Implementing the Alpha Algorithm in Robot Swarm Using Timed Automata and UPPAAL

Szymon Gałecki

Contents

Outline

Problem Statement : purpose and motivation of this work.
Alpha Algorithm : what is it, and why is it interesting?
Timed Automata : definition and real-time system.
Modelling Behaviour : modelling aspects of behaviour.
UPPAAL Implementation : from a single robot to robot swarm.
Related Works : what others did.
Conclusions : what I did.

Problem Statement : 1/3

In swarm robotics, multiple robots work together to solve problems by interacting with each other and the environment in a similar way as bees, ants or birds. [11]

To enable the interaction within the swarm, first we have to achieve aggregation.

There are many algorithms that focus on aggregation task for robot swarm but we have chosen the Alpha algorithm as it was mentioned in reviewed papers [5], [14], [2], [6], [12] and because it achieves aggregation using a single variable.

Problem Statement : 2/3

While the algorithm is well defined and there are works which focus on property verification, the modeling part was not described.

This means that performed experiments are hard or impossible to recreate as the exact model was not given.

Therefore this work aims to breach this gap by focusing on modeling aspect.

Problem Statement : 3/3

We utilized the UPPAAL tool as it is suitable for modeling and verifying real-time systems.

Work includes an explanation of the implementation process in UPPAAL, simplifications made to the algorithm, and the connection between design choices and the anticipated physical behavior of the model.

We demonstrate how the composition of these models creates a system that executes the Alpha algorithm in a robot swarm.

Alpha Algorithm : 1/2

Alpha algorithm was introduced by Julien Nembrini [10]. It was inspired by Kasper Støy's work [13].

Støy proposed and implemented a simple control system for aggregating robots. Instead of relying on environment and localisation information, it uses physical properties of the signal used for communication.

Robot behaviour is solely determined by the change in the number of robots that are in the range of its signal.

Alpha Algorithm : 2/2

Alpha algorithm is an approach to an aggregation task within the category of spatial organisation.

It is based on the assumption that robots send and receive signals through omnidirectional channels like radio or infrared.

Single robots make decisions about their movement only based on the number of connections to other robots.

The inter-connectivity of the swarm is controlled by the alpha parameter which is a threshold on the desired number of connections for a single robot.

Timed Automata : 1/2

Definition 3.1 (Definition of timed automaton [1])**.** A timed automaton is a tuple (Σ, S, S_0, C, E) where:
Σ - input alphabet;
S - finite set of automaton states;
$S_0 \subseteq S$ - set of start states;
C - finite set of clocks;
$E \subseteq S \times S[\Sigma \cup \epsilon] \times 2^C \times \Phi(C)$ - set of transitions

An edge in timed automaton is a tuple $\langle s, s', \sigma, \lambda\delta \rangle$, where:
s - origin state;
s‘ - destination state;
σ - input symbol for the transition;
λ - set of clocks to be reset with this transition;
δ - condition enabling the transition;

Timed Automata : 2/2

While we are able to model Alpha algorithm using a simple automaton we would not be able to model a real-time system like robot swarm.

The timing aspect allows us to model a system with time dependent behaviours and to verify time dependent properties.

We can observe a system behaviour throughout time and evaluate the stability of the implemented algorithm.

Modelling Behaviour : 1/3

Movement

Robot always moves in one of four directions: up, right, down or left.

Initial direction is chosen at random and mapped to vertical and horizontal components.

Direction will not change unless the robot performs random turn or 180 degree turn.

Random turn chooses new direction in the same way that initial direction is determined.

Modelling Behaviour : 2/3

Connection

Number of robot connections is the main parameter determining the robot behaviour.

One of the assumptions of the Alpha algorithm is that physical signal used by robot is omnidirectional.

We assume that two robots are connected if their Euclidean distance is smaller than radius of physical signal that we mimic.

To model the physical signal we need information about robot coordinates and mutual distances.

Modelling Behaviour : 3/3

Initialisation and clocks

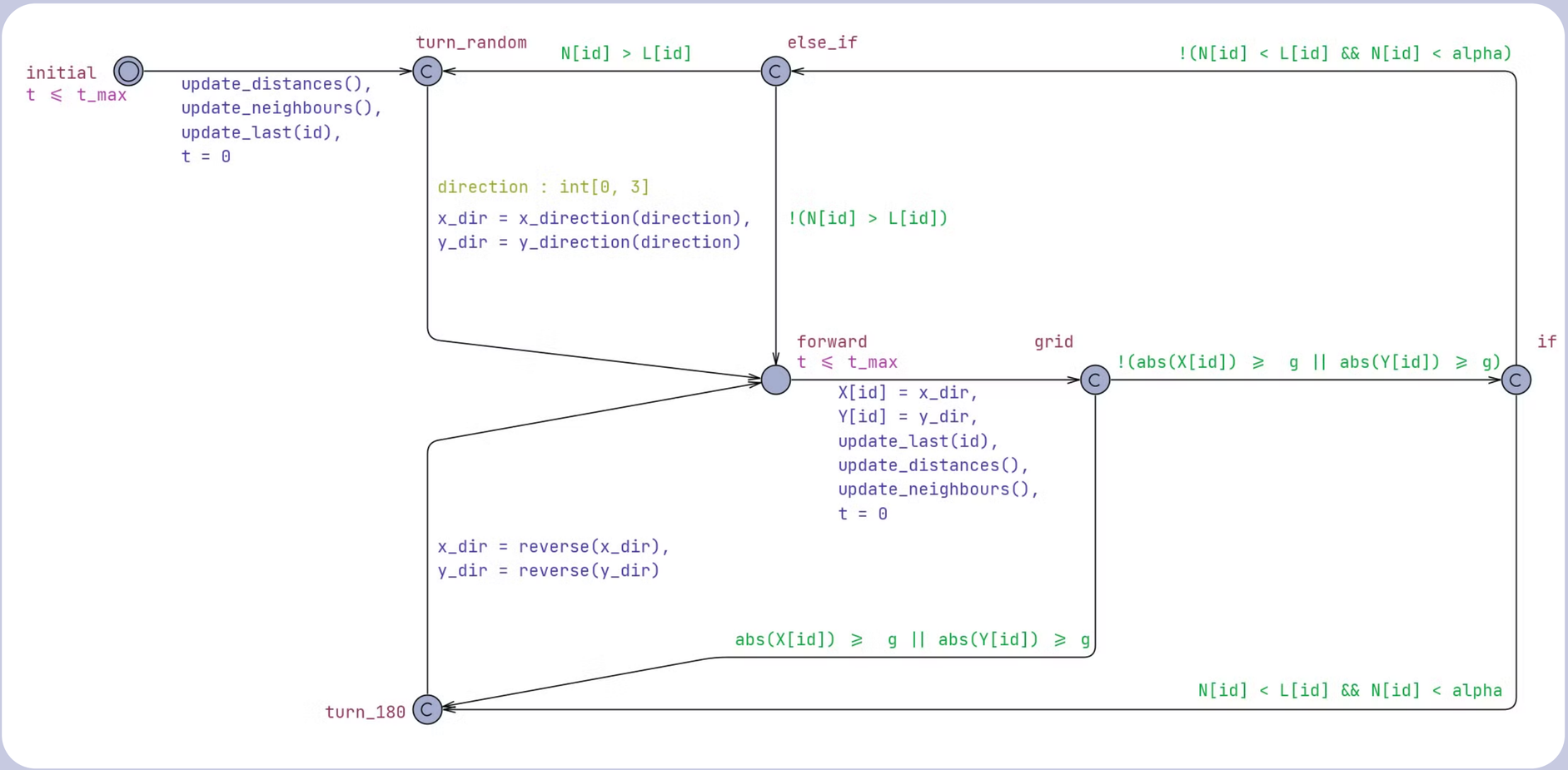
Every robot gets initialised with the same set of parameters apart from its ID.

In the beginning all robots are placed at exactly the same point (x=0, y=0) and have no set direction.

Each robot has its own clock that controls how long it can remain in states with defined invariant before being forced to transition.

UPPAAL Implementation : 1/3

Model



UPPAAL Implementation : 2/3

Verification in UPPAAL

Definition 3.2 (Logical quantifiers in UPPAAL [3])**.** The formulas should be one of the following forms

- `A[]` ϕ – Invariantly ϕ.
- `E<>` ϕ – Possibly ϕ.
- `A<>` ϕ – Always Eventually ϕ.
- `E[]` ϕ – Potentially Always ϕ.
- ϕ `-->` ψ – ϕ always leads to ψ.

where ϕ, ψ are local properties that can be checked locally on a state, i.e. boolean expressions over predicates on locations and integer variables, and clock constraints.

UPPAAL Implementation : 3/3

Robot swarm implementing Alpha algorithm

```
// Robot swarm composed from two robots
// (id, x, y, x_dir, y_dir)
P1 = Robot(0, 0, 0, 0, 0);
P2 = Robot(1, 0, 0, 0, 0);
system P1, P2;
```

```
// System parameters
const int r = 2;     // radius of the simulated signal
const int g = 5;     // grid boundary ⇒ grid : 2g x 2g
const int alpha = 1; // alpha parameter
const int t_max = 5; // time boundary on the robot clock
```

```
Verified properties:
1. A[] not deadlock
2. A[] N[0] == N[1]
3. A[] P1.forward or P1.initial imply P1.t ≤ t max
4. A[] not P1.initial and not P1.forward imply P1.t == 0
5. A[] P1.turn random imply P2.initial or P2.forward
6. A[] P1.forward imply (P1.x dir ≠ 0 and P1.y dir == 0) or (P1.x dir == 0 and P1.y dir ≠ 0)
```

Related Works : 1/2

Paper [5] verified the correctness of the Alpha algorithm by model checking the state-space reduced system with different modes of concurrency. System was reduced to 2-3 robots and grid of size 5x5-8x8 to avoid the state explosion problem.

Examined modes of concurrency were synchrony, strict turn taking, non-strict turn taking and fair asynchrony. Synchrony was found to be the most accurate mode of concurrency for modeling real execution.

Property "no specific robot will remain disconnected forever" was defined using propositional linear-time temporal logic and verified using symbolic model checker - NuSMV. The property was successfully verified only for a system consisting of two robots executing with synchronous mode of concurrency.

Related Works : 2/2

In [14] the Alpha algorithm was simplified in a similar way as in this work.

It used temporal logic to formally specify emergent behaviours of a robotic swarm system. Two such properties were defined but not verified.

Property 1 - ”It is repeatedly the case that for each robot, we can find another robot so that they are connected”.

Property 2 - ”Eventually it will always be the case that every robot is connected to at least k robots, where k is a predefined constant”

Conclusions : 1/2

In this paper we have presented the detailed implementation of the Alpha algorithm [10] using network of timed automata.

The Alpha algorithm has been chosen because of its presence in reviewed literature [5], [14], [2], [6], [12].

We have used an integrated tool for modeling and verification of real-time systems, UPPAAL [9].

We have transformed the pseudocode defining the algorithm into a timed automaton.

Conclusions : 2/2

We have explained the implementation of timed automaton in UPPAAL and simplifications of the algorithm

We have described connections between the design choices and expected physical behaviour of the model.

We have shown how a composition of models became a system implementing the Alpha algorithm in robot swarm.

We have defined and successfully verified properties of the system that increase our confidence in the correctness of implementation.