Systems Design Laboratory

Teaser

Matteo Zavatteri

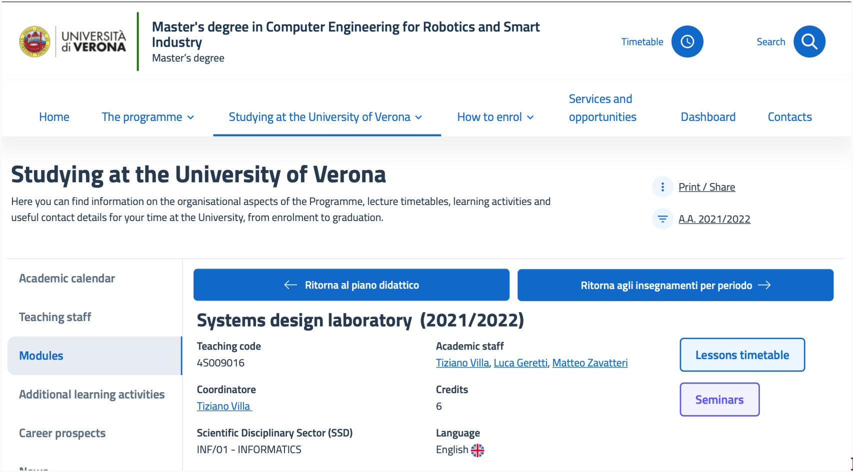
Department of Computer Science, University of Verona, ITALY

First and Most Important

Systems Design Laboratory is not another theory course!

Essential Information: Homepage and Academic Staff

Essential Information: Homepage



Master's degree in Computer Engineering for Robotics and Smart Industry
Master's degree

Studying at the University of Verona

Here you can find information on the organisational aspects of the Programme, lecture timetables, learning activities and useful contact details for your time at the University, from enrolment to graduation.

Systems design laboratory (2021/2022)

Teaching code
4S009016

Academic staff
Tiziano Villa, Luca Geretti, Matteo Zavatteri

Coordinatore
Tiziano Villa

Credits
6

Scientific Disciplinary Sector (SSD)
INF/01 - INFORMATICS

Language
English

http
//www.corsi.univr.it/?ent=cs&aa=2021%2F2022&codiceCs=S81&codins=
4S009016&discr=&discrCd=&id=954&menu=Studiare&tab=Insegnamenti&lang=en

Essential Information: Academic Staff

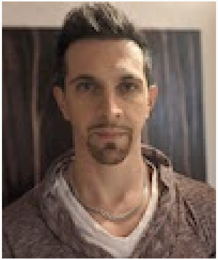
Tiziano Villa (coordinator)

- Formal models for discrete-event systems
- Boolean functions and networks
- Decision diagrams
- Multiple-valued and temporal logics
- Finite and ω-automata

https://www.di.univr.it/?ent=persona&id=3849&lang=en

Essential Information: Academic Staff

Luca Geretti



- Modeling with Hybrid Automata
- Static verification
- Dynamic verification

https://www.di.univr.it/?ent=persona&id=6462&lang=en

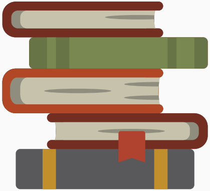
Essential Information: Academic Staff

Matteo Zavatteri

- Modeling with Finite State Automata
- Supervisory Control
- ESCET software

https://www.di.univr.it/?ent=persona&id=20745&lang=en

Essential Information: Educational Material



- Books and papers (some already suggested in previous classes)
- Lecture notes and classroom teaching material
- Reports on the analysis of case studies
- Online documentation and tutorials
- Example code

Essential Information: Tentative Setup

Fully Controllable Setting: We are probably going to meet in a lab but you will do everything on your own computer.
(Fewer problems, no UniVR/IT dependencies).

Essential Information: Exam

Mainly a project

(possibly a short written test depending on the number of students)

Essential Information: Take Home Message

Systems Design Laboratory:

- is a “**hands-on**” modeling course
- provides you with **concrete skills**
- involves the use of several **software libraries**
- is about **solving concrete problems**

Last but not least: plenty of room for **theses**, research, and more (e.g., the ICE lab context)

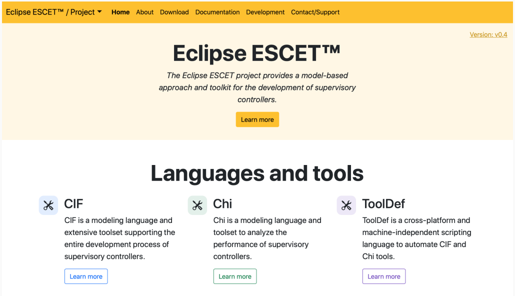
https://www.icelab.di.univr.it/?lang=en

Today's Teaser: Supervisory Control Part

Today you are going to see an example of concrete supervisory control application

- **Modeling** plants/specifications through **Finite State Automata**
- Synthesizing **controllers** for it with **ESCET**
- **Simulating** the controlled plants

Eclipse Supervisory Control Engineering Toolkit (ESCET)



Eclipse ESCET™

The Eclipse ESCET project provides a model-based approach and toolkit for the development of supervisory controllers.

Languages and tools

CIF

CIF is a modeling language and extensive toolset supporting the entire development process of supervisory controllers.

Chi

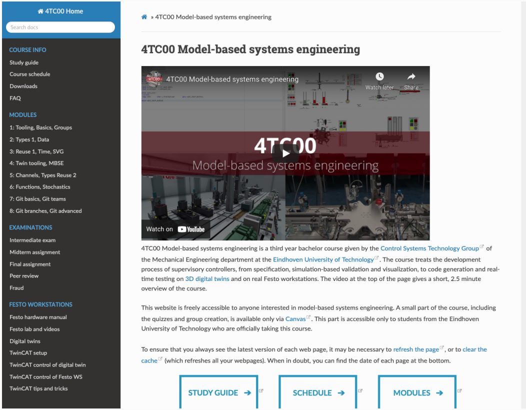
Chi is a modeling language and toolset to analyze the performance of supervisory controllers.

ToolDef

ToolDef is a cross-platform and machine-independent scripting language to automate CIF and Chi tools.

- **Modeling** of hybrid systems
- **Graphical user interface**
- **Simulation**
- **Controller synthesis** for (Extended) Finite State Automata
- **PLC code** generation
- Used in many **real-word case studies**

Eclipse Supervisory Control Engineering Toolkit (ESCET)



4TC00 Model-based systems engineering

4TC00 Model-based systems engineering is a third year bachelor course given by the Control Systems Technology Group of the Mechanical Engineering department at the Eindhoven University of Technology. The course treats the development process of supervisory controllers, from specification, simulation-based validation and visualization, to code generation and real-time testing on 3D digital twins and on real Festo workstations. The video at the top of the page gives a short, 2.5 minute overview of the course.

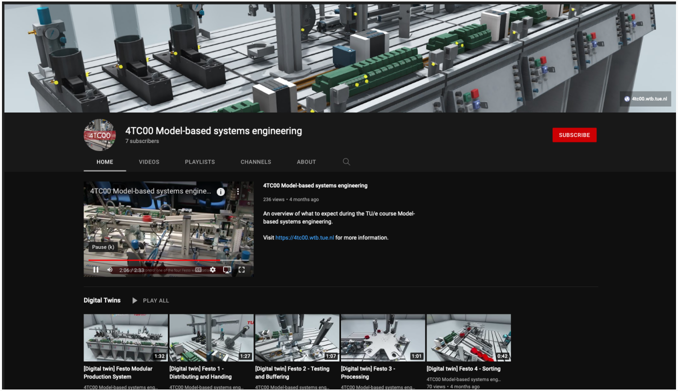
This website is freely accessible to anyone interested in model-based systems engineering. A small part of the course, including the quizzes and group creation, is available only via Canvas. This part is accessible only to students from the Eindhoven University of Technology who are officially taking this course.

To ensure that you always see the latest version of each web page, it may be necessary to refresh the page, or to clear the cache (which refreshes all your webpages). When in doubt, you can find the date of each page at the bottom.

Introduced in the course 4TC00 Model-Based Systems Engineering (bachelor degree, 3rd year) Eindhoven University of Technology (TU/e)

https://cstweb.wtb.tue.nl/4tc00/index.html

Eclipse Supervisory Control Engineering Toolkit (ESCET)



4TC00 Model-based systems engineering

7 subscribers

4TC00 Model-based systems engineering

236 views • 4 months ago

An overview of what to expect during the TU/e course Model-based systems engineering.

Visit https://4tc00.wtb.tue.nl for more information.

Digital Twins

[Digital twin] Festo Modular Production System
4TC00 Model-based systems eng...

[Digital twin] Festo 1 - Distributing and Handing
4TC00 Model-based systems eng...

[Digital twin] Festo 2 - Testing and Buffering
4TC00 Model-based systems eng...

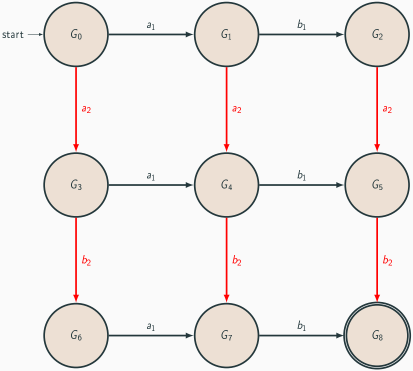
[Digital twin] Festo 3 - Processing
4TC00 Model-based systems eng...

[Digital twin] Festo 4 - Sorting
4TC00 Model-based systems eng...
70 views • 4 months ago

Check out the youtube channel for videos, examples, and more

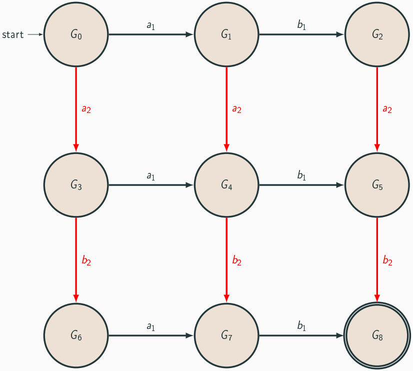
https://www.youtube.com/channel/UC1lkrIkRkgtbYDul9BwI_Bw

The Database Concurrency Example



- Events a_1, b_1 are controllable
- Events a_2, b_2 are uncontrollable
- G_0 is the initial state
- G_8 is the marked state

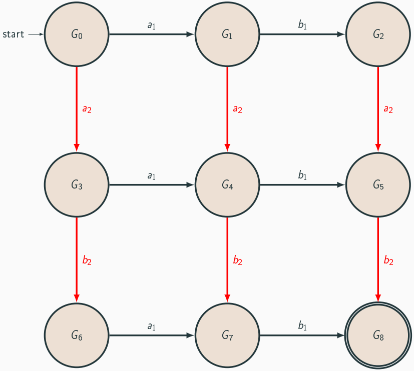
The Database Concurrency Example: Requirement



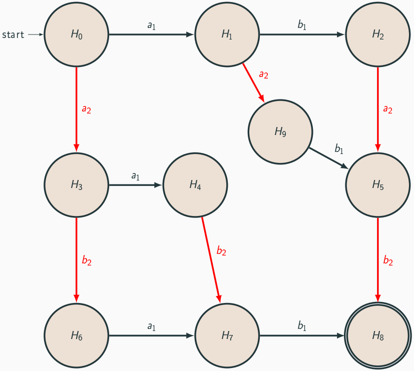
Requirement: a_1 precedes b_1 if and only if a_2 precedes b_2

The Database Concurrency Example: Requirement

Plant

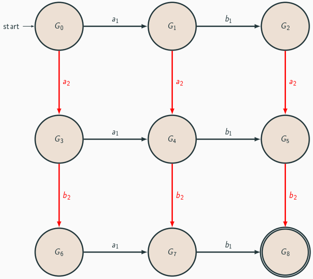


Requirement

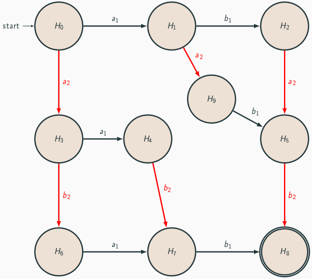


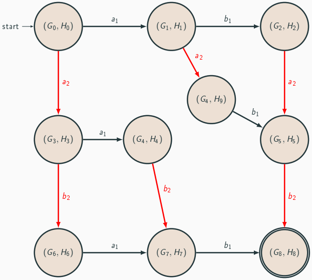
The Database Concurrency Example: Controller Synthesis

Plant



Requirement

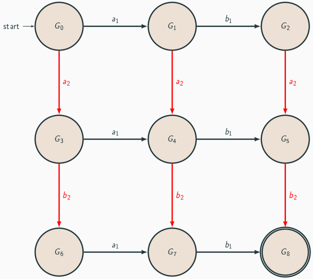




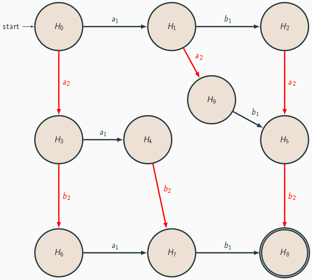
- $G \| H$

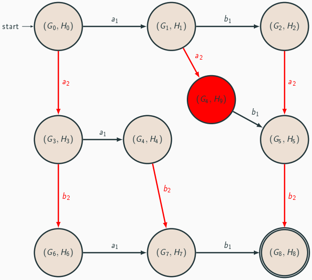
The Database Concurrency Example: Controller Synthesis

Plant



Requirement

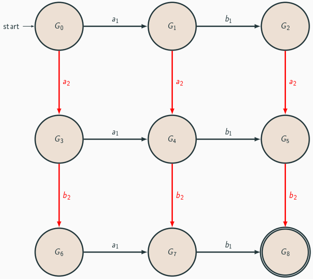




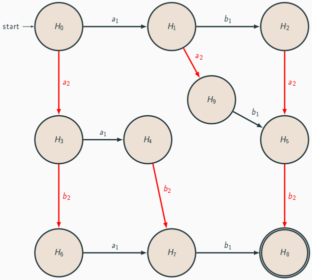
- (G_4, H_9) is uncontrollable

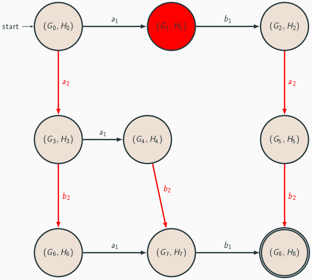
The Database Concurrency Example: Controller Synthesis

Plant



Requirement

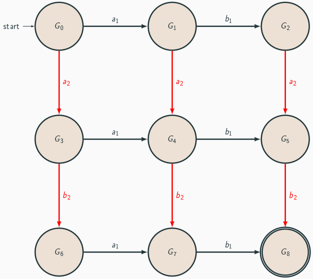




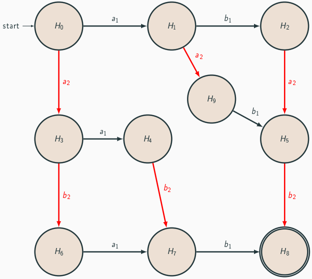
- (G_1, H_1) is uncontrollable

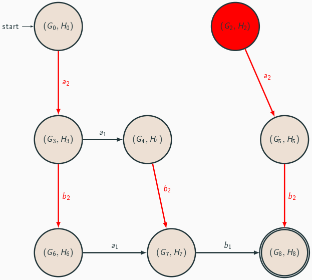
The Database Concurrency Example: Controller Synthesis

Plant



Requirement

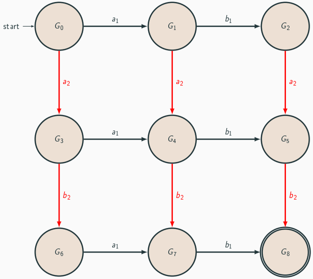




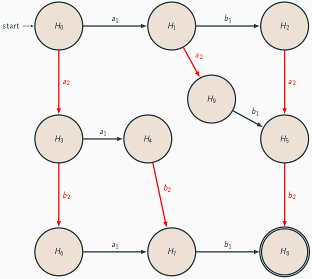
- (G_2, H_2) is not accessible

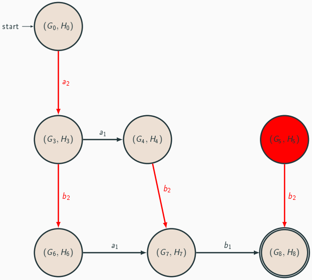
The Database Concurrency Example: Controller Synthesis

Plant



Requirement

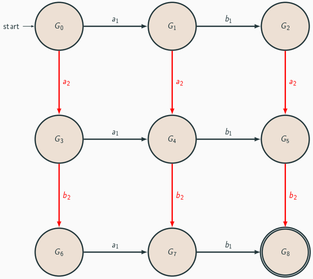




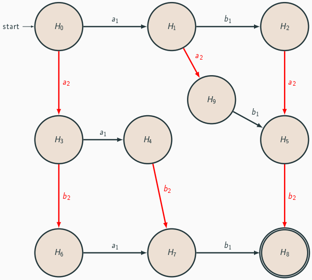
- (G_5, H_5) is not accessible

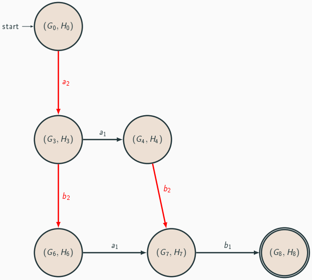
The Database Concurrency Example: Controller Synthesis

Plant



Requirement



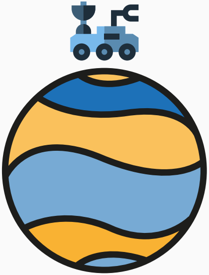


Control Policy:

- At the beginning S disables a_1
- When the **plant** G is in state G_4, S disables b_1.

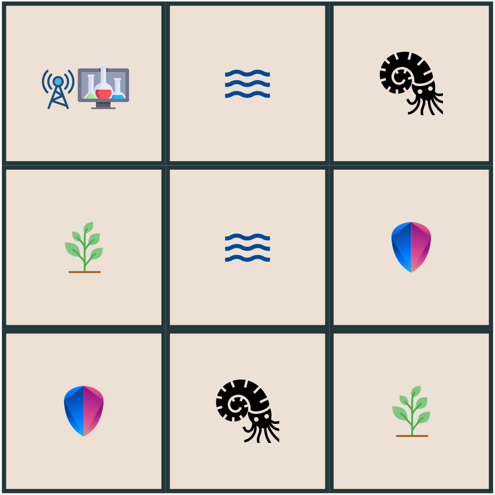
A Sample-Collecting Problem

We sent a rover to some planet.



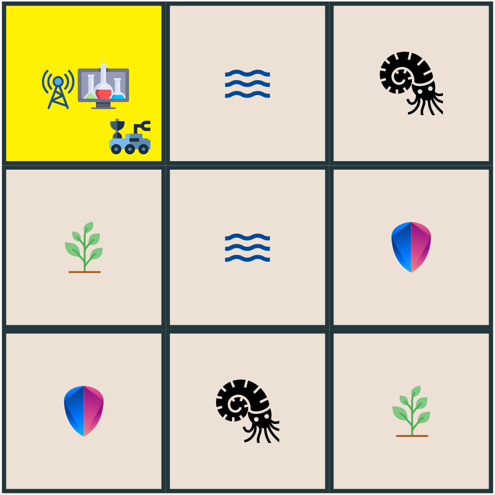
Our purpose is to use it for sample-collecting in the area where it landed.

The Area



- 📡(transmission point)
- 🖥️(analysis point)
- ≋(liquid point)
- 🛡️(mineral point)
- 🌱(flora point)
- 🐚(fossil point)

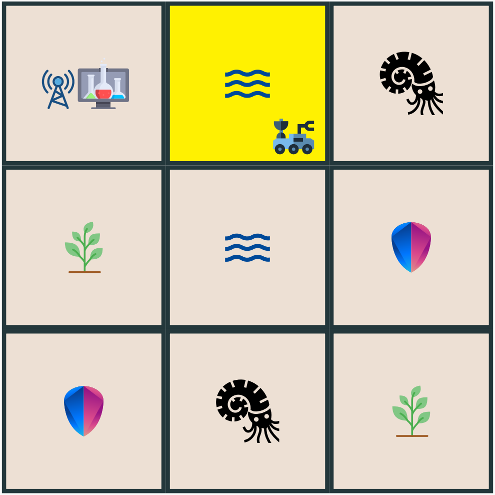
Details on Rover Free Behavior



Possible actions:

- analyze the collected samples
- data transmission
- move right
- move down

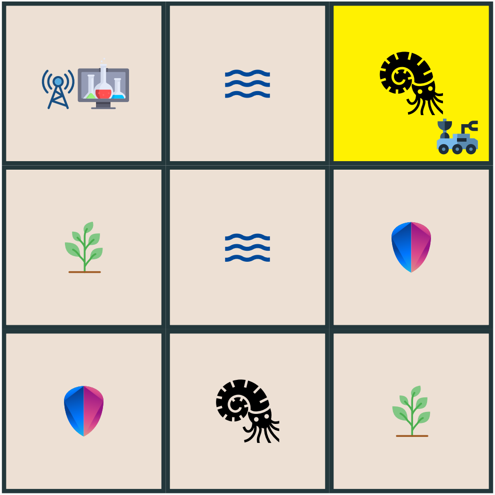
Details on Rover Free Behavior



Possible actions:

- collect a liquid sample
- move left
- move right
- move down

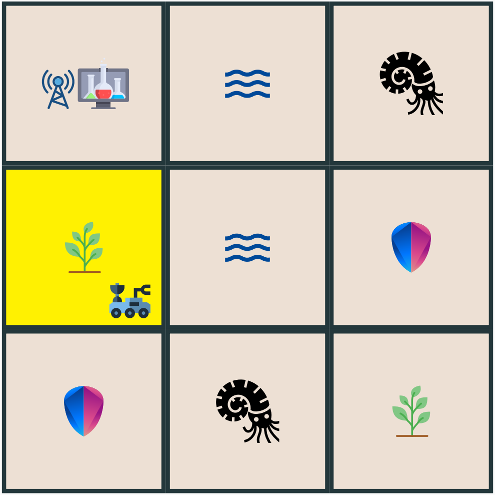
Details on Rover Free Behavior



Possible actions:

- collect a fossil sample
- move left
- move down

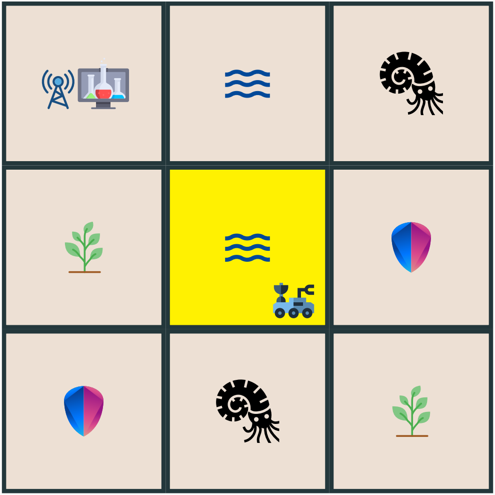
Details on Rover Free Behavior



Possible actions:

- collect a flora sample
- move right
- move up
- move down

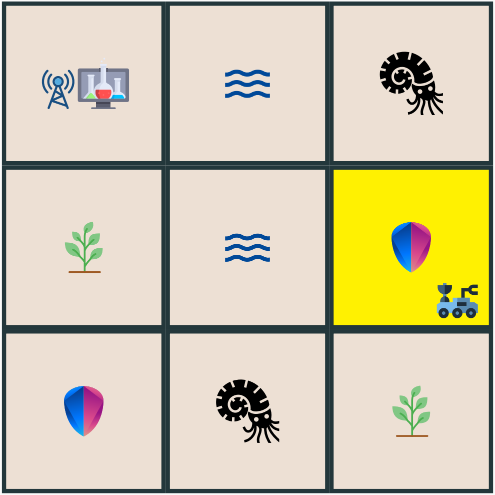
Details on Rover Free Behavior



Possible actions:

- collect a liquid sample
- move left
- move right
- move down
- move up

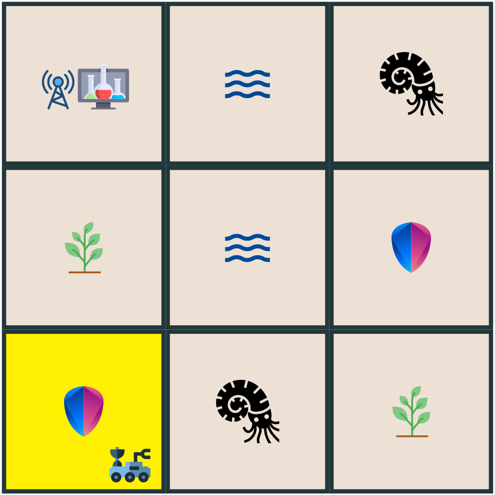
Details on Rover Free Behavior



Possible actions:

- collect a mineral sample
- move up
- move left
- move down

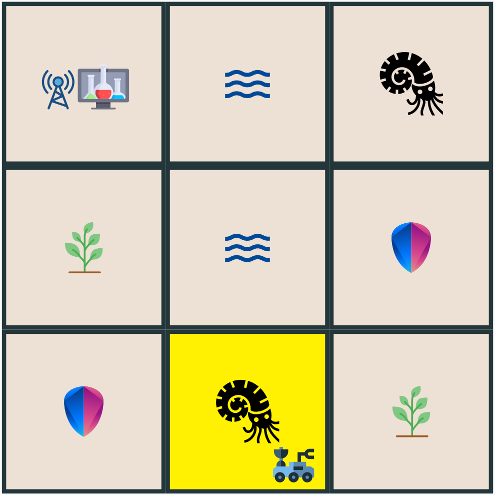
Details on Rover Free Behavior: Fossil points



Possible actions:

- collect a mineral sample
- move up
- move right

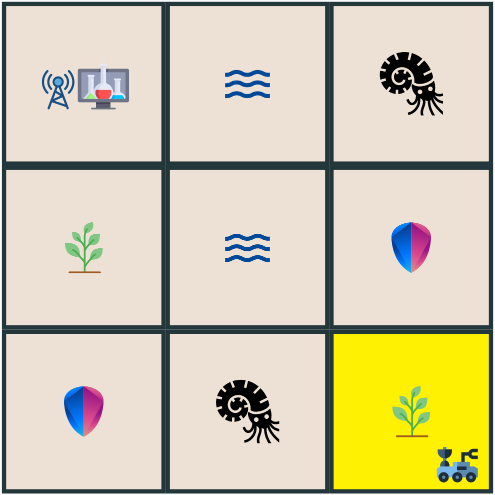
Details on Rover Free Behavior: Fossil points



Possible actions:

- collect a fossil sample
- move up
- move left
- move right

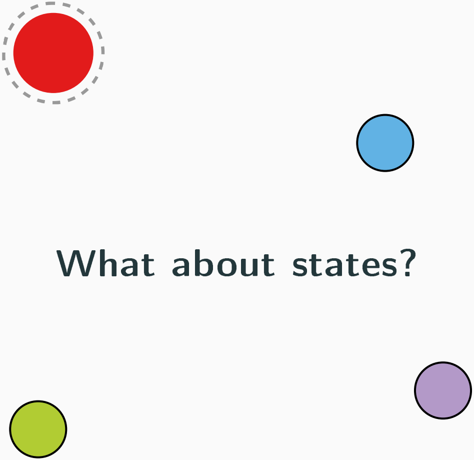
Details on Rover Free Behavior: Fossil points



Possible actions:

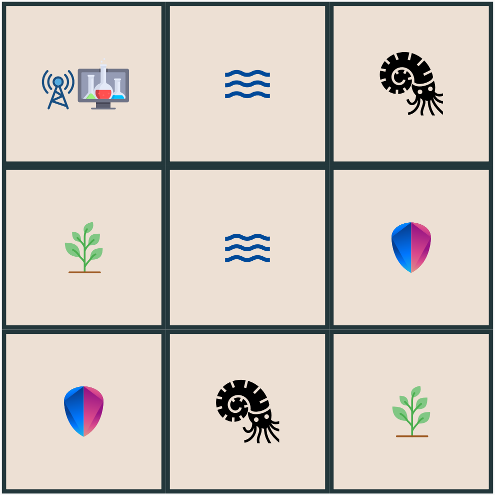
- collect a flora sample
- move up
- move left

Formalizing Plant States

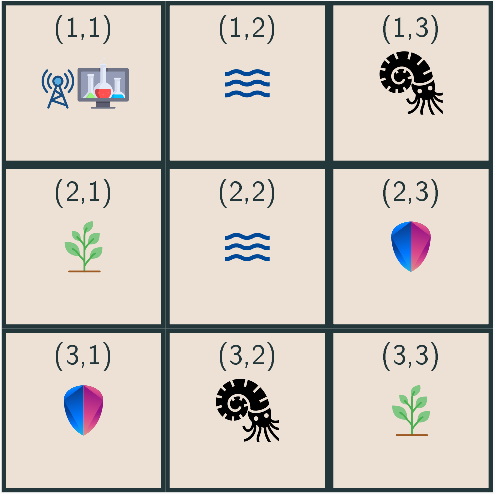


What about states?

Formalizing Plant States



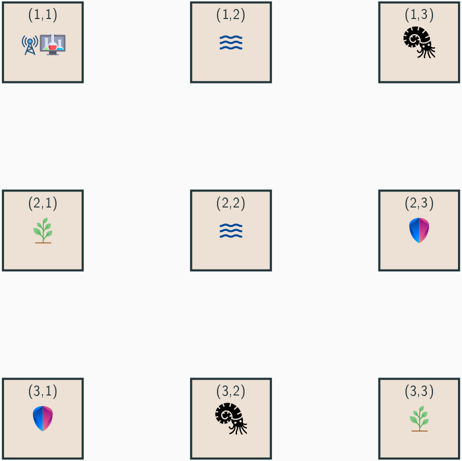
Formalizing Plant States



(1,1)	(1,2)	(1,3)
(2,1)	(2,2)	(2,3)
(3,1)	(3,2)	(3,3)

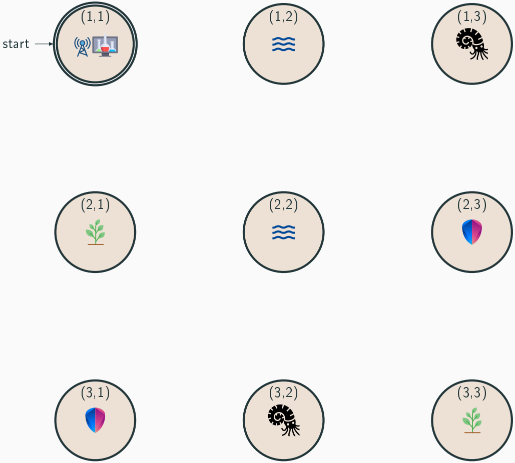
So, it makes sense to introduce a notion of **grid** on which the rover **moves**.

Formalizing Plant States



Does it ring a bell?

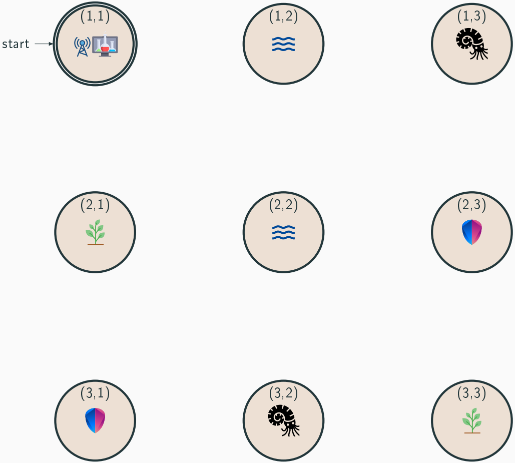
Formalizing Plant States

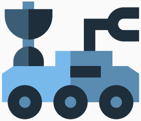


Formalizing Plant Transitions

What about transitions?

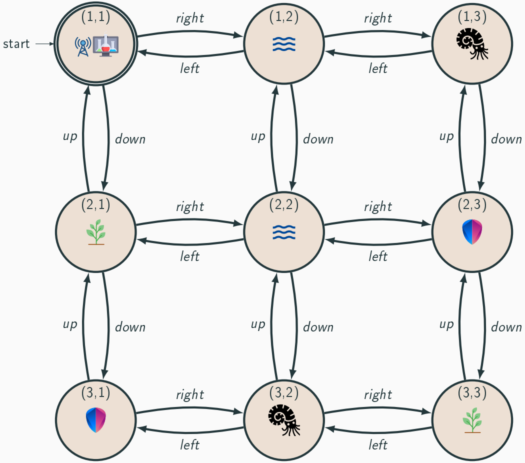
Formalizing Plant Transitions

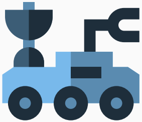




Movements?

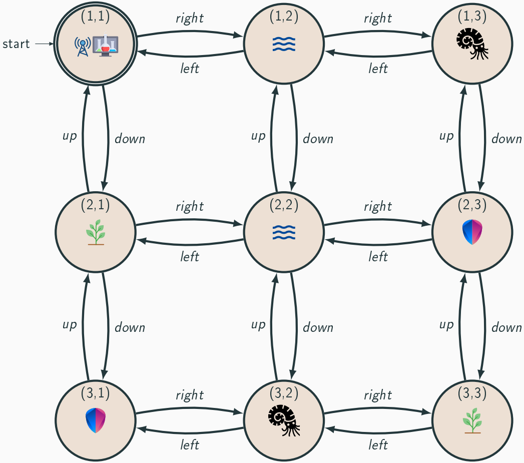
Formalizing Plant Transitions

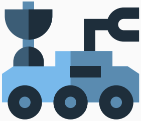




Movements

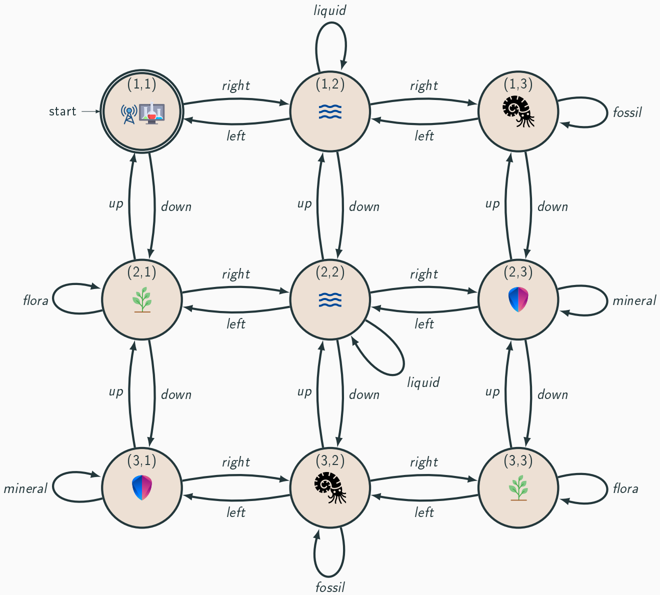
Formalizing Plant Transitions

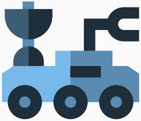




Sample collection?

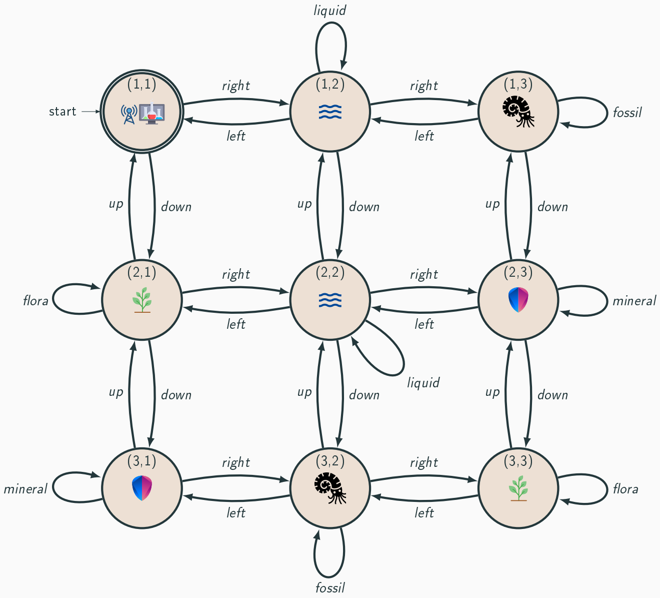
Formalizing Plant Transitions

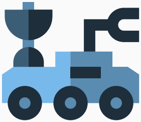




Sample collection

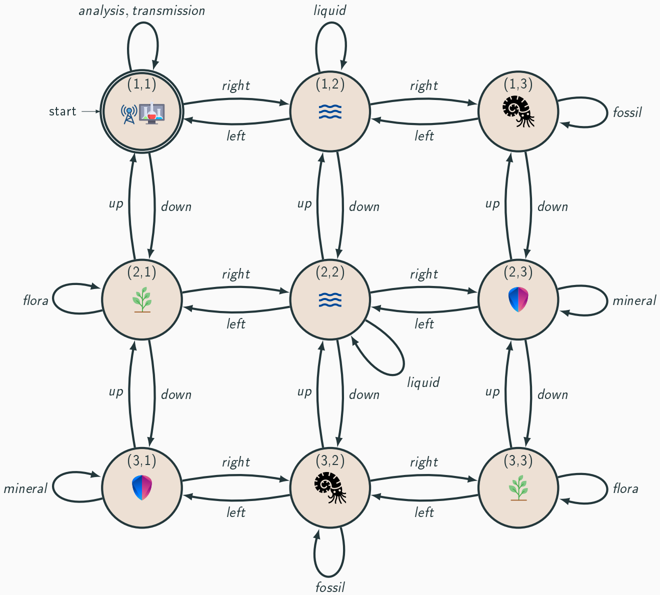
Formalizing Plant Transitions

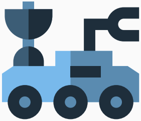




Analysis and Transmission?

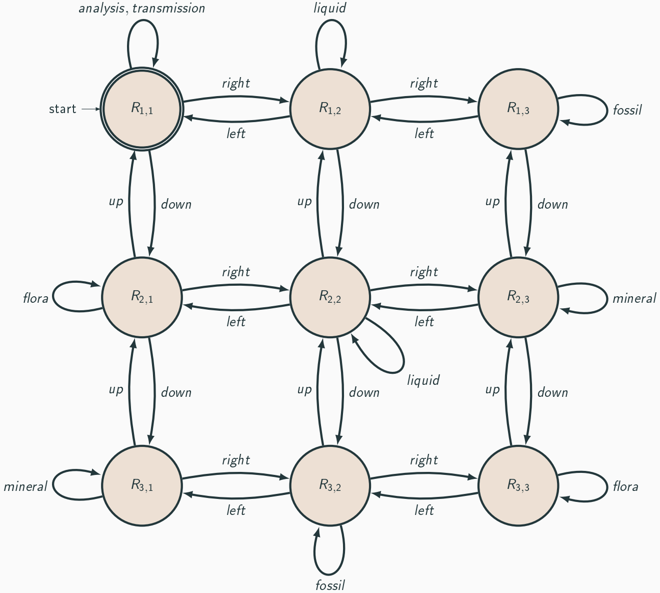
Formalizing Plant Transitions





Analysis and Transmission

Finalizing Plant Formalization: State Names



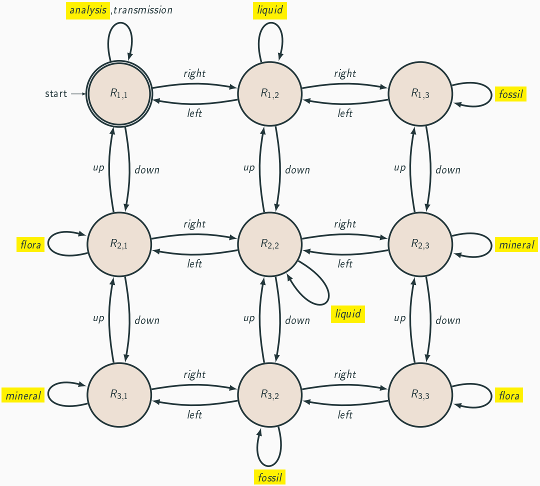
Uncontrolled Rover Behavior

Formalizing Requirements

What about supervisory control?

Requirement 1

Plant



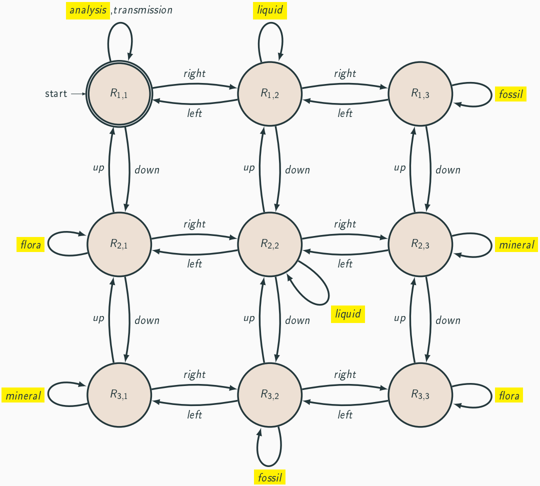
Requirement

- States?
- Events?
- Marking?

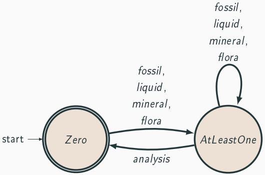
At least 1 sample must be collected before analysis

Requirement 1

Plant



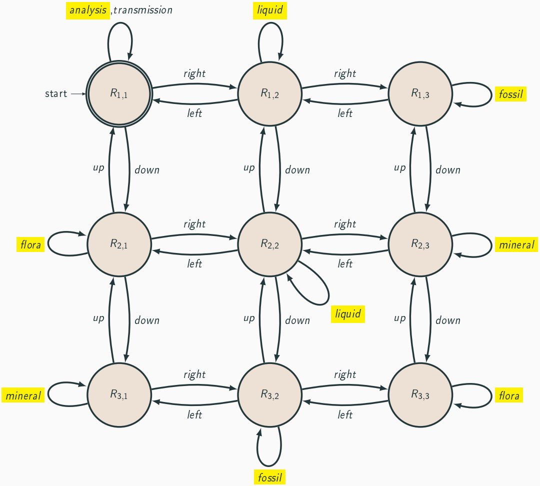
Requirement



At least 1 sample must be collected before analysis

Requirement 2

Plant

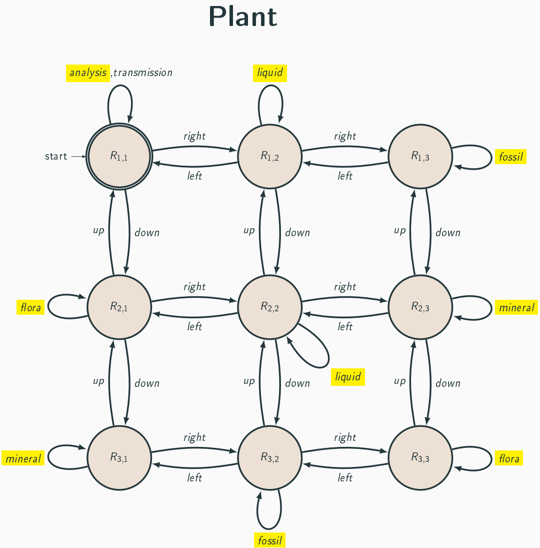


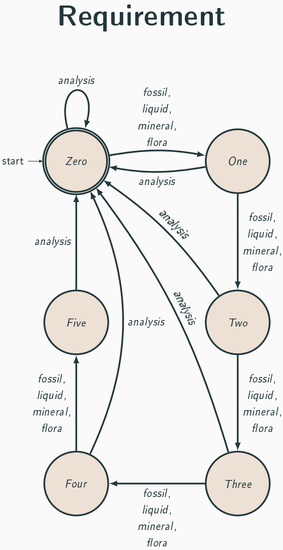
Requirement

- States?
- Events?
- Marking?

At most 5 samples must be collected before analysis

Requirement 2

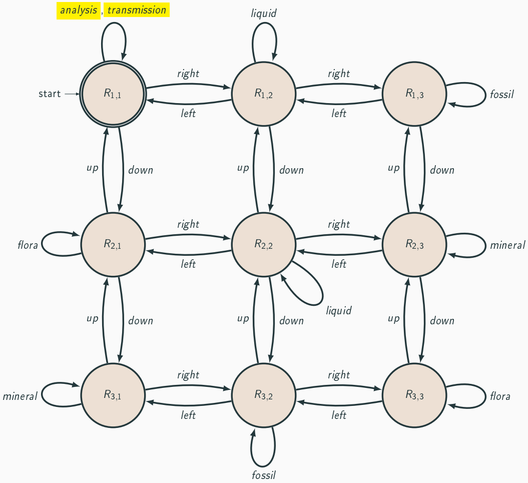



Requirement
analysis
fossil, liquid, mineral, flora
start
Zero
One
analysis
analysis
analysis
fossil, liquid, mineral, flora
Five
analysis
analysis
Two
fossil, liquid, mineral, flora
fossil, liquid, mineral, flora
Four
Three
fossil, liquid, mineral, flora

At most 5 samples must be collected before analysis

Requirement 3

Plant



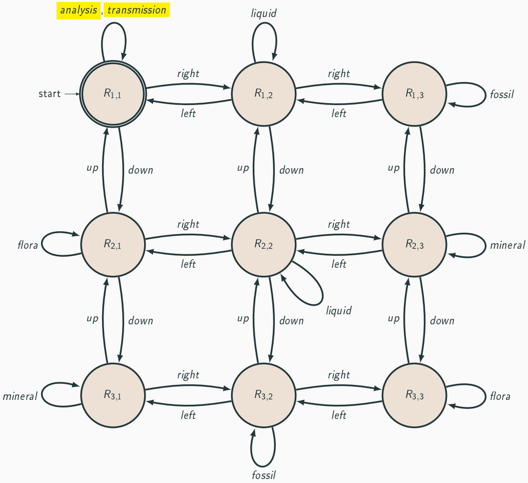
Requirement

- States?
- Events?
- Marking?

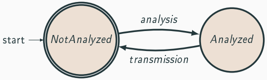
Every analysis must be transmitted exactly once

Requirement 3

Plant



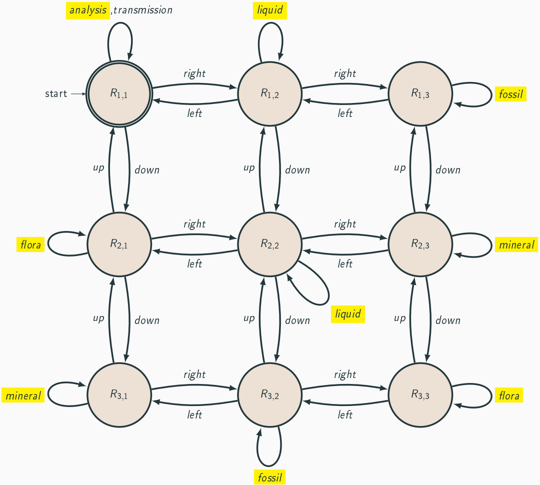
Requirement



Every analysis must be transmitted exactly once

Requirement 4

Plant



Requirement

- States?
- Events?
- Marking?

Exactly 1 liquid, at least 2 fossils, at most 3 florae, from 1 to 3 minerals must be collected before analysis.

Requirement 4: Decomposition

Exactly 1 liquid, at least 2 fossils, at most 3 florae, from 1 to 3 minerals must be collected before analysis:

$\Downarrow$

4a) Exactly 1 liquid must be collected before analysis

4b) At least 2 fossils must be collected before analysis

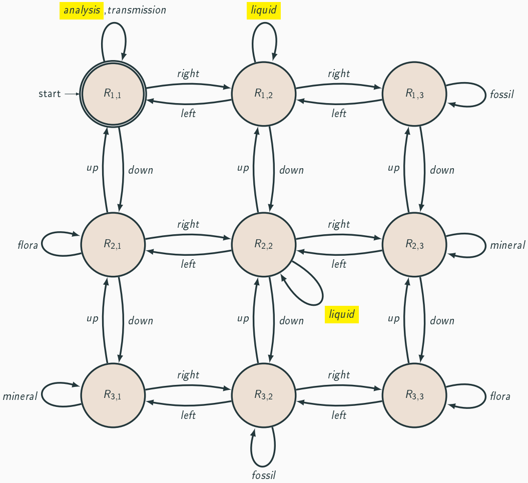
4c) At most 3 florae must be collected before analysis

4d) From 1 to 3 minerals must be collected before analysis

Assumption: Every analysis removes all collected samples from the rover

Requirement 4a

Plant



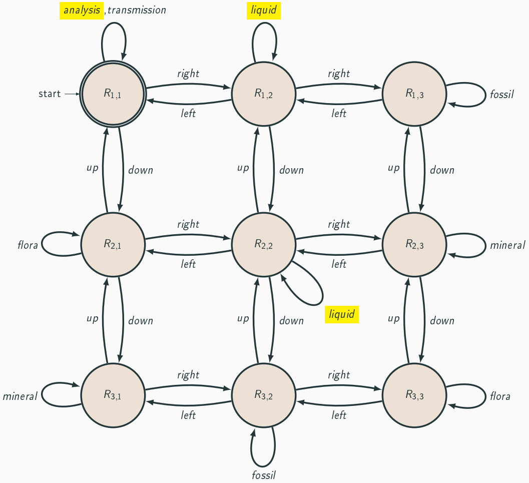
Requirement

- States?
- Events?
- Marking?

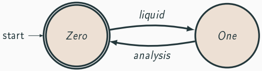
Exactly 1 liquid must be collected before analysis

Requirement 4a

Plant



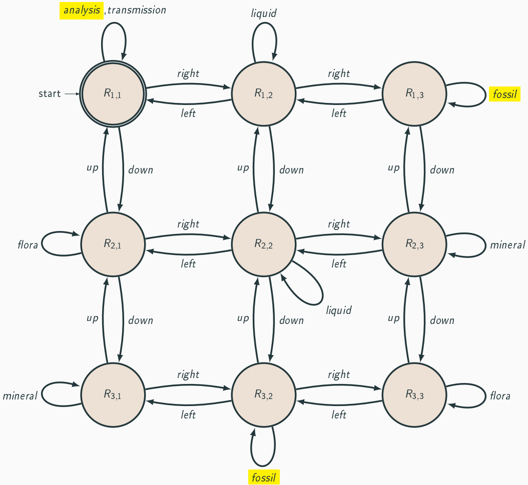
Requirement



Exactly 1 liquid must be collected before analysis

Requirement 4b

Plant



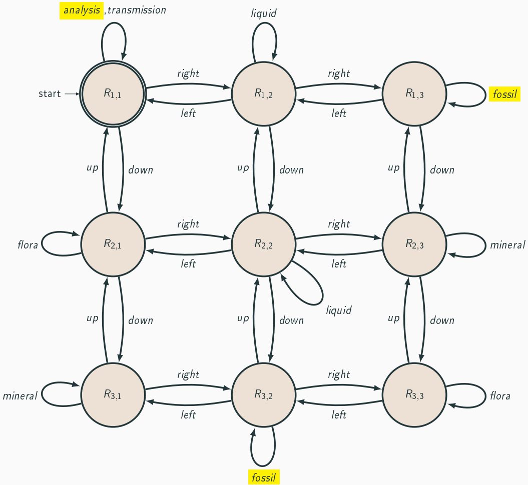
Requirement

- States?
- Events?
- Marking?

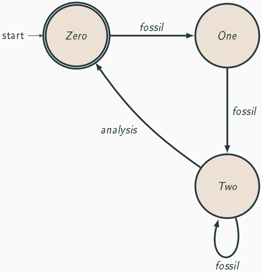
At least 2 fossils must be collected before analysis

Requirement 4b

Plant



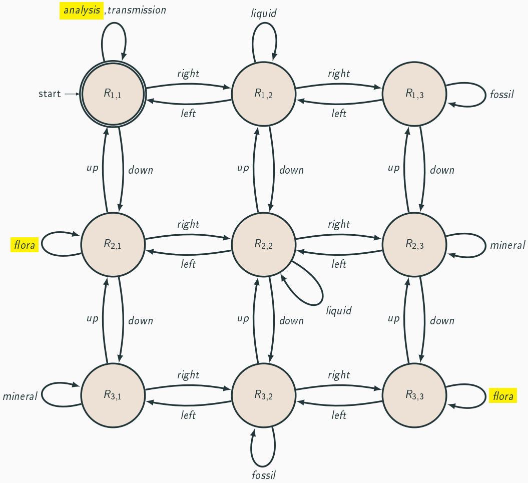
Requirement



At least 2 fossils must be collected before analysis

Requirement 4c

Plant



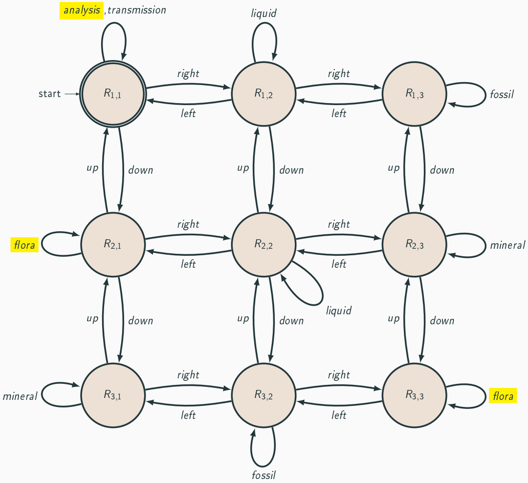
Requirement

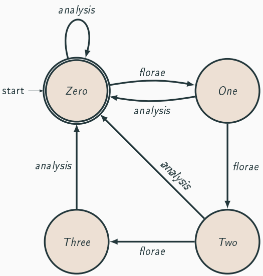
- States?
- Events?
- Marking?

At most 3 florae must be collected before analysis

Requirement 4c

Plant

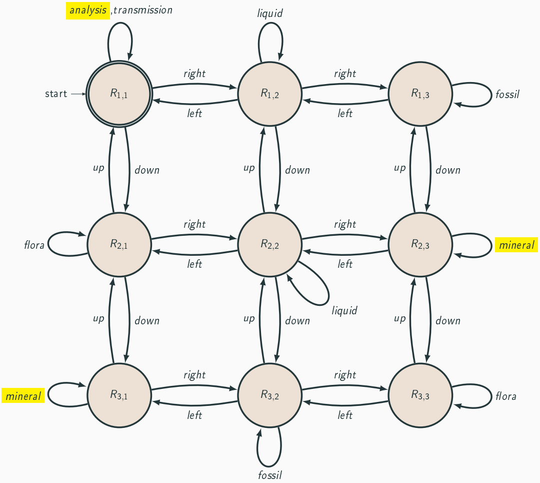




At most 3 florae must be collected before analysis

Requirement 4d

Plant



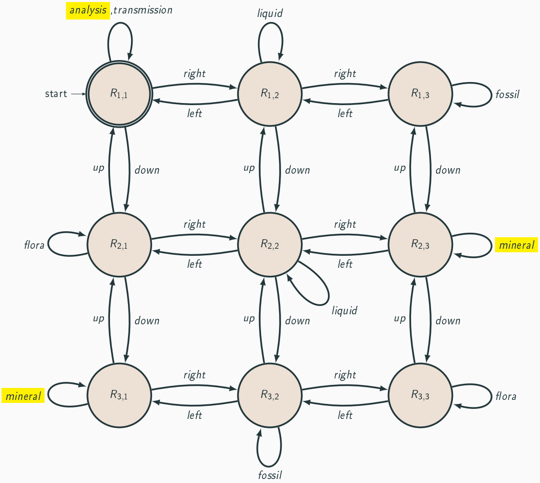
Requirement

- States?
- Events?
- Marking?

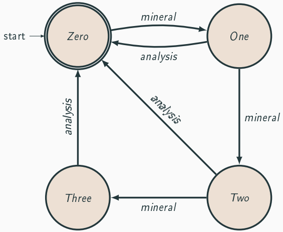
From 1 to 3 minerals must be collected before analysis

Requirement 4d

Plant



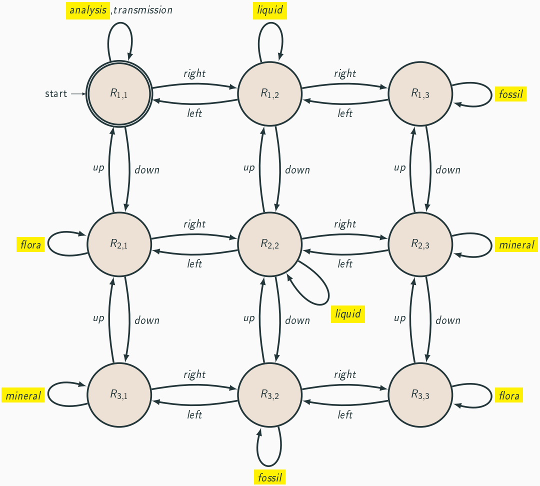
Requirement



From 1 to 3 minerals must be collected before analysis

Requirement 5

Plant



Requirement

- States?
- Events?
- Marking?

At most 1 sample from each subarea containing a fossil must be collected before analysis

Requirement 5: Decomposition

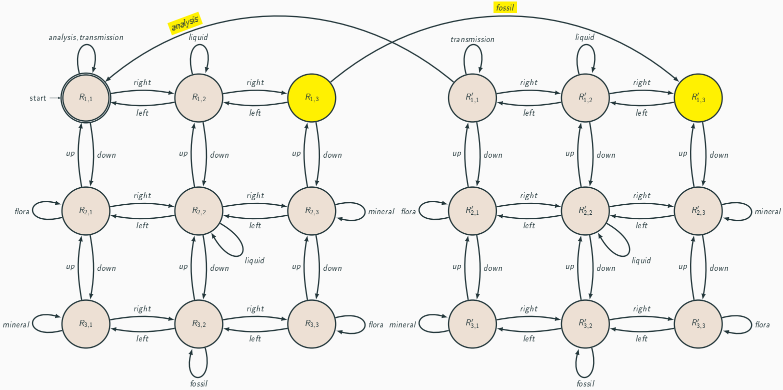
At most 1 sample from each subarea containing a fossil must be collected before analysis

$$\Downarrow$$

5a) **At most 1 sample from subarea** $(1,3)$ **containing a fossil must be collected before analysis**

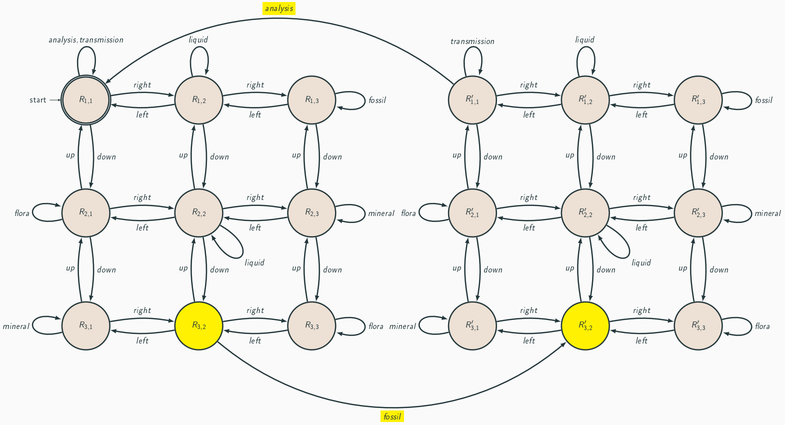
5b) **At most 1 sample from subarea** $(3,2)$ **containing a fossil must be collected before analysis**

Requirement 5a



At most 1 sample from subarea $(1, 3)$ containing a fossil must be collected before analysis

Requirement 5b



At most 1 sample from subarea $(3, 2)$ containing a fossil must be collected before analysis

Size of controller(s) (number of states)

Requirements	Generated States	Removed States	Total States
R_1	18	0	18
R_1, R_2	54	0	54
R_1, R_2, R_3	108	0	108
R_1, R_2, R_3, R_4	1458	846	612
R_1, R_2, R_3, R_4, R_5	1620	900	720