CS 735: Formal Models for Concurrent and Asynchronous Systems

– Introduction

Instructor : Akshay S

Jan 5, 2016

Course hours: Slot11,
Tuesdays and Fridays 3:30-5:00pm

CS 735: Formal Models for Concurrent and Asynchronous Systems

– Introduction

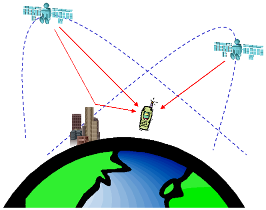
Instructor : Akshay S

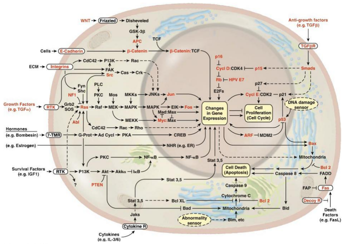
Jan 5, 2016

Course hours: Slot11,
Tuesdays and Fridays 3:30-5:00pm
To be changed?

Goal

Introduce Formal Models for distributed systems

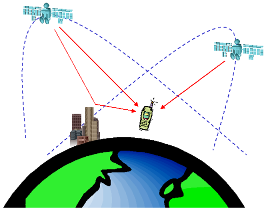


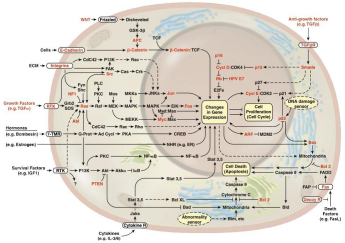


Goal

Introduce Formal Models for distributed systems

- Distributed: Concurrent, asynchronous, communicating,...

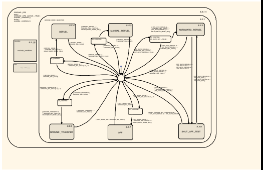


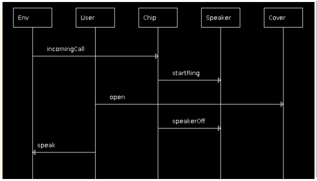


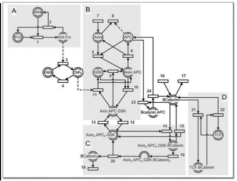
Goal

Introduce Formal Models for distributed systems

- Distributed: Concurrent, asynchronous, communicating,...
- Formal models: Mathematical description, graphical notations, Automata models







Goal

Introduce Formal Models for distributed systems

- Distributed: Concurrent, asynchronous, communicating,...
- Formal models: Mathematical description, graphical notations, Automata models

Questions that we will tackle

- Analysis of such models
- Characterization, relations
- Underlying properties, generalizations

Course contents

Topics and models that we will cover in this course:

1. Petri nets
2. Well-structured transition systems
3. Distributed automata models and their behaviors
4. Extensions and recent advances

Course contents

Topics and models that we will cover in this course:

1. Petri nets
   - Elementary nets, Place/Transition nets
   - Behaviors - traces, posets, unfoldings.
   - Decision problems - reachability, coverability
   - Tools, implementations and case-studies
2. Well-structured transition systems
3. Distributed automata models and their behaviors
4. Extensions and recent advances

Course contents

Topics and models that we will cover in this course:

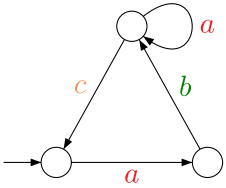
1. Petri nets
   - Elementary nets, Place/Transition nets
   - Behaviors - traces, posets, unfoldings.
   - Decision problems - reachability, coverability
   - Tools, implementations and case-studies
2. Well-structured transition systems
   - A generalized abstraction for infinite-state systems
   - Well-quasi orders and well-founded systems
   - Applications to show termination of infinite systems
   - Theoretical bounds on complexity
3. Distributed automata models and their behaviors
4. Extensions and recent advances

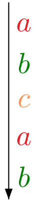
Course contents

Topics and models that we will cover in this course:

1. Petri nets
   - Elementary nets, Place/Transition nets
   - Behaviors - traces, posets, unfoldings.
   - Decision problems - reachability, coverability
   - Tools, implementations and case-studies
2. Well-structured transition systems
   - A generalized abstraction for infinite-state systems
   - Well-quasi orders and well-founded systems
   - Applications to show termination of infinite systems
   - Theoretical bounds on complexity
3. Distributed automata models and their behaviors
   - Asynchronous automata
   - Message passing automata: Also called Communicating finite-state machines
   - Lossy channel machines
4. Extensions and recent advances

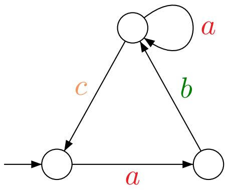
Automata

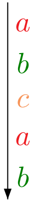




- Behaviours are words, i.e., sequences of actions over a finite alphabet $\Sigma = \{a, b, c\}$.

Automata



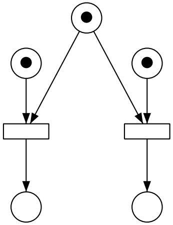


- Behaviours are words, i.e., sequences of actions over a finite alphabet $\Sigma = \{a, b, c\}$.

Questions

- How shall we distribute it?
- How shall we add concurrent behaviors?

Petri Nets

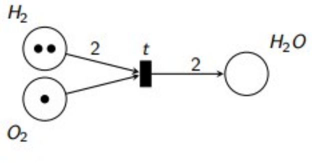


- An old model for distributed systems
  - invented by Carl Petri (-at the age of 13- in 1939? or ’62)
  - to model resource consumption and so on...

Examples of Petri nets

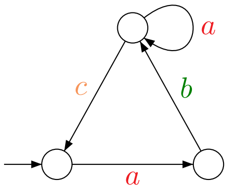
- A chemical reaction: $2H_2 + O_2 \to 2H_2O$.
- A library
- A producer-consumer example
- A coffee machine

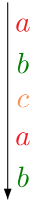
Examples of Petri nets



- A chemical reaction: $2H_2 + O_2 \to 2H_2O$.
- A library
- A producer-consumer example
- A coffee machine

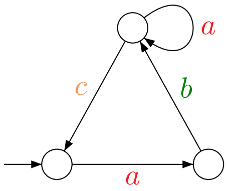
Automata

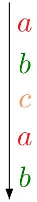




- Behaviours are words, i.e., sequences of actions over a finite alphabet $\Sigma = \{a, b, c\}$.

Automata



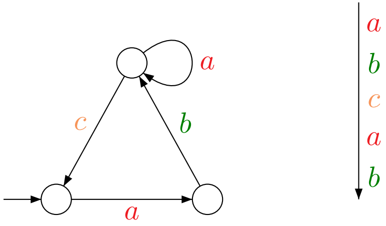


- Behaviours are words, i.e., sequences of actions over a finite alphabet $\Sigma = \{a, b, c\}$.

Questions

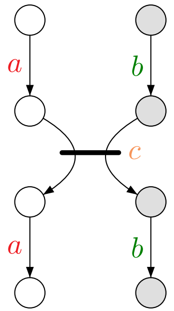
- How shall we distribute it?
- How shall we add concurrent behaviors?

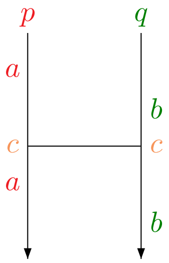
Asynchronous Automata



- Behaviours are words, i.e., sequences of actions over a finite alphabet $\Sigma = \{a, b, c\}$.

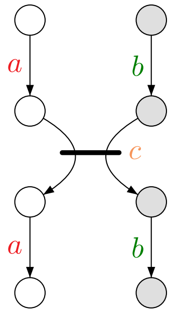
Asynchronous Automata

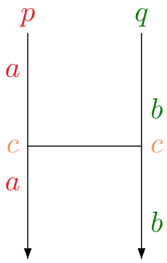




- Actions are distributed across processes (with sharing!)
- Some actions are shared, e.g., c is allowed only if both p and q move on c.

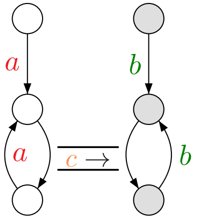
Asynchronous Automata

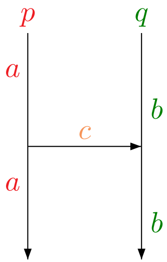




- What are the properties of languages accepted by such automata? E.g. above accepts $\{\underline{abcab}, bacab, bacba, abcba\}$.
- Given a language L, (when) can it be accepted by such an asynchronous automaton?

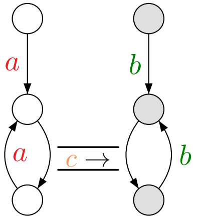
Message Passing Automata

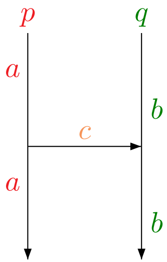




- In fact, this formalism is Turing powerful!
- We will consider decidability issues.

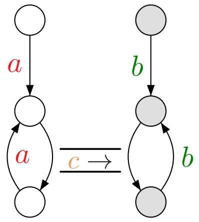
Message Passing Automata

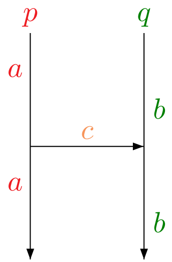




- In fact, this formalism is Turing powerful!
- We will consider decidability issues.
- (Surprising fact: If you are allowed to lose messages randomly then it is decidable!)

Message Passing Automata





- In fact, this formalism is Turing powerful!
- We will consider decidability issues.
- (Surprising fact: If you are allowed to lose messages randomly then it is decidable!) These are called Lossy channel systems.

Distributing automata

- Take products of automata with shared/communicated actions

Distributing automata

- Take products of automata with shared/communicated actions
- Consider local states and local transitions and see how they evolve

Generalizing...

- In general, these are examples of infinite-state objects.

Generalizing...

- In general, these are examples of infinite-state objects.
- If you don't like "state objects", think of them as infinite discrete structures!

Generalizing...

- In general, these are examples of infinite-state objects.
- If you don't like "state objects", think of them as infinite discrete structures!
- Why?

Generalizing...

- In general, these are examples of infinite-state objects.
- If you don't like "state objects", think of them as infinite discrete structures!
- Why?

Another title for this course:

Reasoning about infinite (discrete) structures!

Generalizing...

- In general, these are examples of infinite-state objects.
- If you don't like "state objects", think of them as infinite discrete structures!
- Why?

Another title for this course:

Reasoning about infinite (discrete) structures!

- Theory of well-structured transition systems
- Under-approximate verification
- Fixed-point approaches

Generalizing...

- In general, these are examples of infinite-state objects.
- If you don't like "state objects", think of them as infinite discrete structures!
- Why?

Another title for this course:

Reasoning about infinite (discrete) structures!

- Theory of well-structured transition systems
- Under-approximate verification
- Fixed-point approaches

Pictures and Mathematics

- How do you write these objects mathematically?
- Why write them mathematically?

Some take-aways from this course

- Different formal models for distributed systems
- Mathematical formalisms that reason about (the infinite) behaviors of such systems.
- Techniques to automatically analyze such systems.
- How to use them and where they are applied.

Logistics

Evaluation (flexible... upto a point)

- Continuous evaluation - assignments/quizzes : 40%
- Exam (Midsem/Endsem): 35 %
- Paper presentation: 25 %

There will be guest lectures, research directions given along the way.

Logistics

Evaluation (flexible... upto a point)

- Continuous evaluation - assignments/quizzes : 40%
- Exam (Midsem/Endsem): 35 %
- Paper presentation: 25 %

There will be guest lectures, research directions given along the way.

Course material, references will be posted at

- `http://www.cse.iitb.ac.in/~akshayss/teaching.html`
- Piazza will be set up soon.