Theory of Computation

Turing Machines

Dr Shekhar “Shakes” Chandra
shekhar.chandra@uq.edu.au

V1.0

> “Computing Science is no more about computers than Astronomy is about telescopes.”
>
> - Edsger W. Dijkstra
> (1930-2002)

Getting Help

Post questions on Piazza
Signup here

All lecturers and most tutors are monitoring this forum!

If it is a private matter or related to a personal issue

Dr Shekhar “Shakes” Chandra
shekhar.chandra@uq.edu.au

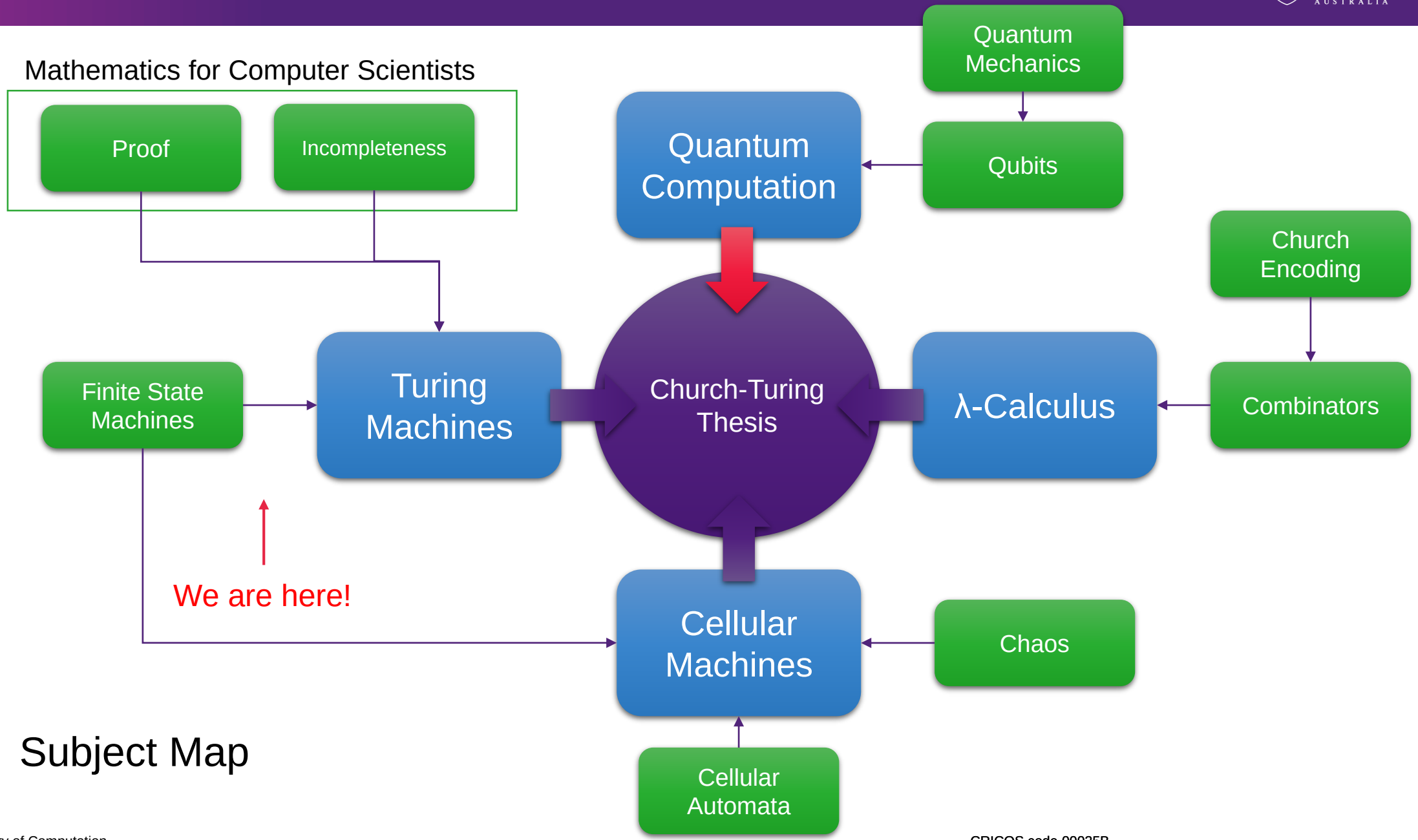
Code for Piazza: Godel

Overview

- Previous Module Summary
- Subject Map
- Relevant Chapters
- What is required of You!
- Upon completion …

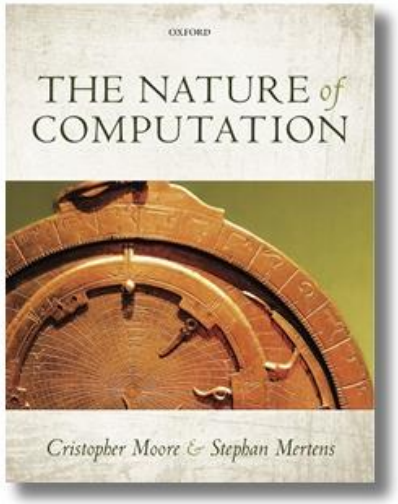
What did we do last time?

- We saw how mathematics consists of theorems
- Theorems are very strong scientific facts
- Despite our best efforts, even with theorems, there will be some results that cannot be proven to be true or false.
- This is Gödel's incompleteness theorem.


Mathematics for Computer Scientists
Proof
Incompleteness
Quantum Mechanics
Qubits
Quantum Computation
Church Encoding
Combinators
Finite State Machines
Turing Machines
Church-Turing Thesis
λ-Calculus
We are here!
Cellular Machines
Chaos
Cellular Automata

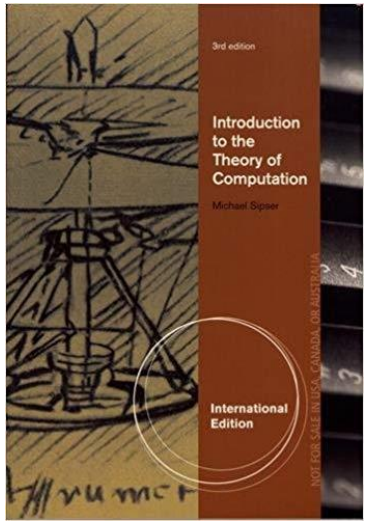
Subject Map

Prescribed & Recommended Texts



Cristopher Moore and Stephan Mertens **(2011).**
The Nature of Computation
Oxford University Press.
[eBook – UQ Library] [Moore's Notes on FSMs]
Chapter 7

Michael Sipser **(2012, originally 1996).**
Introduction to the Theory of Computation
Cengage Learning
[Hard Copy - UQ Library]
Chapter 1



Shakes Chandra **(????).**
Modern Computation
Chapter 3

What is required of You!

- Read Shakes' Notes on Turing Machines (~60 mins)
- Read Moore's Notes on Finite State Machines (sections 1-3, ~60 mins)
- Read Moore and Mertens Chapter 7, section 7.5 (~ 2-3 Hours)
- Review the demonstrations and examples conducted during the lectures. (~15 mins each lecture)
- Attempt the tutorials and complete the codebreaking and busy beaver demos

Upon completion, you should be able to:

- Describe a Turing machine and how it encapsulates the simplest computer
- Describe and understand the halting problem
- Formally describe how a finite state machine (FSM) is defined
- Convert between state diagrams and transition rules for FSMs
- Perform FSM computations for a given FSM and input
- Describe what is a regular language
- Design a FSM
- Formally describe how a Turing machine is defined
- Design and program a Turing machine
- Perform Turing machine computations for a given Turing machine and its inputs

What's Next?

Next we will see that computation "machines" can de constructed based on a just a few very simple rules or even chaotic systems that do not seem to have an order!

THE UNIVERSITY OF QUEENSLAND
AUSTRALIA
CREATE CHANGE

Thank you

Dr Shekhar “Shakes” Chandra | Lecturer
School of Information Technology and Electrical Engineering
shekhar.chandra@uq.edu.au
+61 7 3365 8359

@shakes76
shekhar-s-chandra