Test-Driven Simulation of Robots Controlled by Enzymatic Numerical P Systems Models

Radu Traian Bobe[1] Marian Gheorghe[2] Florentin Ipate[1] Ionuț Mihai Niculescu[1]

[1] Department of Computer Science, Faculty of Mathematics and Computer Science, University of Bucharest, Str Academiei 14, Bucharest, 010014, Romania

[2] Faculty of Engineering and Informatics, University of Bradford, Bradford, West Yorkshire, BD7 1DP, United Kingdom

Summary

Modelling Robot Controllers

Enzymatic Numerical P Systems Models

Testing Cyber-Physical Systems

Test-Driven Approach for Model Validation

Simulation Tools

Points of interest

- Software Testing. Test-driven development
- Cyber-Physical Systems
- Membrane Computing
- (Enzymatic) Numerical P Systems

The importance of software testing

- A process that aims to ensure the proper functionalities, according to the requirements
- The remarkable progress of late years confers software testing an increased attention
- The safety of software systems for large-scale use is ensured by testing

Software Testing: Previous work

- Exhaustive enumeration of a program’s input => infeasible for reasonable sized programs
- Random Methods => unreliable and unlikely to exercise “deeper” features of software that are not exercised by mere chance
- Previous efforts have been limited by the size and complexity of the software involved => metaheuristic search techniques

Search-based Software Testing (SBST)

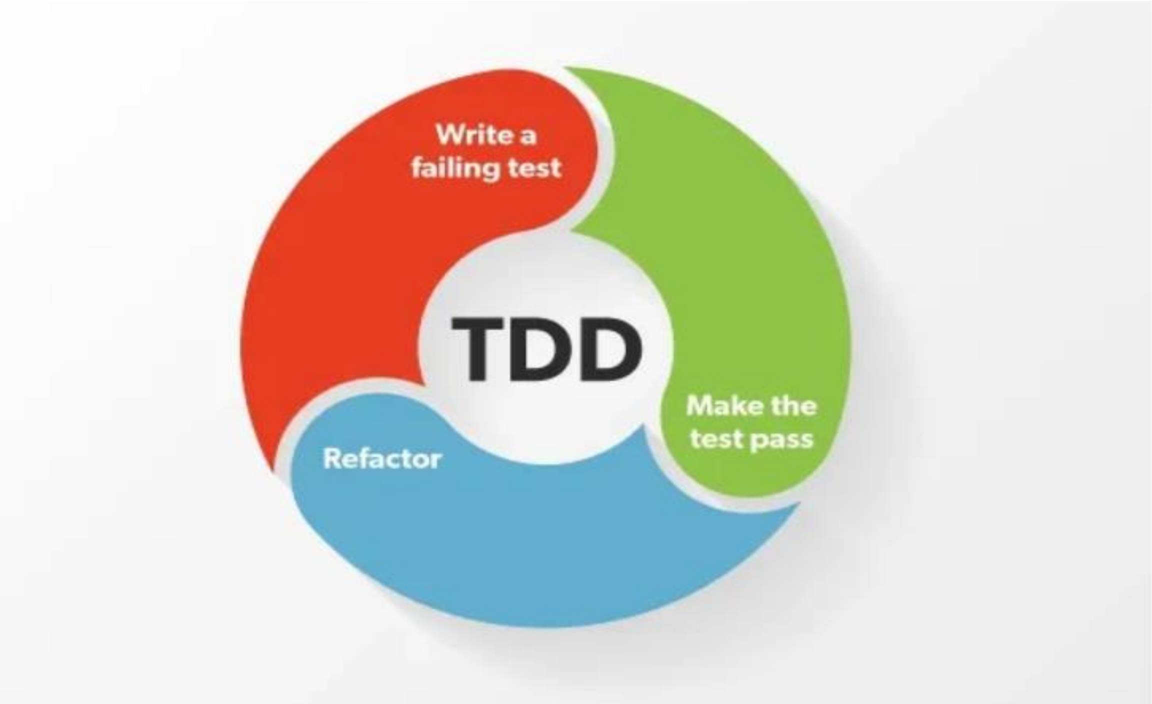
- Software Testing technique that involves using search algorithms to automatically generate test inputs or test cases

- The application of optimizing search techniques (for example, Genetic Algorithms) to solve problems in software testing

- “Used to generate test data, prioritize test cases, minimize test suites, optimize software test oracles, reduce human oracle cost, verify software models, test service-orientated architectures, construct test suites for interaction testing, and validate real time properties (among others)” (https://sbst22.github.io)

Test-driven development

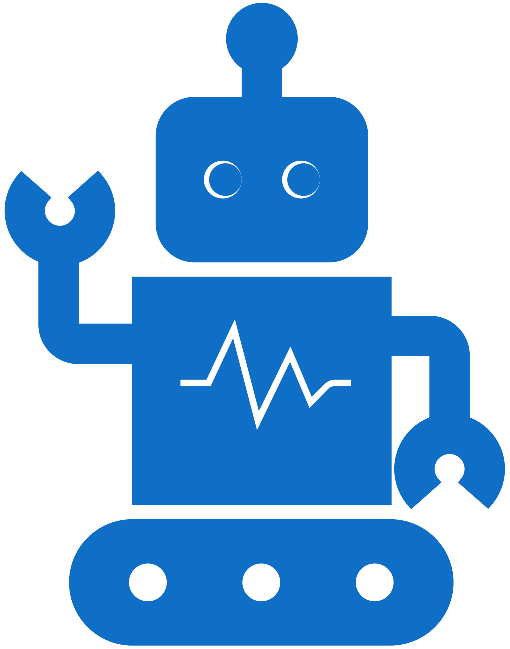
- Software Development practice that accents writing tests before writing the actual code



Source: levelup.gitconnected.com

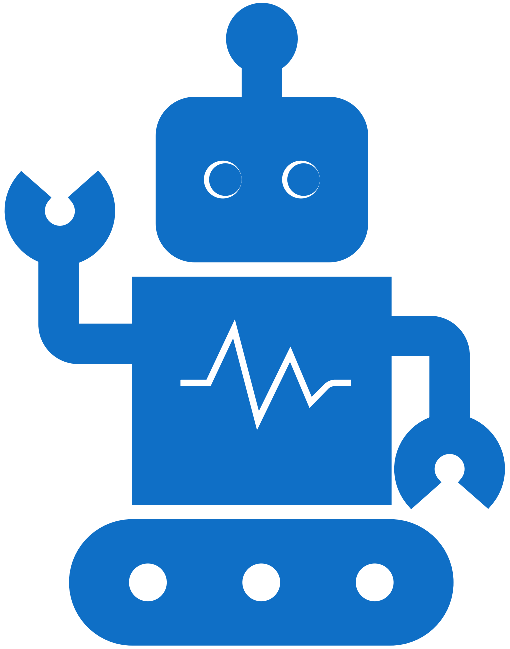
Cyber-Physical Systems

- Systems that integrate physical and computational components to monitor and control the physical processes seamlessly
- The safety of this systems is an essential aspect and is ensured by testing



Cyber-Physical Systems

- Robots
- Personalized Medical devices
- Energy-neutral buildings
- Self-driving cars



Modelling Robot Controllers

- We aimed to model variants of an obstacle avoidance controller
- The controllers are based on enzymatic numerical P systems models
- Experimental environment involves tools to model, simulate and test the models that design the controller: Pep, Webots, AmbieGen

Our approach

- We defined three testing scenarios that challenge four different lane keeping controllers designed to move an educational robot called E-puck
- We designed each model after analyzing the results produced by testing the previous ones
- Testing scenarios: corridors, a square, roads generated by AmbieGen (open source search-based software testing tool)

Membrane Computing

- Field of computing introduced by Gh. Păun in 1998
- Inspired by the structure and functionality of the living cells

Membrane Computing

- Field of computing introduced by Gh. Păun in 2002
- Inspired by the structure and functionality of the living cells
- Our experiment involves two types of membrane systems (P systems): numerical P systems and enzymatic numerical P systems

(Enzymatic) Numerical P Systems

- Computational models that only inherit the membrane structure from the membrane systems

- The membranes contain variables

- The values of the variables are processed by the programs every time unit

(Enzymatic) Numerical P Systems

The (enzymatic) numerical P system (EN P system) is defined by the tuple:

$$\Pi = (m, H, \mu, (Var_1, Pr_1, Var_1(0)), \ldots, (Var_m, Pr_m, Var_m(0)))$$

where:

- $m \geq 1$ is degree of the system Π (the number of membranes);
- H is an alphabet of labels;
- μ is membrane structure;
- Var_i is a set of variables from membrane $i, 1 \leq i \leq m$;
- $Var_i(0)$ is the initial values of the variables from region $i, 1 \leq i \leq m$;
- Pr_i is the set of programs from membrane $i, 1 \leq i \leq m$.

Basic Model

$$leftSpeed = cruiseSpeed + \sum_{i=1}^{n} weightLeft_i \cdot prox_i$$

$$rightSpeed = cruiseSpeed + \sum_{i=1}^{n} weightRight_i \cdot prox_i$$

leftSpeed, rightSpeed – the speed of the two wheels of the robot

cruiseSpeed – constant representing the movement speed

$prox_i$ –the proximity sensors located on E-puck

n- the number of sensors

weightLeft, weightRight- constants used to obtain the desired behavior (empirically chosen)

Basic Model with Rotation

$$weightLeft = \sum_{i=1}^{n} weightLeft_i \cdot prox_i$$

$$weightRight = \sum_{i=1}^{n} weightRight_i \cdot prox_i$$

$$leftSpeed = cruiseSpeed \cdot weightLeft + f(weightLeft) \cdot cruiseSpeed$$

$$rightSpeed = cruiseSpeed \cdot weightRight + f(weightRight) \cdot cruiseSpeed$$

, where *f* is defined as follows:

$$f(x) = \begin{cases} 1, & \text{if } x = 0 \\ 0, & \text{otherwise} \end{cases}$$

Refined Model with Rotation

- Developed to adjust the “zig-zag” motion of the robot
- Recentering the robot after avoiding an obstacle
- Simulating a Finite State Machine inside the enzymatic numerical P system :

a) state 0 - the robot is moving in a straight line

b) state 1 - the robot is moving in the presence of an obstacle

c) state 2 - the robot is moving to approximately the center of the lane

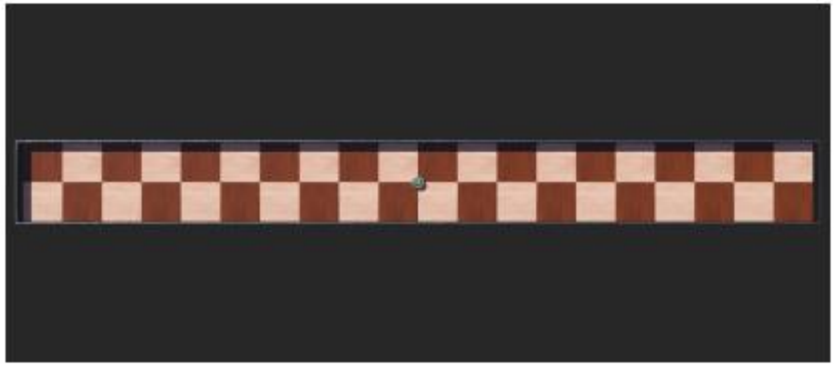
d) state 3 - the robot is recentering on the lane

Extended Refined Model

- Problematic behavior when the robot approached perpendicularly the obstacle (it remained locked)

$$
\begin{aligned}
directionLeft =& eq(|weightLeft|, |weightRight|) \cdot gt(|weightLeft|, 0) \cdot \\
& \cdot gt(|weightRight|, 0) \cdot weightLeft \cdot cruiseSpeed \cdot 0 \\
directionRight =& eq(|weightLeft|, |weightRight|) \cdot gt(|weightLeft|, 0) \cdot \\
& \cdot gt(|weightRight|, 0) \cdot weightRight \cdot 0 + cruiseSpeed
\end{aligned}
$$

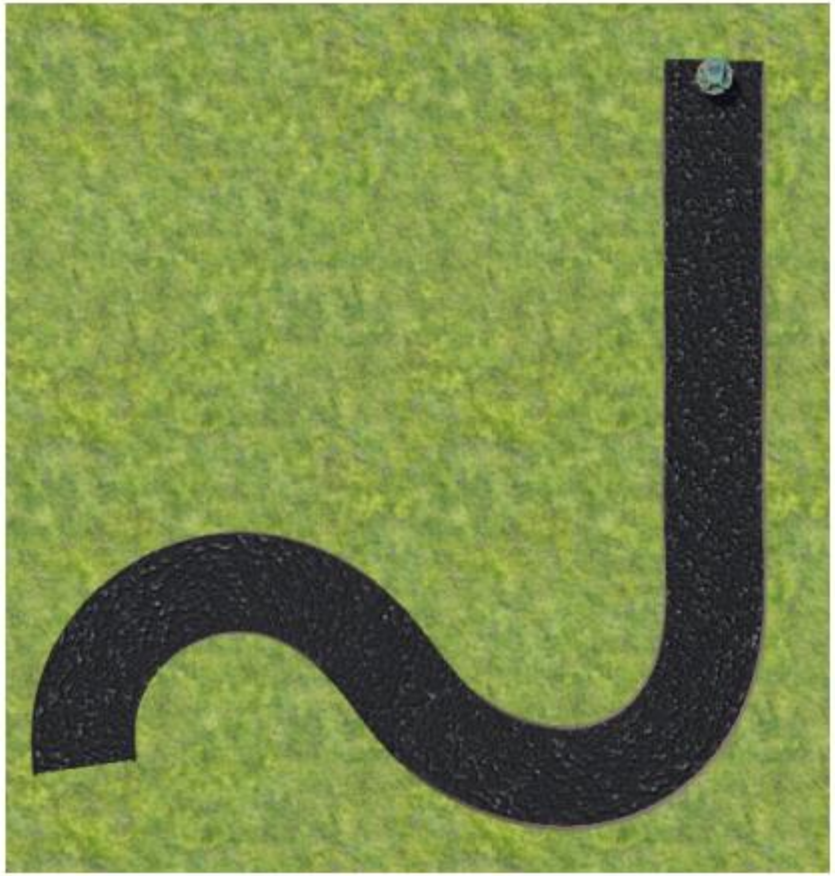
Representing Test Cases in Webots



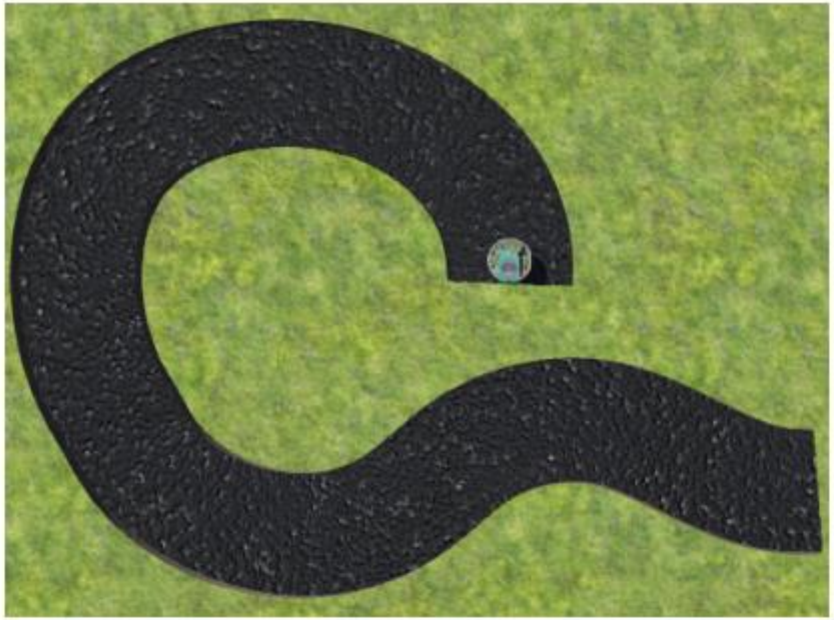
Corridor

Square

Representing Test Cases in Webots

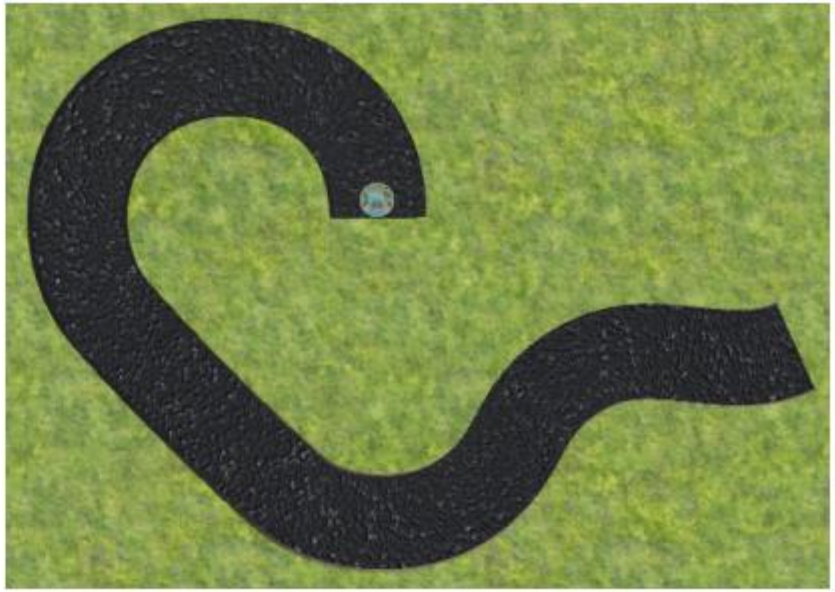


Road 1

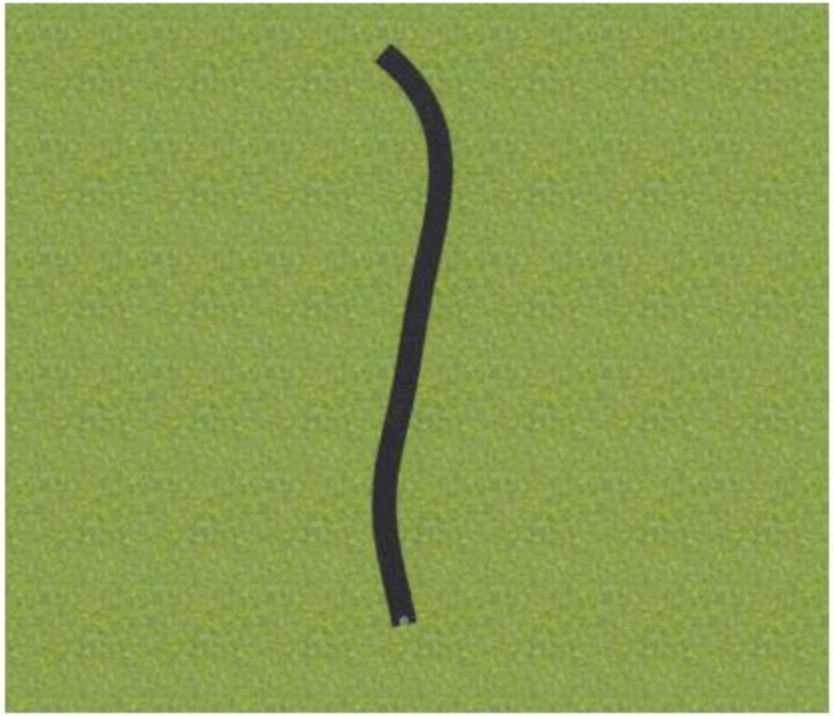


Road 2

Representing Test Cases in Webots



Road 3



Road 4

Experimental results

Test type	Π_{M_1}	Π_{M_2}	Π_{M_3}	Π_{M_4}
Corridor straight	Failed	Failed	Failed	Passed
Corridor angle	Failed	Passed	Failed	Passed
Square straight	Failed	Failed	Passed	Passed
Square angle	Failed	Passed	Passed	Passed
Road 1	Failed	Passed	Passed	Failed
Road 2	Failed	Passed	Passed	Passed
Road 3	Failed	Passed	Passed	Failed
Road 4	Passed	Passed	Passed	Failed

Conclusions and Future Work

- The main purpose of this experiment was to introduce advanced and modern testing approaches in the area of membrane computing
- Another challenge was to refine the models based on previous testing results
- The future work relies on two directions: involving other types of P systems in our experiments and dynamically assigning values to weights based on the controller behavior during the previous tests

Thank you