner: starting with all registers being empty, a number is introduced in a distinguished register (say, the first register), computation is made starting with the instruction labeled l_0; a number is inputted, say n in this register while all other registers hold the value 0. If the computation reaches the instruction labeled by l_h: HALT (that is, it halts), then the number is recognized, otherwise it is not recognized. The set of all numbers recognized by M is denoted by $N(M)$. It is known (see Minsky (1967) and Freund and Oswald (2002)) that register machines (with three registers only, recognize exactly the family *NRE*, of Turing computable numbers.

A register machine can also compute functions by designating certain registers as input registers and others as output registers. We start with the arguments of the function in the input registers. If the machine halts, we get the value of the function in the output registers; if the computation never halts, then the function is not defined for the given input. We denote by NRE the family of Turing computable enumerable sets of numbers.

5.3 A Nondeterministic Alternative P System with Specialization Rules Based on Ideal Ordering

We introduce a class of membrane system considered in this thesis, an alternative P system based on ideal ordering (or *ideal alternative P system*). We show universality for the (cell-like) P system, in the accepting mode using symbol objects, processed by some specialized rules. For now, we consider a system with only one membrane and

obtain universality by means of a register machine (Peter *et al.*, 2017d). In general, an alternative P system (of degree $m \geq 1$) based on ideal ordering is a construct of the form

$$\Pi = (O \cup H, \mu, w_1, \ldots, w_m, R_1, \ldots, R_n)$$

where

O is a finite alphabet, called objects,

H $= \{h_1, \ldots, h_n\}$ is a set of labels of the membranes in Π,

μ is a membrane structure of degree m,

w_i is a string over O representing the multiset of objects present in region i of the membrane system, where $1 \leq i \leq m$,

R_j is the set of rules which act in the region of membrane j such that $1 \leq j \leq n$,

The form of the rules are as follows:

i. Rules simulating reversible chemical reaction are of the form $u + v \underset{rev}{\longrightarrow} w$.

ii. Rules simulating derived reaction are of the form $u \rightarrow v + w$.

iii. Rules simulating catalytic reaction are of the form $u + v \rightarrow u + w$.

iv. Rules simulating mixture are of the form $u + \emptyset \Longrightarrow w$.

v. Rules simulating physical change are of the form $u + v \rightarrow u$.

vi. Rules simulating the null reaction are of the form $u + \emptyset \Longrightarrow u$.

Note that the null reaction does not mean the absence of a reaction, but it means the presence of a chemical environment in which the potential for a reaction to take place exists, without which no other reactions or any change can take place. Hence, the null reaction is the "most essential" of all reactions as it always exists. Therefore, it is considered one of the rules in the P system and is said to be among the rules that function at the background.

Since from Proposition 5.1, rules 3 and 5 are directly involved in the creation of rule 2, such rules are said to function at the background in the computation process of the P system. On the other hand, rules 2 and 4 are said to function only at the foreground. Furthermore, rule 1 functions at both background and foreground. Therefore, even though only rules 1, 2 and 4 are directly involved in the proof of computational completeness of the P system, the remaining rules are indirectly involved by virtue of their role in the creation of rule 2 (in the case of rules 3 and 5) and by virtue of the provision of a chemical environment (in the case of rule 6). In other words, all other rules imply rule 6 and rule 2 implies rules 3 and 5.

5.4 The Computational Power of Cooperative Alternative P System with Specialization Rules in One Membrane based on Ideal Ordering

The P system can generate all recursively enumerable sets of natural numbers. We denote by $NSPI_1(coo)$ the family of sets $N(\Pi)$ of numbers computed by a nondeterministic cooperative alternative P system with specialization rules based on ideal ordering

Theorem 5.2 $NSPI_1(coo) = NRE$

Proof

Consider a register machine $M = (m, B, l_o, l_h, P)$ and the set of alphabets $U = \{a_1, a_2, \dots, a_m\}$ where a symbol a_i is associated with register i. The content of the register is represented by the number of occurrence of a_i in the P system being constructed. An accepting P system of degree 1 with specialization rules based on ideal ordering is a construct of the form

$$\Pi = (O, [_1]_1, l_o, E, R_1)$$

where

$O = U \cup \{l_1, l_2, l_3\}$

$R_1 = \{(l_1 \to a_r + l_2) \mid \text{for } l_1 : (\text{ADD}(r), l_2) \in R\}$

$\cup \left\{\left(l_1 + a_r \underset{rev}{\longrightarrow} l_2\right), (l_1 + \emptyset \Longrightarrow l_3) \mid \text{for } l_1 : (\text{SUB}(r), l_2, l_3) \in R\right\}$

Assume that we introduce n copies of the object a in the system of Π just like starting the work of the register machine M where n is in the first register. We show that Π simulates the work of the register machine when analyzing the input n.

Each of the ADD instructions of M is simulated by the rule $l_1 \to a_r + l_2$. It introduces an object a_r together with the instruction label l_2 erasing the instruction label l_1.

The SUB instruction is simulated by the two rules $l_1 + a_r \underset{rev}{\longrightarrow} l_2$ and $l_1 + \emptyset \Longrightarrow l_3$. In the first case if there exists an object a_r in the system then the reaction erases both the object a_r and the label instruction l_1 while introducing the label instruction l_2. If, on the other hand, no element a_r exists in the system, then the rule $l_1 + \emptyset \Longrightarrow l_3$ is executed. It transforms the label instruction l_1 to the label instruction l_3. We continue to simulate the instructions of M in this way. In the event where the register machine halts, that is, it reaches the label l_h then the computation for Π halts as well, and conversely. Therefore, $N(\Pi) = N(M)$. □

5.5 Some Applications of P Systems to Arithmetic Operations on Non-negative Integers – a Case of Determinism Based on Weak Rule Priority

In this section, we construct a simple variant of P systems for arithmetic operations on non-negative integers based on the weak priorities for rule application. Consequently, we obtain deterministic P systems. It is verified that two membranes suffice. Moreover, there are at most four objects for multiplication and five objects for division throughout the computation processes. The P systems so obtained serve as bases upon which other complex operations may be built, and upon which basic operations may be extended to cover the set of positive and negative real numbers (Peter and Singh, 2017b).

5.5.1 Weak versus strong rule priority relations

Rules are sometimes assigned priorities to determine which rule is applied first and which is applied next. So far, there are two types of priority relations in membrane

computing, namely, strong and weak priority relations. In a strong priority relation, when a rule with higher priority has been applied, the rule with lower priority is not applied even if the two rules do not compete for objects. For instance, if $a \rightarrow c > b \rightarrow d$ are rules with the priority relation '$>$', meaning that $a \rightarrow c$ has priority over $b \rightarrow d$ and both a and b are available, then only the first rule is used, even though it has nothing to do with the object b. The relevance of this type of priority relation corresponds with the way of using priorities in *ordered grammars* in the regulated rewriting. It also has biochemical relevance. This relevance emanates from the idea that the rules could consume not only objects, but also a common resource such as energy. If a rule with higher priority is used, then no energy would be left for a rule with lower priority. On the other hand, in a weak priority relation, a rule with lower priority is applied after a rule with higher priority has been applied or cannot be applied, provided there are objects available for the rule with lower priority. In other words, a rule of lower priority is always used where a rule of higher priority has or has not been used provided the former can be applied. Both interpretation of rule priority are of interest. See Păun (2000) for further details.

We present P systems for the basic binary operations. There are two rules for addition and one rule for subtraction. No priority exists among the rules for addition – they are applied in parallel. On the other hand, there exist priorities among the rules in membrane 2 for both multiplication and division.

5.5.2 Addition P system

An addition P system is a construct of the form

$$\Pi^{+} = (V, \mu, w_1, (R_1, \rho_1), i_0)$$

where

$V = \{a, b, c\}$	is a set of objects.
$\mu = [_1]_1$	is a membrane structure of degree 1.
$w_1 = a^m b^n$	is the initial multiset of objects in the membrane.
$R_1 = \{a \to c\ , b \to c\}$	is a set of rules.
$i_0 = 1$	is the number indicating the output membrane.



Figure 11: The configurations of an addition P system

There are m copies of an object a and n copies of an object b in membrane 1 which is both an input and an output membrane. The m copies of a and n copies of b encode the numbers to be added. There are two rules necessary for the addition to take place, namely $a \to c$ and $b \to c$. The rule $a \to c$ consumes the m copies of a and produces m copies of c while the rule $b \to c$ consumes the n copies of b and produces n copies of c. The rules are applied in parallel in a single transition. Afterwards, no

copies of a or b exists within the membrane. Consequently, none of the rules can be applied. Therefore, the computation halts after the first transition. The result of the computation is encoded by the $m + n$ copies of c.

5.5.3 Subtraction P system

A subtraction P system is a construct of the form

$$\Pi^{-} = (V, \mu, w_1, (R_1, \rho_1), i_0)$$

where

$V = \{a, b, c\}$ is a set of objects.

$\mu = [_1]_1$ is a membrane structure of degree 1.

$w_1 = a^m b^n$ is the initial multiset of objects in the membrane.

$R_1 = \{ab \rightarrow (c, out)\}$ is a set of rules.

$i_0 = 1$ is the number symbolizing the output membrane.



Figure 12: The configurations of subtraction P system

The computation works as follows: m copies of an object a and n copies of an object b are placed in the only membrane labeled 1 which is both an input and output membrane. We assume $m \leq n$. The m and n encode the two numbers involved in the

binary operation of subtraction where m is to be subtracted from n. There is only one rule in the membrane, namely, $ab \rightarrow (c, out)$. The rule consumes m copies of the pair ab of a and b and produces m copies of a new object c which it sends out of the membrane. Afterwards, no copy of the pair ab exists in the membrane. Therefore, the rule cannot be applied again. The computation halts after the first transition. Consequently, there are $n - m$ copies of b in the output membrane. The result of the computation is determined by the $n - m$ copies of b.

5.5.4 Multiplication P system

Let m and n be non-negative integers. A multiplication P system for the product $m \times n$ is a construct of the form

$$\Pi^{\times} = (V, \mu, w_1, w_2, (R_1, \rho_1), (R_2, \rho_2), i_0)$$

where

$V = \{a, b, c, e\}$ is a set of objects.

$\mu = [_1[_2]_2]_1$ is a membrane structure of degree 2.

$w_1 = \emptyset$ is the empty multiset (no object present in membrane 1 at the initial configuration).

$w_2 = a^m b^n e$ is the initial multiset of objects in membrane 2.

$R_1 = \{r_1: a \rightarrow (a, in_2),\ r_2: e \rightarrow (e, in_2),\ r_6: a \rightarrow (a, out)\}$ is the set of rules in membrane 1.

$\rho_1 = \{r_1 > r_6\}$ is the priority relation of the rules in membrane 1.

$R_2 = \{r_3: be \rightarrow (e, out),\ r_4: e \rightarrow \delta, r_5: a \rightarrow (ac, out)\}$ is the set of rules in membrane 2.

$\rho_2 = \{r_3 > r_4 > r_5\}$ is the priority relation of the rules in membrane 2.

$i_0 = 1$ is the symbol for the output membrane.

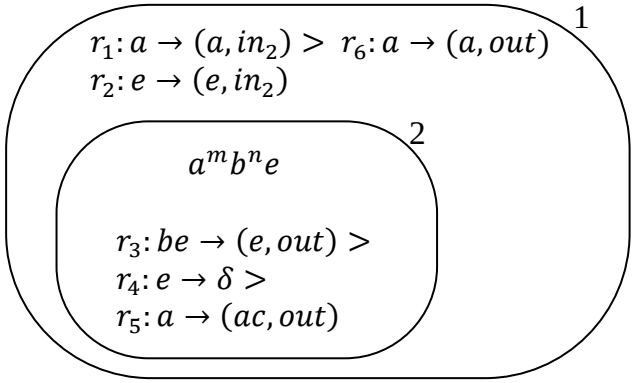


Figure 13: Initial configuration of a multiplication P system

At the initial configuration of the system, there are m copies of object a, n copies of object b in membrane 2 which encode the numbers to be multiplied and one copy of object e in membrane 2, the input membrane. Membrane 1 has three rules r_1, r_2 and r_6 where $r_1 > r_6$ is the weak rule priority relation which determines the precedence in selecting the rules. Membrane 2 also has three rules, r_3, r_4 and r_5 and they have been assigned the weak rule priority relations $r_3 > r_4 > r_5$.

Since there are no objects in membrane 1 at the initial configuration, none of the rules in membrane 1 can be applied. Since r_3 has priority over r_4 in membrane 2, r_3 will be applied. It consumes one copy of b and sends the only copy of e to

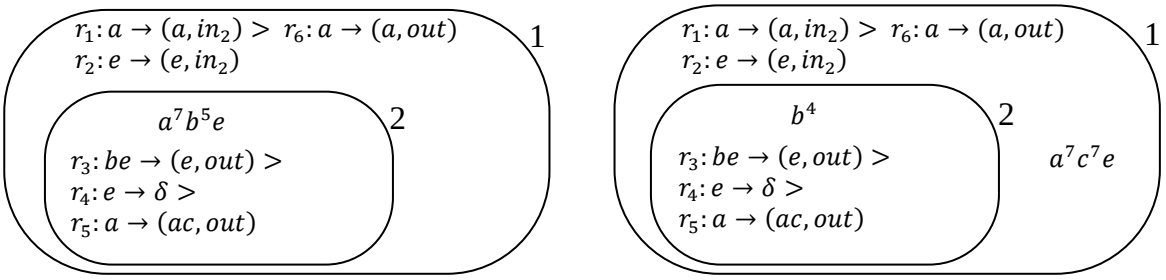
membrane 1, leaving m copies of a and $n-1$ copies of b behind. Next, even though r_4 has priority over r_5, r_4 cannot be applied since no object e is available in membrane 2 after r_3 has been applied. Therefore, r_5 will be applied in parallel with r_3 owing to the weak priority relation on the rules. It produces m copies of c according as there are m copies of a in membrane 2. The m copies of c and a are sent to membrane 1. Afterwards, there will be $n-1$ copies of b in membrane 2. This is the first transition.

Next, the copies of a and the only copy of e are sent back to membrane 2 by r_1 and r_2, respectively. Each time that the use of r_3 decreases the copies of b by one in membrane 2 and at the same time that r_4 cannot be applied, m copies of c are produced and sent to membrane 1 by r_5. The process continues until there are no copies of b left in membrane 2. Consequently, r_3 will no longer be applied. However, r_4 being the next in priority to r_3 will be applied owing to the availability of e in the membrane. It dissolves membrane 2. Eventually, r_1 and r_2 cannot be applied since they point to membrane 2. This will cause r_6 to be applied It sends the copies of a out of the membrane. At this point, no rule can be applied and the computation halts. The result of the computation is determined by the multiplicity of c available in membrane 1, which is the output membrane.

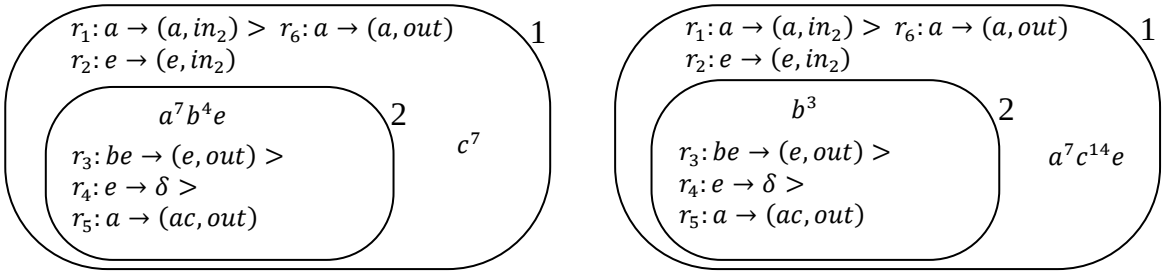
5.5.5 Multiplication P system exemplified

The following example demonstrates the computation of 5×7 by the multiplication P system. At the initial configuration there are seven copies of a, five copies of b and

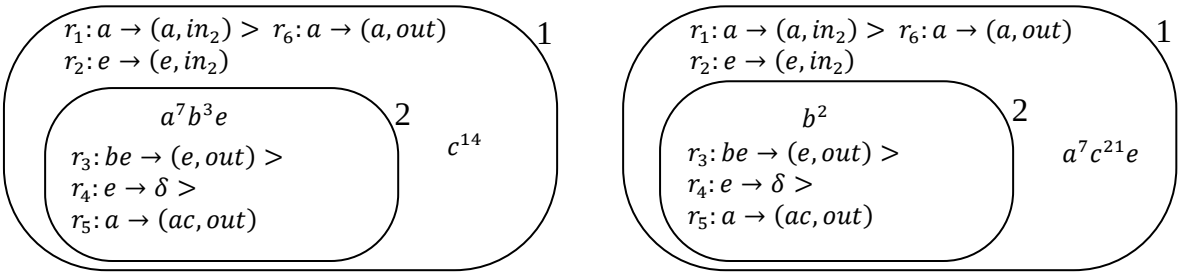
one copy of e in membrane 2, the input membrane. The input integers for the computation are encoded by the multiplicities of a and b. No rule can be applied in membrane 1 at this moment as it has no objects in it. Meanwhile, r_3 has priority over r_4 in membrane 2, therefore, r_3 will be applied. It consumes one copy of b and sends the only copy of e to membrane 1. The rule r_4 is next in priority, however, it will not be applied since no object e exists in membrane 2 having applied r_3. Rather, r_5 will be applied. It produces seven copies of c which it sends to membrane 1 together with the seven copies of a.



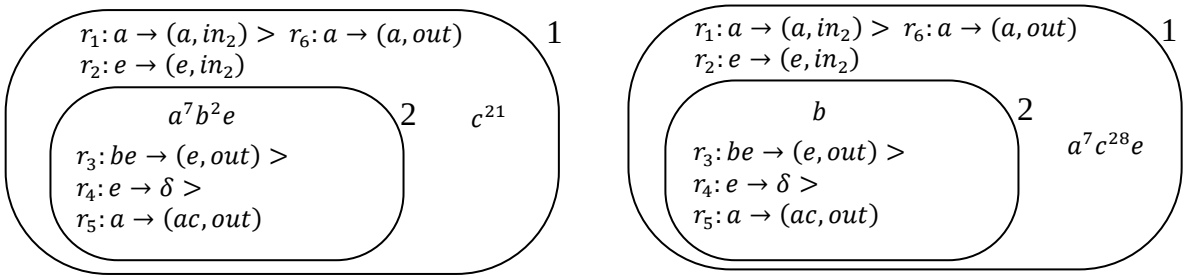
There are now seven copies of each of a and c and one copy of e in membrane 1. Since r_1 has priority over r_6 then r_1 and r_2 will be applied. The rule r_1 sends the seven copies of a to membrane 2 while r_2 sends the only copy of e to membrane 2. Thus, membrane 1 contains seven copies of c while membrane 2 contains seven copies of a, four copies of b and one copy of e.



In membrane 2, r_3 consumes one copy of b and sends one copy of e to membrane 1, at the same time r_5 produces seven copies of c and sends them to membrane 1 together with the seven copies of a. Thereafter, there would be seven copies of a, fourteen copies of c and one copy of e in membrane 1.

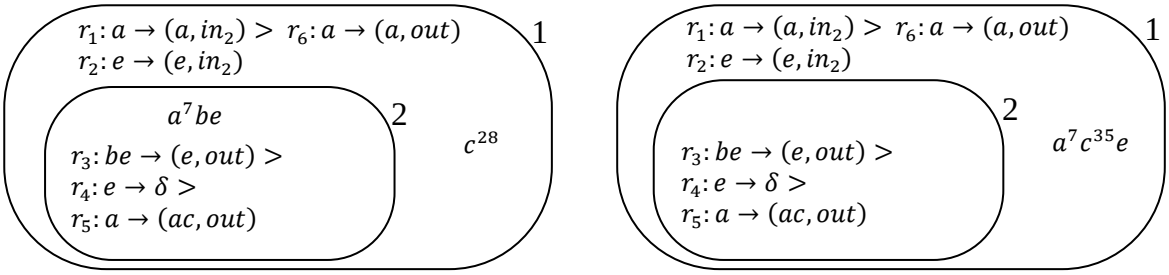


In membrane 1, r_1 sends seven copies of a to membrane 2 while at the same time r_2 sends one copy of e to membrane 2. In membrane 2, r_3 consumes one copy of b and sends the only copy of e to membrane 1, at the same time r_5 produces seven copies of c and sends them to membrane 1 together with the seven copies of a. There are now seven copies of a, twenty-one copies of c and one copy of e in membrane 1 while there are two copies of b in membrane 2.

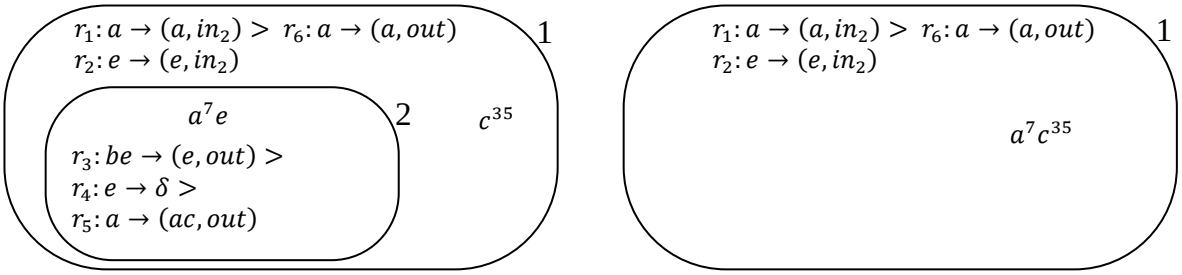


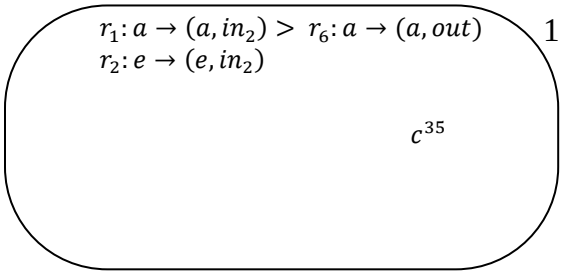
Again in membrane 1, r_1 sends seven copies of a to membrane 2 while r_2 sends the only copy of e to membrane 2. In membrane 2, r_3 consumes one copy of b and sends one copy of e to membrane 1, while again r_5 produces seven copies of c

and sends them to membrane 1 together with the seven copies of a. There are now seven copies of a, twenty-eight copies of c and one copy of e in membrane 1, while only one copy of b is left in membrane 2.



When the rules r_1 and r_2 are used again, the seven copies of a and the only copy of e are sent to membrane 2. Again in membrane 2, r_3 sends the only copy each of b and e to membrane 1, while again r_5 produces seven copies of c and sends them to membrane 1, together with the seven copies of a. There would then be seven copies of a, thirty-five copies of c and one copy of e in membrane 1. Finally, when r_1 and r_2 send, respectively, the seven copies of a and the only copy of e to membrane 2, there would be thirty-five copies of c in membrane 1 while membrane 2 would contain seven copies of a and one copy of e.





The availability of a copy of e without any copy of b in membrane 2 makes possible the use of r_4 for the first time. It dissolves membrane 2, consuming the only copy of e. Consequently, rules r_1 and r_2 cannot be applied as their target membrane has been dissolved. Rule r_6 will be applied. It sends the seven copies of a out of membrane 1, after which the computation halts. There are now only thirty-five copies of c which encode the result of the computation in membrane 1, the output membrane.

5.5.6 Division P system

Consider two non-negative integers m and n where $m \leq n$. The P system to compute $n \div m$ is a construct of the form

$$\Pi^{\div} = (V, \mu, w_1, w_2, (R_1, \rho_1), (R_2, \rho_2), i_0)$$

where

$V = \{a, b, c, d, e\}$ is a set of objects.

$\mu = [_1[_2]_2]_1$ is a membrane structure of degree 2.

$w_1 = \emptyset$ is the empty multiset (no object present in membrane 1 at the initial configuration).

$w_2 = a^m b^n d$ is the initial multiset of objects in membrane 2.

$R_1 = \{r_1: c \rightarrow (a, in_2), r_2: d \rightarrow (d, in_2), r_6: d \rightarrow (d, out)\}$ is the set of rules in membrane 1.

$\rho_1 = \{r_2 > r_6\}$ is the set of priority relation of the rules in membrane 1.

$R_2 = \{r_3: ab \rightarrow (c, out)\ , r_4: a \rightarrow \delta, r_5: d \rightarrow (de, out)\}$ is the set of rules in membrane 2.

$\rho_2 = \{r_3 > r_4 > r_5\}$ is the set of priority relations of the rules in membrane 2.

$i_0 = 1$ is the symbol for the output membrane.

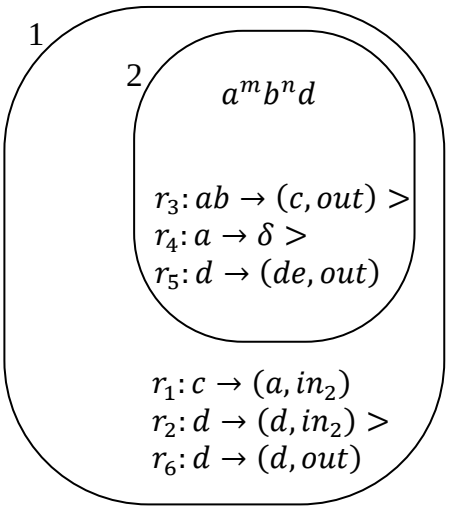


Figure 14: Initial configuration of a division P system

Membrane 2 has three rules r_3, r_4 and r_5 where $r_3 > r_4 > r_5$ are the weak rule priority relations among the rules. Membrane 1 has three rules, namely, r_1, r_2 and r_6

where $r_2 > r_6$. At the initial configuration, there are m copies of object a and n copies of object b in membrane 2 which is the input membrane, where m and n (assuming that $m \leq n$) encode the input integers for the division.

Since there are no objects in membrane 1 at the moment, none of the rules can be applied within. Since r_3 has priority over r_4 in membrane 2, r_3 will be applied. It consumes m copies of the pair of a and b and produces m copies of c and sends them to membrane 1. Since no object a exists in membrane 2 after r_3 has been applied, a rule with lesser priority is to be applied, which is r_4. But then it will not be applied since no object a is present in membrane 2. Therefore, r_5 will be applied in parallel with r_3. It produces one copy of a new object e and sends it to membrane 1 together with the only copy of d. Thus, $n - m$ copies of b are left in membrane 2, while m copies of c, one copy of d and one copy of e are present in membrane 1.

In membrane 1, r_6 will not be applied since it competes for the object d with r_2 which has a higher priority. Therefore, both r_1 and r_2 will be applied. The rule r_1 consumes the object c and produces the object a which it sends to membrane 2 while r_2 sends d back to membrane 2. As the iteration continues, the multiplicity of b is being reduced by m until it is less than m in membrane 2. Two cases to consider are where the multiplicity of b is zero and where it is greater than zero.

If the multiplicity of b is zero, then r_3 cannot be used again rather r_4 will be used for the first time. It dissolves membrane 2 and consumes the object a, at the

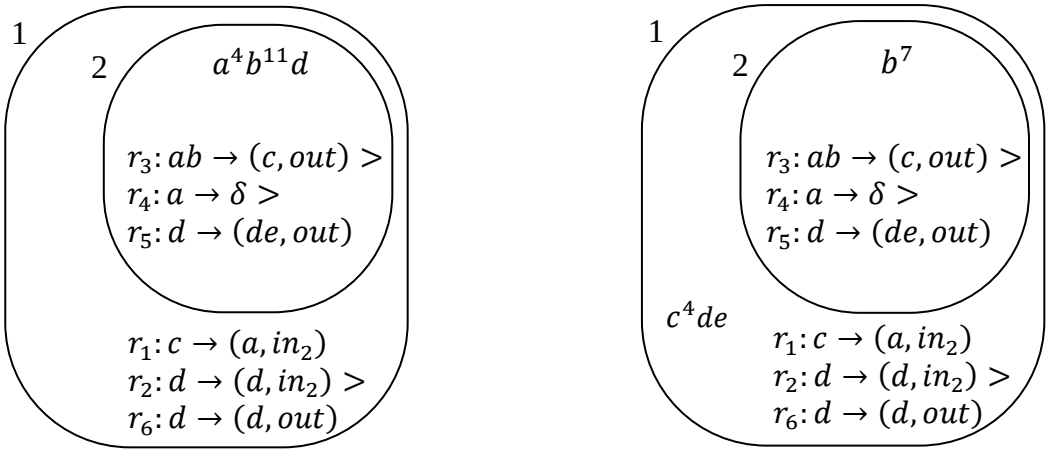
same time r_5 sends d to membrane 1 for the last time. Thus, r_1 and r_2 cannot be used again as their target membrane has been dissolved. Therefore, r_6 will be used for the first time. It sends d out of membrane 1. No rule can be applied at this point and the computation halts. Thus, there are only copies of e in membrane 1 and they encode the quotient in the division.

On the other hand, if the multiplicity of b is greater than zero then r_3 will be used for the last time. This time however, it is copies of a and not b that are left in membrane 2, together with the only copy of d. The rule r_4 consumes the remaining copies of a and dissolves membrane 2 while r_5 produces one copy of e and sends it to membrane 1 together with the only copy of d. Thus, copies of c and e and one copy of d are in membrane 1. The dissolution of membrane 2 renders r_1 and r_2 unusable, hence the use of r_6 for the first time. It sends the only copy of d out of the membrane. At this point, no rule can be applied and the computation halts. The copies of e encode the quotient in the division while the copies of c encode the remainder.

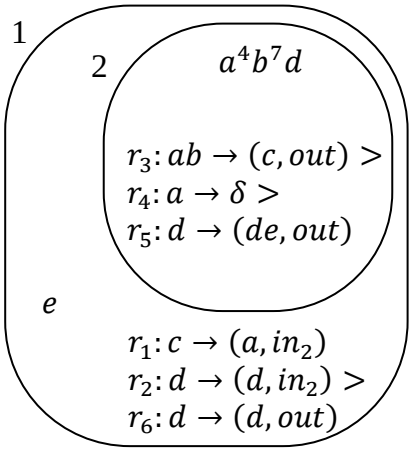
5.5.7 Division P system exemplified

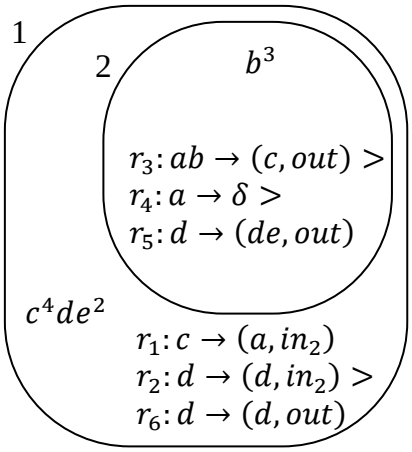
In the following example, we compute $11 \div 4$. At the initial configuration of the system, four copies of an objects a, eleven copies of an objects b and one copy of an object d are put in membrane 2. Any of the rules in membrane 2 can be applied, however, due to the priority among the rules, r_3 will be applied first. It consumes four copies of the pair of a and b and produces four copies of c which it sends to

membrane 1. Now no object a exists in membrane 2 and so r_5 will be applied in parallel with r_3. It produces one copy of a new object e and sends it to membrane 1 together with the only copy of object d. No other rules can be applied in membrane 2 at this point. There are now four copies of c, one copy of d and one copy of e in membrane 1.

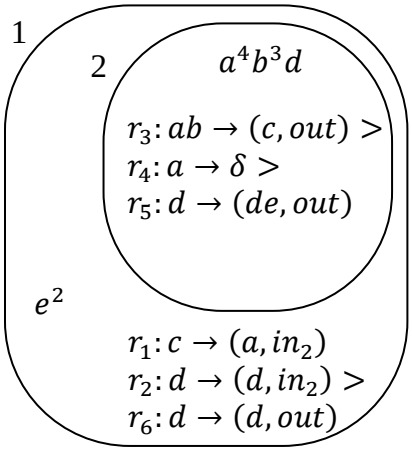


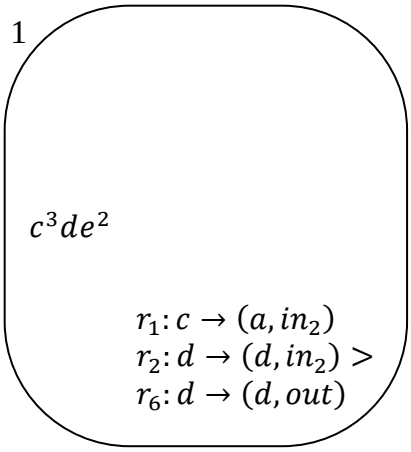
In membrane 1, both r_1 and r_2 can be applied. While r_1 consumes the four copies of c and produces four copies of a which it sends to membrane 2, r_2 sends the only copy of d to membrane 2. One copy of e is left in membrane 1. There are now four copies of a, seven copies of b and one copy of d in membrane 2. The rules r_3 and r_5 will be applied. While r_3 consumes four copies of the pair of a and b and sends four copies of c to membrane 1, r_5 sends the only copy of d to membrane 1 and produces a new copy of e which it sends to membrane 1. There are now four copies of c, one copy of d and two copies of e in membrane 1.





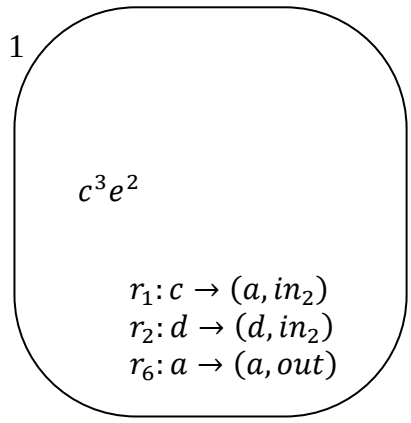
Again, r_1 consumes the four copies of c and produces four copies of a and sends them to membrane 2 while r_2 sends d to membrane 2. Consequently, all the rules in membrane two are lost and the object d falls back to membrane 1. Moreover, rules r_1 and r_2 cannot be applied in membrane 1 since their target membrane has been dissolved, hence, r_6 will be applied. It sends d out of the membrane.



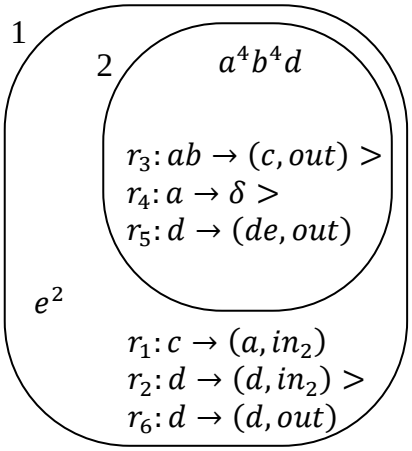


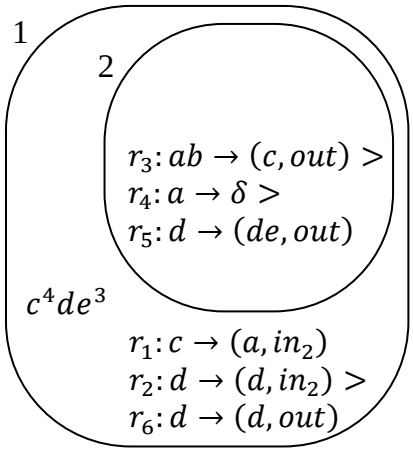
The computation halts. There are now three copies of c and two copies of e in membrane 1 which is the output membrane. The result of the computation is encoded

by the two copies of e which represent the dividend and the three copies of c which represent the remainder.

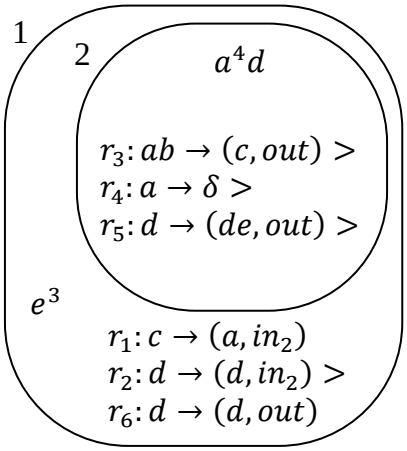


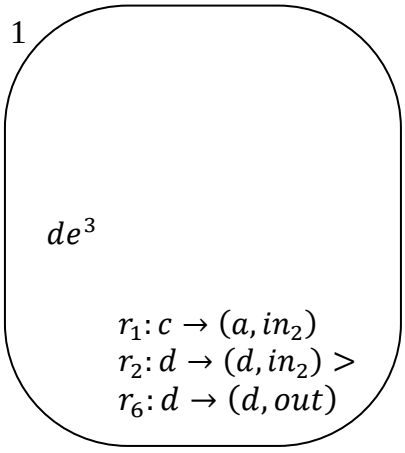
We now examine the nature of the result in this example if the task was to divide 12 by 4 starting from the configuration before the dissolution of membrane 2, where there are four copies of each of a and b and one copy of d in membrane 1 and two copies of e in membrane 2. The rule r_3 would consume the four copies of each of a and b and produce four copies of c which it would send to membrane 1. At the same time r_5 would produce a copy of e and send it to membrane 1 together with the only copy of d. There would be four copies of c, three copies of e and one copy of d in membrane 1, while membrane 2 would contain no objects.

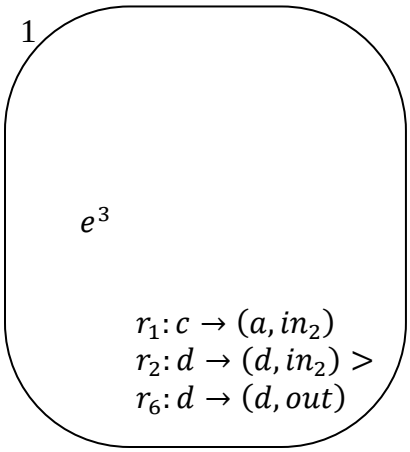




Next, r_1 would consume the four copies of c and produce four copies of a which it would send to membrane 2 while r_2 would send the only copy of d to membrane 2. In membrane 1, r_3 would be applied for the first time. It would consume the object a and at the same time dissolve membrane 2, allowing d to return to membrane 1.







Again, r_1 and r_2 could not be applied since they both point to the dissolved membrane 2. Therefore, r_6 would be applied for the first time. It sends the only copy of d out of the membrane. The computation would then halt. There would be three copies of e which encode the quotient. The non-availability of c is an indication that there exists no remainder from the division.

CHAPTER SIX

SUMMARY, CONCLUSION AND RECOMMENDATIONS

6.0 Summary

Some formalism of multiset orderings has been presented, in particular, a grid form of the Jouannaud-Lescanne set-based and submultiset-based multiset orderings are developed. The grid forms prove to be more flexible at implementation of the orderings as seen from the several results presented.

The possibility that there exists a set-based ordering stronger than the one introduced by Jouanaud-Lascanne was investigated. Such an ordering is obtained by an extension of the said Jouannaud-Lascanne ordering to accommodate simple multisets of incomparable objects, hence stronger than one thought.

A property-restricted consequence of partial ordering called *ideal consequence* of partial ordering which is named *ideal ordering* in this thesis, has been closely observe with examples.

The nondeterministic model of alternative P system presented, with only three rules at the foreground, has been shown to be computationally complete by means of a register machine. Moreover, such rules simulate chemical, biological, physical and other cellular activities in a specialized manner, hence, better simulating the activities of the cell than a conventional P system. It is also necessary to state that the model of

P system is an improvement to its symport and antiport counterpart presented in Freund and Păun (2003) having minimal rules and no more than one membrane.

The four basic arithmetic operations in deterministic P systems based on the weak rule priority has also been presented. Addition P system and subtraction P system use two rules and one rule respectively, both with one membrane. On the other hand, each of multiplication P system and division P system use six rules within two membranes.

6.1 Conclusion

A grid-based multiset ordering was introduced and was shown to be equivalent to the Jouannaud-Lascanne's partition-based multiset ordering (Theorem 4.5 and Theorem 4.12). However, the preference of the grid-based multiset ordering over its Jouannaud-Lascanne counterpart lies in its flexibility at implementation (Theorem 4.3, Theorem 4.4, Theorem 4.5, Theorem 4.6, Theorem 4.7, Theorem 4.11, Theorem 4.12, Theorem 4.13 and Theorem 4.14).

It seems possible that there exists a set-based ordering stronger than the one introduced by Jouanaud-Lascanne. Such an ordering is obtained by extending the Jouannaud-Lascanne ordering to accommodate simple multisets of incomparable objects (Theorem 4.26, Theorem 4.27 and Theorem 4.28).

The notion of *ideal ordering* was introduced and used to develop an alternative P system based on specialization rules (Proposition 5.1), which was shown to be computationally complete by means of a simple register machine (Theorem 5.2).

A deterministic P system for arithmetical operations based on weak rule priority relation having a minimal relational structure compared to an existing model (Guo and Zhang, 2008) was developed (Section 5.5).

6.2 Recommendations for Future Research

1. In this thesis, we did not determined the multisets A, B, C, D, E, F, G and H satisfying the conditions that $A \gg_{\mathbb{S}} B$ implies $A \gg_{\mathfrak{M}} B$, $C \gg_{\mathbb{S}} D$ implies $C \#_{\mathfrak{M}} D$, $E \gg_{\mathfrak{M}} F$ implies $E \gg_{\mathbb{S}} F$ and $G \gg_{\mathfrak{M}} H$ implies $G \#_{\mathbb{S}} H$ where $\gg_{\mathbb{S}}$ is the set-based ordering, $\gg_{\mathfrak{M}}$ is the submultiset-based ordering, $\#_{\mathbb{S}}$ is incomparability with respect to $\gg_{\mathbb{S}}$ and $\#_{\mathfrak{M}}$ is incomparability with respect to $\gg_{\mathfrak{M}}$. An idea is to identify two categories of set-based difference grids namely $\mathbb{S}_1$ and $\mathbb{S}_2$ such that $\gg_{\mathbb{S}_1}$ implies $\gg_{\mathfrak{M}}$ and $\gg_{\mathbb{S}_2}$ implies $\#_{\mathfrak{M}}$. Similarly, one needs to identify two categories of submultiset-based difference grids $\mathfrak{M}_1$ and $\mathfrak{M}_2$ such that $\gg_{\mathfrak{M}_1}$ implies $\gg_{\mathbb{S}}$ and $\gg_{\mathfrak{M}_2}$ implies $\#_{\mathbb{S}}$.

2. The deterministic P system on non-negative integers presented in Section 5.5 has not been able to accommodate negative integers. In order to encode negative integers, one could designate a separate membrane 3 within the skin membrane. A rule is placed within the membrane which would send objects to membrane 2 – such objects would be viewed as *borrowed objects* to membrane 2. It is easy to see that one can encode negative integers with less effort in this manner.

3. The deterministic P system presented has not also been able to accommodate real numbers. To encode real numbers, it is important to identify rational and irrational components. It is easy to encode positive rational numbers – the P system for every rational number is a division P system for two integers where the P system of the denominator does not represent zero. However, encoding irrational numbers will be more challenging.

INDEX OF RESULTS OBTAINED

Corollary 4.30----------100

Lemma 4.1----------60

Lemma 4.2----------64

Lemma 4.8----------73

Lemma 4.9----------76

Lemma 4.10 ----------77

Lemma 4.15----------84

Lemma 4.16----------85

Lemma 4.17----------85

Lemma 4.18----------86

Lemma 4.19----------86

Lemma 4.20----------87

Lemma 4.21----------87

Lemma 4.22----------88

Lemma 4.23----------88

Lemma 4.25----------90

Proposition 5.1 ----------113

Theorem 4.3----------67

Theorem 4.4----------69

Theorem 4.5----------70

Theorem 4.6----------71

Theorem 4.7----------72

Theorem 4.11 ----------80

Theorem 4.12 ----------81

Theorem 4.13 ----------83

Theorem 4.14 ----------84

Theorem 4.24 ----------89

Theorem 4.26----------92

Theorem 4.27----------93

Theorem 4.28 ----------95

Theorem 4.29 ----------99

Theorem 5.2----------118

REFERENCES

Alhazov, A. and Pan, L. (2004). Polarizationless P Systems with Active Membranes. *Grammars*, *7*(1), 141-159.

Alhazov, A., Pan, L. and Păun, Gh. (2004). Trading polarizations for labels in P systems with active membranes. *Acta Informatica*, *41*(2-3), 111-144.

Ardelean, I. I. and Cavaliere, M. (2003). Modelling biological processes by using a probabilistic P system software. *Natural Computing*, *2*(2), 173-197.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

Blizard, W. D. (1989). Multiset theory. *Notre Dame Journal of formal logic*, 30(1):36-66.

Casasnovas, J., Miro, J., Moya, M. and Rossello, F. (2004). An approach to membrane computing under inexactitude. *International Journal of Foundations of Computer Science*, *15*(06), 841-864.

Cavaliere, M., Ibarra, O. H., Păun, Gh., Egecioglu, O., Ionescu, M. and Woodworth, S. (2009). Asynchronous spiking neural P systems. *Theoretical Computer Science*, *410*(24), 2352-2364.

Ciobanu, G., Dumitriu, D., Huzum, D., Moruz, G. and Tanas A, B. (2003). Client–Server P Systems in modelling molecular interaction. In *Membrane Computing*(203-218). Springer Berlin Heidelberg.

Dassow, J. and Păun, Gh. (1989). *Regulated rewriting in formal language theory*. EATCS monographs in theoretical computer science, volume 18. Springer-Verlag.

Davis, L. (1991). *Handbook of genetic algorithms* (115). New York: Van Nostrand Reinhold.

Dershowitz, N. and Manna, Z. (1979), Proving Termination with Multiset Ordering, *Comm. ACM*, Vol. 22, 465-476.

Freund, R. (1999). Generalized P-systems with splicing and cutting/recombination. *Grammars*, *2*(3), 189-199.

Freund, R. (2001). Sequential P-systems. *Romanian Journal of Information Science and Technology*, *4*(1-2), 77-88.

Freund, R. (2004). Generalized Homogeneous P-Systems. *Grammars and Automata for String Processing: From Mathematics and Computer Science to Biology, and Back*, *9*, 381.

Freund, R. and Oswald, M. (2002). A short note on analysing P systems with antiport rules. *Bulletin of the EATCS*, *78*, 231-236.

Freund, R. and Oswald, M. (2005a). P colonies working in the maximally parallel and in the sequential mode. In *Symbolic and Numeric Algorithms for Scientific Computing, 2005. SYNASC 2005. Seventh International Symposium* (8-15). IEEE.

Freund, R. and Oswald, M. (2005b). *Tissue P systems with symport/antiport rules of one symbol are computationally complete*. Citeceer.

Freund, R. and Păun, G. (2001). On the number of non-terminal symbols in graph-controlled, programmed and matrix grammars. In *International Conference on Machines, Computations, and Universality* (214-225). Springer, Berlin, Heidelberg.

Freund, R. and Păun, Gh. (2003). On deterministic P systems. *See P Systems Web Page at http://psystems. disco. unimib. it*.

Freund, R., Martín-Vide, C., Obtułowicz, A. and Păun, Gh. (2003). On three classes of automata-like P systems. In *Developments in Language Theory* (292-303). Springer Berlin Heidelberg.

Frisco, P. (2000). On two variants of splicing super-cell systems. In*Workshop on Multiset Processing* (1-2).

Guo, P. and Zhang, H. (2008). Arithmetic operation in single membrane. In *Computer Science and Software Engineering, 2008 International Conference on* (Vol. 3, 532-535). IEEE.

Hauschildt, D. and Jantzen, M. (1994). Petri net algorithms in the theory of matrix grammars. *Acta Informatica*, 31(8):719-728.

Hopcroft, J. E. (1979). *Introduction to automata theory, languages, and computation.* Pearson Education India.

Huet, G. and Oppen, D. C. (1980). Equations and rewrite rules. *Formal language theory: perspectives and open problems*, 349-405.

Ibarra, O. H. (2005). On determinism versus non-determinism in P systems.*Theoretical Computer Science*, *344*(2), 120-133.

Ionescu, M., Păun, Gh. and Yokomori, T. (2006). Spiking neural P systems.*Fundamenta informaticae*, *71*(2), 279-308.

Jouannaud, J. and Lescanne P. (1982). On multiset ordering, *Information processing letters*, Volume 15, Number 2, 57-63.

Kelemen, J., Kelemenová, A. and Păun, Gh. (2004). On the power of a biochemically inspired simple computing model: P colonies. *Downloadable version at [9]*.

Kitano, H. (2002). Computational systems biology. *Nature*, *420*(6912), 206-210.

Knuth, D. E. (1973), Sorting and Searching (The Art of computer programming, vol. 3), Addison-Wesley, *Reading Mass*.

Krishna, S. N. and Păun, Gh. (2005). P systems with mobile membranes. *Natural Computing*, *4*(3), 255-274.

Krishna, S. N. and Rama, R. (2001a). A note on parallel rewriting in P systems. *Bulletin-European Association for Theoretical Computer Science*, *73*, 147-151.

Krishna, S. N. and Rama, R. (2001b). P systems with replicated rewriting. *Journal of Automata Languages and Combinatorics*, *6*(3), 345-350.

Lindenmayer, A. (1968). Mathematical models for cellular interactions in development i and ii. *Journal of theoretical biology*, 18(3):280-315.

Manca, V., Martin-Vide, C. and Păun, Gh. (2001). On the power of P systems with replicated rewriting. *Journal of Automata Languages and Combinatorics*, *6*(3), 359-374.

Martin-Vide, C. and Păun, Gh. (2000). Computing with membranes: One more collapsing hierarchy. *Bulletin-European Association for Theoretical Computer Science*, *72*, 183-187.

Martin-Vide, C. and Păun, Gh. (2001). *Elements of formal language theory for membrane computing*. Research Group on Mathematical Linguistics, Rovira i Virgili University, Report GRLMC, 21(01).

Minsky, M. L. (1967). *Computation: finite and infnite machines*. Prentice-Hall, Inc.

Mutyam, M., and Kriithivasan, K. (2001, May). P systems with active objects: Universality and efficiency. In *Proc. Conf. Universal Machines and Computations, Chi sin au, Springer, Berlin*.

Nishida, T. Y. (2002). Simulations of photosynthesis by a K-Subset transforming system with membrane. *Fundamenta Informaticae*, *49*(1), 249-259.

Nishida, T. Y. (2004). An approximate algorithm for NP-complete optimization problems exploiting P systems. In *Proc. Brainstorming Workshop on Uncertainty in Membrane Computing* (185-192).

Păun, A. (2001). On P systems with active membranes. In *Unconventional Models of Computation, UMC'2K* (187-201). Springer London.

Păun, A. and Păun, M. (2001). On membrane computing based on splicing. In *Where mathematics, computer science, linguistics and biology meet* (409-422). Springer Netherlands.

Păun, Gh. (1998) Computing with membranes. *TUCS Report 208. Turku Centre for Computer Science* ISBN 978-952-12-0303-9.

Păun, Gh. (1999). *P systems with active membranes: Attacking NP complete problems, submitted 1999, and Auckland University* (No. 102). CDMTCS Report.

Păun, Gh. (2000). Computing with membranes. *Journal of Computer and System Sciences*, *61*(1), 108-143.

Păun, Gh. (2002). *Membrane computing: an introduction*. Springer Science and Business Media.

Păun, Gh. (2004). Membrane computing: Some non-standard ideas. In *Aspects of Molecular Computing* (322-337). Springer Berlin Heidelberg.

Păun, Gh. (2006). Introduction to membrane computing. In *Applications of Membrane Computing* (1-42). Springer Berlin Heidelberg.

Păun, Gh. and Păun, R. A. (2005). Membrane computing as a framework for modelling economic processes. In *Symbolic and Numeric Algorithms for Scientific Computing, 2005. SYNASC 2005. Seventh International Symposium on* (8-28), IEEE.

Păun, Gh., and Rozenberg, G. (2002). A guide to membrane computing. *Theoretical Computer Science*, *287*(1), 73-100.

Păun, Gh., Pérez-Jiménez, M. J. and Rozenberg, G. (2006). Spike trains in spiking neural P systems. *International Journal of Foundations of Computer Science*, *17*(04), 975-1002.

Păun, Gh., Pérez-Jiménez, M. J., Pazos, J. and Rodríguez-Patón, A. (2005). Symport/antiport P systems with three objects are universal. *Fundamenta Informaticae*, *64*(1), 353-367.

Păun, Gh., Rozenberg, G. and Salomaa, A. (1998). *DNA computing: new computing paradigms*. Springer.

Păun, Gh., Sakakibara, Y. and Yokomori, T. (2002). P systems on graphs of restricted forms. *Publicationes Mathematicae Debrecen*, *60*, 635-660.

Păun, Gh., Suzuki, Y. and Tanaka, H. (2001). P systems with energy accounting. *International Journal of Computer Mathematics*, *78*(3), 343-364.

Peter, C. M. and Singh, D. (2013), Grid ramification of set-based multiset ordering, *Asian Journal of Fuzzy and Applied Mathematics* (ISSN: 2321 – 564X), Volume 01- Issue 03, October, 51-60.

Peter, C. M. and Singh, D. (2015), Redefined submultiset-based multiset ordering via grid, *Asian Journal of Fuzzy and Applied Mathematics* (*ISSN: 2321 – 564X*), Volume 03- Issue 06, December, 158-167.

Peter, C. M. and Singh, D. (2017a), A comprehensive alternative proof of the equivalence of Dershowitz-Manna and Huet-Oppen orderings, *FUDMA Journal of Sciences, vol. 1, No. 1*, 19-22.

Peter, C. M. and Singh, D. (2017b) Arithmetic operations in deterministic P systems based on the weak rule priority, *13th International Conference of the Nigerian Computer Society, Vol. 28, no. 10, Abuja*, 92-100.

Peter, C. M. and Singh, D. (2017c), A possibility of the existence of set-based ordering up to simple mutisets of incomparable objects (submitted).

Peter, C. M., Singh, D. and Atawodi, S. (2017d), A consideration of property–restricted ordering for specialization P systems (submitted).

Rozenberg, G. and Salomaa, A. (1980). *Mathematical Theory of L Systems*. Academic Press, Inc.

Rozenberg, G. and Salomaa, A. (1997). *Handbook of Formal Languages: Beyonds words*, volume 3. Springer Science and Business Media.

Singh, D. and Peter, C. M. (2011). Multiset-based tree model for membrane computing. *Computer Science Journal of Moldova*, *19*(1), 3-28.

Singh, D., Ibrahim, A., Yohanna, T. and Singh, J. (2007). An overview of the applications of multisets. *Novi Sad Journal of Mathematics*, 37(3):73- 92.

Suzuki, Y. and Tanaka, H. (2000). A new molecular computing model, artificial cell systems. In *Genetic and Evolutionary Computation Conference* (833-840).

Suzuki, Y., Ogishima, S. and Tanaka, H., (2003), Modelling the p. 53 signaling network by using P systems. In *Proceedings of the Brainstorming Week on Membrane computing; Tarragona, February 2003.* Technical Report 26/03, Rovira I Virgili University, Tarragona.

Syropoulos, A. (2000). Mathematics of multisets. In *Workshop on Membrane Computing* (347-358). Springer, Berlin, Heidelberg.

Tella, Y., Singh, D. and Singh, J. N. (2014). A Comparative Study of Multiset Orderings. *International Journal of Mathematics and Statistics Invention (IJMSI), 2*(5), 59-71.

Tomita, M., (2001) Whole-cell simulation: A grand challenge of the 21st century. Trends in Biotechnology, 19, 205-210.

Zandron, C., Ferretti, C. and Mauri, G. (2001). Solving NP-complete problems using P systems with active membranes. In *Unconventional Models of Computation, UMC'2K* (289-301). Springer London.