ConAction, Second Edition.

Galen Seilis

2022-09-16

Table of contents

Prefaces

Preface to the First Edition

Throughout the writing of this thesis I have received support and assistance that should be acknowledged.

My thanks to Dr. Alex Aravind who initially took me on as a student in 2020. He was instrumental in guiding me in the computer science aspects of this project. While he was my supervisor he always told me to keep going and enjoy the process. I have tried my best to stay true to his advice throughout this thesis.

I must equally thank Dr. Edward Dobrowolski for taking me on as a student in 2021 when Dr. Aravind unexpectedly became unable to continue as my supervisor. His encouragement and guidance have been invaluable in embracing my strengths in this thesis project. I have benefited from his facility with natural languages.

I also thank my committee members Dr. Brent Murray and Dr. Mohammad El Smaily for both encouraging and challenging me. Their ernest enquiries to understand my work have pushed me to think more clearly.

I wish to show gratitude to particular members of the UNBC faculty. I thank Dr. Stephen Rader, Dr. Margot Mandy, and Dr. Alia Hamieh for their advice and support. I thank Dr. Andy Wan for allowing me to sit in on his Mathematics of Machine Learning course, and I thank both Dr. Andy Wan and Dr. Geoffrey McGregor for always supporting my participation in the UNBC Interdisciplinary Weekly Seminar Series as a speaker and as an audience participant.

Thanks is also due to my friends and colleagues in the former Aravind research group including Dylan Fossl, Daniel Kopf, Conan Veitch, and Daniel O'Reilly for their sincerity and support in a confusing loss of Dr. Aravind as a supervisor.

And I thank my others friends who inspired or encouraged me during this thesis including Simon Harris, Amy Jimmo, Catherine Sprangers, Alix Schebel, and Richard Nhan.

Most of all I thank my family including Aaron Seilis, Pat Seilis, Mark Seilis, Flo Dew, and Derwyn Dew for their unwavering support. My grandfather, Derwyn Dew, passed away in January of 2022 due to complications of esophageal cancer. He is loved and missed.

Preface to the Second Edition

This second edition was motived by learning about the existence of the Quarto document preparation system. As an exercise I have migrated my thesis to an online book format. Even after completing my thesis and graduating with a masters in Computer Science, I keep busy learning.

I still feel gratitude to all the people who helped me get this far.

Developing this second edition has also given me the opportunity to address errata in the first edition and add clarifications or additional information.

1 Abstract

Complexity poses a pervasive challenge in understanding formal and natural systems which can arise from a combination of a system's state and state transition rules. In situations in which many aspects of a system are changing together, it is desirable to quantify how much they do so. We motivate and define five mathematical functions that can be used to quantify coordinated changes in structure. We also developed ConAction, a Python package which implements these novel mathematical tools in a way that is performant, easy to install, and easy to use. These new tools can be applied to real research problems, which we exemplified by evaluating a classic isolation by distance model for *Dendroctonus ponderosae* populations in western North America.

2 Introduction

2.1 Complexity

Defining "complexity" can seem ironic because it turns out to be a 'complex' endeavour. But in this section we will attempt to explain some of its character, and why it is a challenge in the sciences. The complexity of a system generally comes from a combination of two sources: the system's state and the system's state transition rules.\

Formal systems are structures like sets, sequences, vectors, scalars, matrices, tensors, graphs, hypergraphs, or any of a myriad of algebraic structures or spaces. In the simple case of a set as a system, one can think of the elements of the set as its state along with some rules for changing the elements of the set. Two contrasting examples of formal systems are cellular automata and systems of diffrential equations.\

Figure **??** (a) shows an elementary (1-dimensional) cellular automata known as Rule 110. In such a system the entire state of the system is represented as a one-hot vector (i.e. a vector of zeros and ones). In such a system time can be thought of discretely as a count of the number of applications of the rules, which are applied simultaneously on the current state of the system to get the next state. The state transition rules dictate how a given bit is replaced with another depending on its value as well as the values of its left and right neighbours. It is possible *prima facie* to believe that such systems are only capable of the simplest of patterns based on the belief that simple rules imply simple behaviour. But this is not so. In 2004 Mathew Cook published a proof that Rule 110 is Turing complete (Cook (2004)), i.e. equivalent to a Turing machine, a classification indicating that Rule 110 can run any program depending on the size and configuration of its input.[1] As a practical point of comparison, a phone or a PC can be entirely represented by a Turing machine. This includes everything from playing your favourite music or video, playing the latest computer games, calling a friend, running advanced scientific simulations, and even representing the entirety of whatever the Internet is doing right at this moment. Thus complexity really **can** come from simple rules with a suitable system configuration.

Stephen Wolfram has argued that some systems, including those with simple rules and relatively small state spaces, can still be what he calls "*computationally irreducible*"(Wolfram

[1] A system would have to be infinitely-large to be Turing complete. If Rule 110 elementary automata were given an infinitely-large input this would be apt, however it still speaks to the ability of a system to represent complex patterns if it is a finite approximation to a Turing machine.