

CS6104 : Quantitative Verification 2025

Lecture 0: Introduction and logistics

Instructor: S. Akshay

IIT Bombay, India

06-01-2025

CS6104: Quantitative Verification (QV)

Course information

- ▶ Course Slot 13: Mon/Thu 19:00-20:25
- ▶ Course Venue: CC 105
- ▶ Course Webpage: <https://tinyurl.com/CS6104-2025>
- ▶ My office (temp): CC 313 (drop by if you have questions)

CS6104: Quantitative Verification (QV)

Course information

- ▶ Course Slot 13: Mon/Thu 19:00-20:25
- ▶ Course Venue: CC 105
- ▶ Course Webpage: <https://tinyurl.com/CS6104-2025>
- ▶ My office (temp): CC 313 (drop by if you have questions)

What is this course about?

CS6104: Quantitative Verification (QV)

Course information

- ▶ Course Slot 13: Mon/Thu 19:00-20:25
- ▶ Course Venue: CC 105
- ▶ Course Webpage: <https://tinyurl.com/CS6104-2025>
- ▶ My office (temp): CC 313 (drop by if you have questions)

What is this course about?

Who cares about this?

CS6104: Quantitative Verification (QV)

Course information

- ▶ Course Slot 13: Mon/Thu 19:00-20:25
- ▶ Course Venue: CC 105
- ▶ Course Webpage: <https://tinyurl.com/CS6104-2025>
- ▶ My office (temp): CC 313 (drop by if you have questions)

What is this course about?

Who cares about this?

Are you ready for this course?

What is Verification?

What is Verification?



System

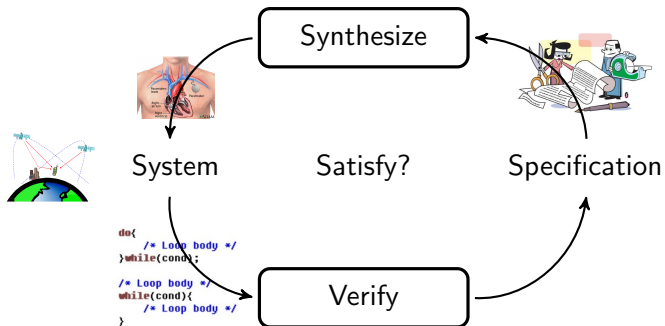
Satisfy?



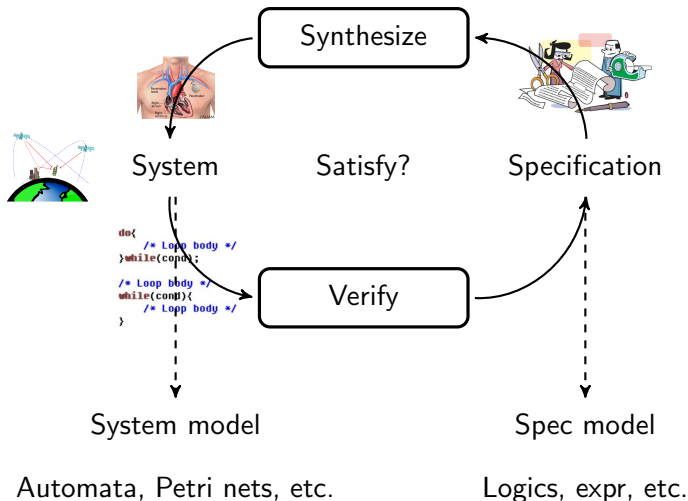
Specification

```
do{  
    /* Loop body */  
}while(cond);  
  
/* Loop body */  
while(cond){  
    /* Loop body */  
}
```


What is Verification?




What is Verification?




Verification and Model checking: Two Turing Awards!

1996	Amir Pnueli		"For seminal work introducing temporal logic into computing science and for outstanding contributions to program and systems verification " ^[73]
------	-------------	---	---

Edmund M. Clarke	
------------------	---


2007 Turing Award Winners Announced

, February 04, 2008

E. Allen Emerson	
------------------	---





For their groundbreaking work on Model Checking

Edmund M. Clarke, E. Allen Emerson, and Joseph Sifakis are the recipients of the 2007 A.M. Turing Award for their work on an automated method for finding design errors in computer hardware and software.

Joseph Sifakis	
----------------	---





The method, called [Model Checking](#), is the most widely used technique for detecting and diagnosing errors in complex hardware and software design. It has helped to improve the reliability of complex computer chips, systems and networks.

Verification and Model checking: Two Turing Awards!

1996	Amir Pnueli		"For seminal work introducing temporal logic into computing science and for outstanding contributions to program and systems verification" ^[73]
Edmund M. Clarke		2007 Turing Award Winners Announced	
E. Allen Emerson		, February 04, 2008	
Joseph Sifakis		For their groundbreaking work on Model Checking Edmund M. Clarke, E. Allen Emerson, and Joseph Sifakis are the recipients of the 2007 A.M. Turing Award for their work on an automated method for finding design errors in computer hardware and software. The method, called Model Checking , is the most widely used technique for detecting and diagnosing errors in complex hardware and software design. It has helped to improve the reliability of complex computer chips, systems and networks.	

- ▶ Used in hardware/software industries: Intel, Google, TCS, etc.
- ▶ For a more colorful video, see https://www.youtube.com/watch?v=AM_gwEKjnGY

Verification and Model checking: Two Turing Awards!

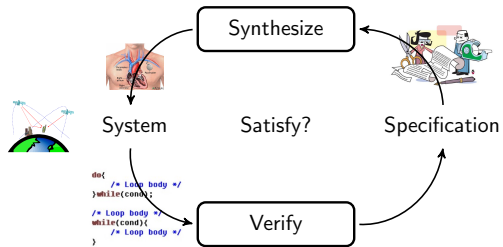
1996	Amir Pnueli		"For seminal work introducing temporal logic into computing science and for outstanding contributions to program and systems verification" ^[73]
2007 Turing Award Winners Announced			
, February 04, 2008			
For their groundbreaking work on Model Checking			
Edmund M. Clarke		Edmund M. Clarke, E. Allen Emerson, and Joseph Sifakis are the recipients of the 2007 A.M. Turing Award for their work on an automated method for finding design errors in computer hardware and software.	
E. Allen Emerson			
Joseph Sifakis		The method, called Model Checking , is the most widely used technique for detecting and diagnosing errors in complex hardware and software design. It has helped to improve the reliability of complex computer chips, systems and networks.	

- ▶ Used in hardware/software industries: Intel, Google, TCS, etc.
- ▶ For a more colorful video, see https://www.youtube.com/watch?v=AM_gwEKjnGY

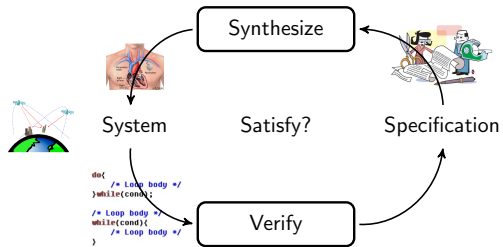
So what are the challenges left?

- ▶ Can we use it for Intelligent Systems?
- ▶ Rigorous guarantees of safety but high cost of analysis?

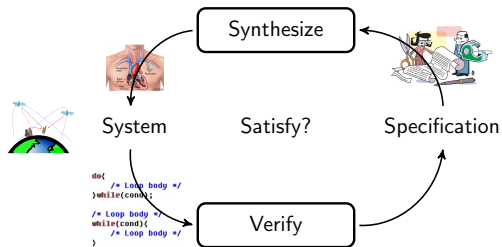
What is missing?



What is missing? Quantities!



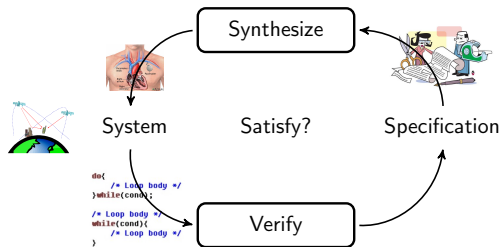
What is missing? Quantities!



Two Grand Challenges for the New Age!

- Expressivity
- Scalability

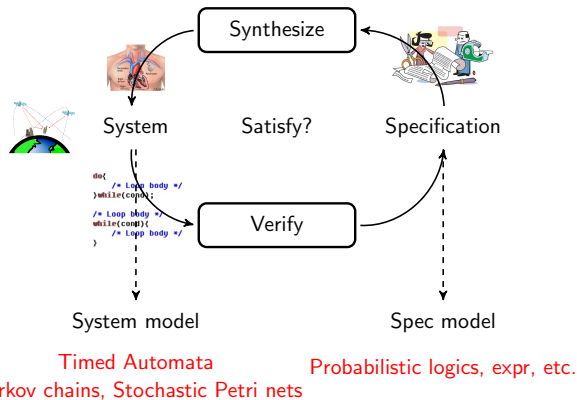
What is missing? Quantities!



Two Grand Challenges for the New Age!

- Expressivity : Richer models like MDPs, Cyber-physical systems, Neural Networks
- Scalability

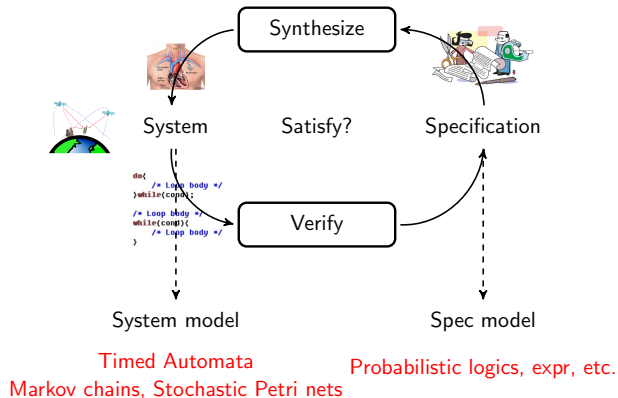
What is missing? Quantities!



Two Grand Challenges for the New Age!

- Expressivity : Richer models like MDPs, Cyber-physical systems, Neural Networks
- Scalability

What is missing? Quantities!



Two Grand Challenges for the New Age!

- Expressivity : Richer models like MDPs, Cyber-physical systems, Neural Networks
- Scalability : Exploit power of SAT/SMT-solvers, Data-driven approaches

Who cares about it?

Quantitative Verification already widely used in industry

- ▶ Hardware and software verification
- ▶ Cyber-physical systems: Avionics, Automobiles, space!
- ▶ AI models: Neural networks, Tree ensembles.

Who cares about it?

Quantitative Verification already widely used in industry

- ▶ Hardware and software verification
- ▶ Cyber-physical systems: Avionics, Automobiles, space!
- ▶ AI models: Neural networks, Tree ensembles.

Including Amazon, Intel, IBM, Microsoft Research, Google, Samsung, TCS, NASA, Mathworks ...

Who cares about it?

Quantitative Verification already widely used in industry

- ▶ Hardware and software verification
- ▶ Cyber-physical systems: Avionics, Automobiles, space!
- ▶ AI models: Neural networks, Tree ensembles.

Including Amazon, Intel, IBM, Microsoft Research, Google, Samsung, TCS, NASA, Mathworks ...

Widely used in Academia too

- ▶ Research groups in all major universities around the world
- ▶ An interplay of mathematics and computer science, logic and coding, theory and practice.

Our goal

A two-fold approach

1. Develop automated techniques that are, at the same time,
 - ▶ **expressive** : richer models
 - ▶ **scalable** : built on open-source implementations
 - ▶ **reliable** : formal guarantees of correctness and performance.

Our goal

A two-fold approach

1. Develop automated techniques that are, at the same time,
 - ▶ **expressive** : richer models
 - ▶ **scalable** : built on open-source implementations
 - ▶ **reliable** : formal guarantees of correctness and performance.
2. Understand theoretical hardness and limitations
 - ▶ Figure out how to subvert them
 - ▶ Lead to foundational improvements

Our goal

A two-fold approach

1. Develop automated techniques that are, at the same time,
 - ▶ **expressive** : richer models
 - ▶ **scalable** : built on open-source implementations
 - ▶ **reliable** : formal guarantees of correctness and performance.
2. Understand theoretical hardness and limitations
 - ▶ Figure out how to subvert them
 - ▶ Lead to foundational improvements

“Over the past 30 years, slowly but surely, a virtuous cycle has formed: automated reasoning in specific and critical application areas drives more investment in foundational tools, while improvements in the foundational tools drive further applications. Around and around.”

Our goal

A two-fold approach

1. Develop automated techniques that are, at the same time,
 - ▶ **expressive** : richer models
 - ▶ **scalable** : built on open-source implementations
 - ▶ **reliable** : formal guarantees of correctness and performance.
2. Understand theoretical hardness and limitations
 - ▶ Figure out how to subvert them
 - ▶ Lead to foundational improvements

“Over the past 30 years, slowly but surely, a virtuous cycle has formed: automated reasoning in specific and critical application areas drives more investment in foundational tools, while improvements in the foundational tools drive further applications. Around and around.”

- Byron Cook, Automated Reasoning's Scientific Frontiers, '22.

What this course is not about and what it is about

The idea of this course is not to...

- ▶ ... cover tools for automated reasoning in general (CS 433)
- ▶ ... cover techniques for model checking in general (CS 738)
- ▶ ... even exhaustively cover all logics, automata, models

What this course is not about and what it is about

The idea of this course is not to...

- ▶ ... cover tools for automated reasoning in general (CS 433)
- ▶ ... cover techniques for model checking in general (CS 738)
- ▶ ... even exhaustively cover all logics, automata, models

The idea of this course is to...

- ▶ ... focus on foundations of **Quantitative** Verification
- ▶ ... look at some select quantitative models and techniques in detail
- ▶ ... work out theory, write proofs, read/present relevant papers (also code if there is time!)
- ▶ ... prepare students to do internships/R&D/BTP/MTP/PhD in this area!

Course structure

Topics to be covered in this course:

- ▶ Module 1: Probabilistic Models: Markov chains and Markov decision processes (MDPs)
- ▶ Module 2: Quantitative Automata Models

Course structure

Topics to be covered in this course:

- ▶ Module 1: Probabilistic Models: Markov chains and Markov decision processes (MDPs)
- ▶ Module 2: Quantitative Automata Models
- ▶ Module 3: Quantitative Properties and Reasoning

Course structure

Topics to be covered in this course:

- ▶ Module 1: Probabilistic Models: Markov chains and Markov decision processes (MDPs)
- ▶ Module 2: Quantitative Automata Models
- ▶ Module 3: Quantitative Properties and Reasoning
- ▶ Module 4: Applications to AI/ML

Module 1: Probabilistic models and verification

- ▶ Markov chains: basic model of probabilistic transition system
- ▶ Adding non-determinism: MDPs
- ▶ Reachability and other problems
- ▶ Different algorithms and techniques: two tools - PRISM/STORM
- ▶ State-based and distribution-based views.

Module 2: Quantitative Automata models and verification

- ▶ Timed automata
 - ▶ Event-Clock automata
 - ▶ Time-Travel Automata

Module 2: Quantitative Automata models and verification

- ▶ Timed automata
 - ▶ Event-Clock automata
 - ▶ Time-Travel Automata (or, more boringly, Generalized Timed Automata)

Module 2: Quantitative Automata models and verification

- ▶ Timed automata
 - ▶ Event-Clock automata
 - ▶ Time-Travel Automata (or, more boringly, Generalized Timed Automata)
 - ▶ Tools: Uppaal (state-of-the-art) vs TChecker (Open source tool)

Module 2: Quantitative Automata models and verification

- ▶ Timed automata
 - ▶ Event-Clock automata
 - ▶ Time-Travel Automata (or, more boringly, Generalized Timed Automata)
 - ▶ Tools: Uppaal (state-of-the-art) vs TChecker (Open source tool)
- ▶ Hybrid automata

Module 2: Quantitative Automata models and verification

- ▶ Timed automata
 - ▶ Event-Clock automata
 - ▶ Time-Travel Automata (or, more boringly, Generalized Timed Automata)
 - ▶ Tools: Uppaal (state-of-the-art) vs TChecker (Open source tool)
- ▶ Hybrid automata
- ▶ Probabilistic automata
- ▶ Weighted automata

Module 2: Quantitative Automata models and verification

- ▶ Timed automata
 - ▶ Event-Clock automata
 - ▶ Time-Travel Automata (or, more boringly, Generalized Timed Automata)
 - ▶ Tools: Uppaal (state-of-the-art) vs TChecker (Open source tool)
- ▶ Hybrid automata
- ▶ Probabilistic automata
- ▶ Weighted automata
- ▶ Algorithms and hardness!

Module 3: Quantitative Properties and Reasoning

- ▶ Quantitative Logics
 - ▶ Time temporal logics,
 - ▶ Probabilistic computation-tree logic
 - ▶ Logic of intervals

Module 3: Quantitative Properties and Reasoning

- ▶ Quantitative Logics
 - ▶ Time temporal logics,
 - ▶ Probabilistic computation-tree logic
 - ▶ Logic of intervals
- ▶ Quantitative notions of satisfaction and reasoning
 - ▶ does every run satisfy the property?
 - ▶ can we say most runs satisfy?
 - ▶ probabilistic/PAC guarantees?

Module 4: Applications to AI/ML models (and CPS?)

Models

- ▶ Neural networks
- ▶ Decision tree ensembles
- ▶ Others?

Module 4: Applications to AI/ML models (and CPS?)

Models

- ▶ Neural networks
- ▶ Decision tree ensembles
- ▶ Others?

Problems of interest

- ▶ Robustness
- ▶ Sensitivity/Fairness
- ▶ Explainability

Module 4: Applications to AI/ML models (and CPS?)

Models

- ▶ Neural networks
- ▶ Decision tree ensembles
- ▶ Others?

Problems of interest

- ▶ Robustness
- ▶ Sensitivity/Fairness
- ▶ Explainability



Machine Learning and Logic: Fast and Slow Thinking

Moshe Y. Vardi (Rice University)

ICS: Thursday, February 16, 2023 06:30 PM-07:30 PM

[YouTube Video Link](#)

Abstract: Computer science seems to be undergoing a paradigm shift. Much of earlier research was conducted in the framework of well-understood formal models. In contrast, some of the hottest trends today shun formal models and rely on massive data sets and machine learning. A canonical example of this change is the shift in AI from logic programming to deep learning. I will argue that the correct metaphor for this development is not paradigm shift, but paradigm expansion. Just as General Relativity augments Newtonian Mechanics, rather than replace it -- we went to the moon, after all, using Newtonian Mechanics -- data-driven computing augments model-driven computing. In the context of Artificial Intelligence, machine learning and logic correspond to the two modes of human thinking: fast thinking and slow thinking. The challenge today is to integrate the model-driven and data-driven paradigms. I will describe one approach to such an integration -- making logic more quantitative.

Course logistics

Mode of instruction

- ▶ Slides/board lectures: 2.5-3 hrs per week (at least for 1st two modules)

Course logistics

Mode of instruction

- ▶ Slides/board lectures: 2.5-3 hrs per week (at least for 1st two modules)
- ▶ Tutorials/Doubt clearance sessions: ~ 0.5 -1 hr per week

Course logistics

Mode of instruction

- ▶ Slides/board lectures: 2.5-3 hrs per week (at least for 1st two modules)
- ▶ Tutorials/Doubt clearance sessions: ~ 0.5 -1 hr per week
- ▶ Weekly reading & writing assignments, self study: 3 hrs per week!

Course logistics

Mode of instruction

- ▶ Slides/board lectures: 2.5-3 hrs per week (at least for 1st two modules)
- ▶ Tutorials/Doubt clearance sessions: ~ 0.5 -1 hr per week
- ▶ Weekly reading & writing assignments, self study: 3 hrs per week!
- ▶ Office hours for further doubts (to be fixed)

Course logistics

Mode of instruction

- ▶ Slides/board lectures: 2.5-3 hrs per week (at least for 1st two modules)
- ▶ Tutorials/Doubt clearance sessions: ~ 0.5 -1 hr per week
- ▶ Weekly reading & writing assignments, self study: 3 hrs per week!
- ▶ Office hours for further doubts (to be fixed)
- ▶ Piazza/Teams for online discussion (to be set up)
 - ▶ All course-related info will only be through this mode.
 - ▶ References (Papers/book/videos) will be provided for each module...

Course logistics

Mode of instruction

- ▶ Slides/board lectures: 2.5-3 hrs per week (at least for 1st two modules)
- ▶ Tutorials/Doubt clearance sessions: ~ 0.5 -1 hr per week
- ▶ Weekly reading & writing assignments, self study: 3 hrs per week!
- ▶ Office hours for further doubts (to be fixed)
- ▶ Piazza/Teams for online discussion (to be set up)
 - ▶ All course-related info will only be through this mode.
 - ▶ References (Papers/book/videos) will be provided for each module...

Assessment/Evaluation Disclaimer: Tentative!

Course logistics

Mode of instruction

- ▶ Slides/board lectures: 2.5-3 hrs per week (at least for 1st two modules)
- ▶ Tutorials/Doubt clearance sessions: ~ 0.5 -1 hr per week
- ▶ Weekly reading & writing assignments, self study: 3 hrs per week!
- ▶ Office hours for further doubts (to be fixed)
- ▶ Piazza/Teams for online discussion (to be set up)
 - ▶ All course-related info will only be through this mode.
 - ▶ References (Papers/book/videos) will be provided for each module...

Assessment/Evaluation Disclaimer: Tentative!

- ▶ Quizzes: 30%
- ▶ Paper presentation (in lieu of midsem): $25 \pm 5\%$
- ▶ Final exam : 35%

Course logistics

Mode of instruction

- ▶ Slides/board lectures: 2.5-3 hrs per week (at least for 1st two modules)
- ▶ Tutorials/Doubt clearance sessions: ~ 0.5 -1 hr per week
- ▶ Weekly reading & writing assignments, self study: 3 hrs per week!
- ▶ Office hours for further doubts (to be fixed)
- ▶ Piazza/Teams for online discussion (to be set up)
 - ▶ All course-related info will only be through this mode.
 - ▶ References (Papers/book/videos) will be provided for each module...

Assessment/Evaluation Disclaimer: Tentative!

- ▶ Quizzes: 30%
- ▶ Paper presentation (in lieu of midsem): $25 \pm 5\%$
- ▶ Final exam : 35%
- ▶ Other: Class participation, interactions : $0 \pm 10\%$

Pre-requisites

For UG CSE students

For PG CSE students

For non-CSE students

For Audit students

Pre-requisites

For UG CSE students

- ▶ Discrete structures: CS105/CS207
- ▶ Logic: CS 228
- ▶ (Optional): Automata course or RnD project in related theory topics
- ▶ Self-reading will be required in some topics (so should you do it later? – upto you!)
- ▶ (If your grades in these courses are < 7 , meet me after class.)

For PG CSE students

For non-CSE students

For Audit students

Pre-requisites

For UG CSE students

For PG CSE students

- ▶ Discrete structures in UG
- ▶ Automata in UG
- ▶ (Optional): Elective/RnD project in related theory topics (e.g., CS 771)

For non-CSE students

For Audit students

Pre-requisites

For UG CSE students

For PG CSE students

For non-CSE students

- ▶ Come meet me after class.

For Audit students

Pre-requisites

For UG CSE students

For PG CSE students

For non-CSE students

For Audit students

- ▶ Attendance + 1 presentation