From Agents to Swarms, from Model-Checking to Proof

Michael Fisher

Department of Computer Science and
Centre for Autonomous Systems Technology
University of Liverpool, UK

Northern Concurrency Workshop, January 2017

Overview

- *Agents and Swarms*
  Autonomy $\longrightarrow$ Agent $\longrightarrow$ Rational Agent
  Agents $\longrightarrow$ Swarm

- *Specifying Agents*
  Agent $\longrightarrow$ Temporal Logic
  Rational Agent $\longrightarrow$ Temporal & Multi-Modal Logic

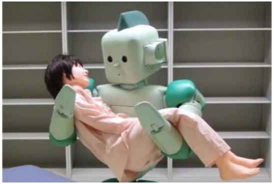
- *Verification — Model-Checking*
  Rational Agent $\longrightarrow$ Program Model-Checking
  Swarm $\longrightarrow$ Model-checking multiple probabilistic processes

- *Verification — Proof*
  First-Order Temporal Logic
  Finite Failures

- *Current Work*

What is Autonomy?

the ability of a system to make its own decisions and to act on its own, and to do both without direct human intervention.



`rtc.nagoya.riken.jp/RI-MAN`

`www.volvo.com`

Agents

An *agent* captures the core concept of autonomy, in that it is *able to make its own decisions without human intervention*.

But: in many cases this isn't enough, as we need to know *why*!

We need the concept of a "*rational agent*":

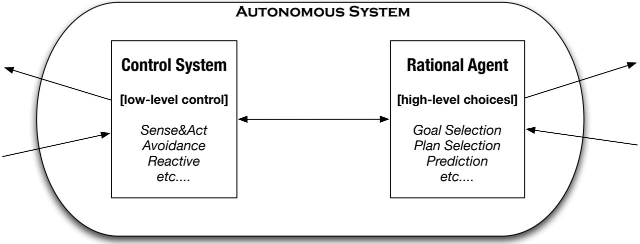
> a rational agent must have explicit *reasons* for making the choices it does, and should be able to explain these if needed

Aside: Autonomous Systems Architectures

Requirement for *reasoned* decisions and explanations has led on to *hybrid agent architectures* combining:

1. *rational agent* for high-level autonomous decisions, and
2. traditional *control systems* for lower-level activities,

These have been shown to be easier to *understand*, *program*, *maintain* and, often, much more *flexible*.



Example: from Pilot to Rational Agent

Autopilot can essentially fly an aircraft

- keeping on a particular path,
- keeping flight level/steady under environmental conditions,
- planning routes around obstacles, etc.

Human pilot makes high-level decisions, such as

- where to go to,
- when to change route,
- what to do in an emergency, etc.

Rational Agent now makes the decisions the pilot used to make.

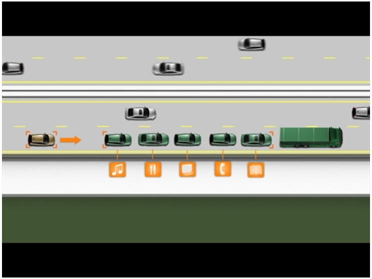
Example: Aerospace

In remote or dangerous environments it is impractical or costly to have direct human control, so *autonomy* is increasingly being built into the controlling software.

For example, *autonomous choices* are an important element in many *aerospace* applications, such as cooperative

formation flying satellites or *unmanned air vehicles*

Example: Automotive



Road train (‘convoy’) system has control of speed, direction, etc.

Driver can, in principle, take control back.

Example: Robotic Assistants

Robotic Assistants are now being designed to help the elderly or incapacitated.

⟵ Here is `Care-O-bot`

And here ↓ is `Care-O-bot` deployed in a house

Example: Robot Swarms

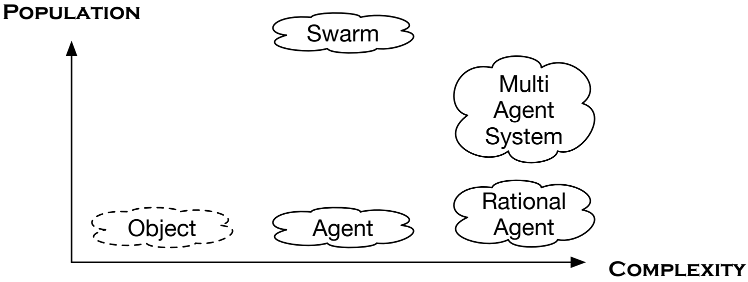
Utilize *many* small, identical robots, each with quite limited communication.

Then: get them to undertake one or more simple tasks.

Such *large* collections/swarms

- can be resilient and fault-tolerant,
- can cooperate at a basic level, and
- can exhibit non-trivial emergent behaviours.

A Simple Picture



A Little Temporal Logic

Temporal Logic (TL) extends classical logic with, eg:

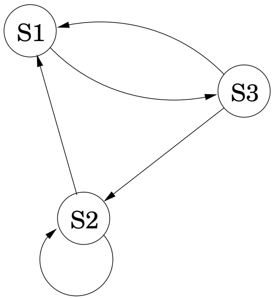
- $\bigcirc$ (*in the next moment*)
- $\lozenge$ (*at some future moment*)
- $\square$ (*in all future moments*)

TL widely used in specification and verification.

Simple Agents and Temporal Logic

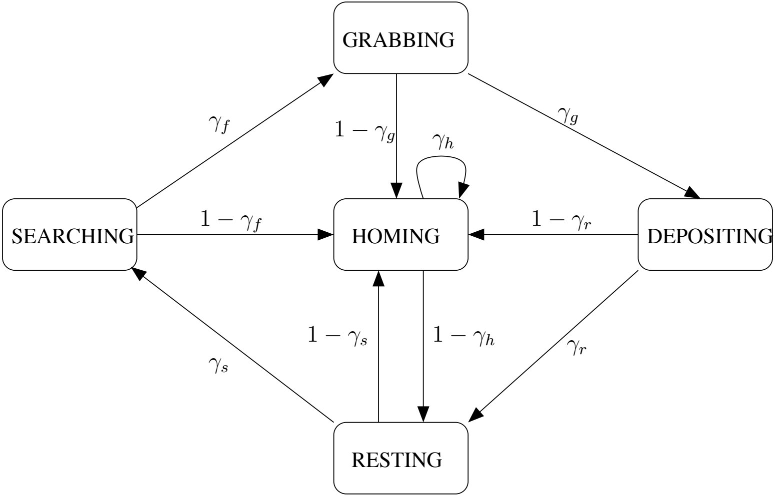
Simple finite state automata can often be concisely described using propositional temporal logic.

So, we can use a propositional temporal formula to represent a single, simple agent, e.g:



$$\longrightarrow \quad \begin{array}{rcl} In_{S1} & \Rightarrow & \bigcirc In_{S3} \\ In_{S2} & \Rightarrow & \bigcirc(In_{S1} \vee In_{S2}) \\ In_{S3} & \Rightarrow & \bigcirc(In_{S1} \vee In_{S2}) \end{array}$$

Automaton for Simple (Swarm) Agent



Describing Rational Agents

Logical theories for rational agents typically consist of

- Dynamism: temporal or dynamic logic;
- Information: modal/probabilistic logics of belief/knowledge;
- Motivation: modal logics of goals, intentions, desires.

This requires *combinations* of logics.

For example, the well known BDI approach comprises

- a (branching) temporal/dynamic logic,
- a KD45 modal logic of belief,
- a KD modal logic of desire, and
- a KD modal logic of intention.

Complexity — Gets Worse

- Uncertainty of real world . . . . *probabilistic*
- Location and tracking . . . . *spatial, dynamic*
- Importance of messages . . . . *priority*
- Roles and responsibilities . . . . *modal attitudes*
- Privacy and security . . . . *knowledge/belief*
- Timed systems . . . . *real-time*
- Multi-agent . . . . *cooperation/coalition*
- Autonomy . . . . *intentions/motivations*

$\Rightarrow$ In realistic scenarios, we will need to *combine* several logics.

Sample Logical Requirement: Robot Assistant

"If a patient is in danger, then the patient believes that there is a probability of 95% that, within 2 minutes, a helper robot will want to assist the patient."

$B^{\geq 0.95}_{patient}$ .... *patient believes with 95% probability*

$\Diamond^{\leq 2}$ .... *within 2 minutes*

G_{helper} .... *helper robot has a goal*

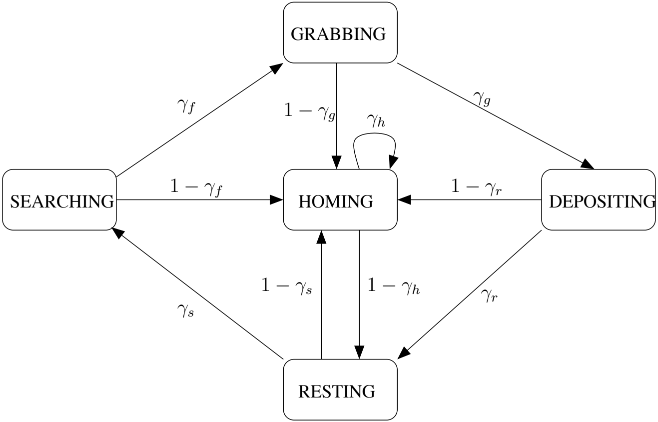
$$in\_danger(patient) \Rightarrow B^{\geq 0.95}_{patient} \Diamond^{\leq 2} G_{helper}\, assist(patient)$$

Requirements: Robot Swarms

Verify behaviour of *large* collections/swarms.

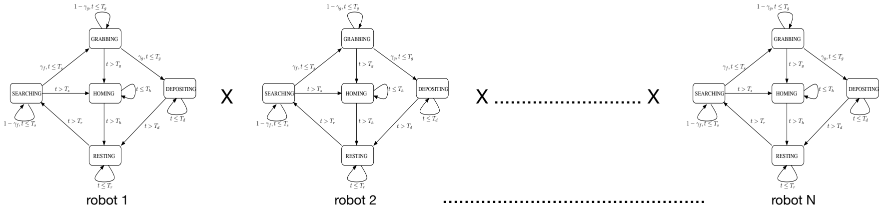
- ⇒ Will the swarm *cooperate*?
- ⇒ Will the swarm collectively discover the target?
- ⇒ If one robot *fails*, will the swarm still function?
- ⇒ What if several robots *fail*? What is the probability that the whole swarm will still be effective?

Model-Checking a Simple Agent



Verification is typically probabilisitic model-checking, e.g. PRISM.

Model-Checking Swarms



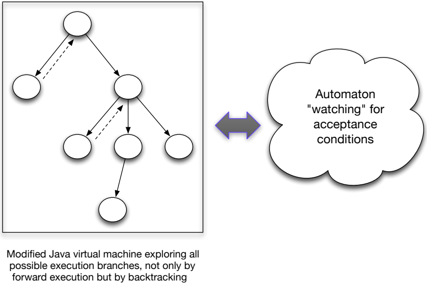
Verification again by probabilisitic model-checking.

Complexity.....

Verifying (Rational) Agents

We have many options of how to carry out formal verification.

We often use Program Model-Checking whereby logical specification is checked against the *actual* agent code.



Combines (backtracking) symbolic execution and a monitoring automaton.

Swarms, and back to Temporal Logic

Temporal Logic (TL) extends classical logic with, eg:

- $\bigcirc$ (*in the next moment*)
- $\lozenge$ (*at some future moment*)
- $\square$ (*in all future moments*)

Propositional TL used in specification and verification.

First-order TL in general, is *not* recursively enumerable.

But, monodic FOTL has 'good' properties: axiomatisability and, sometimes, decidability.

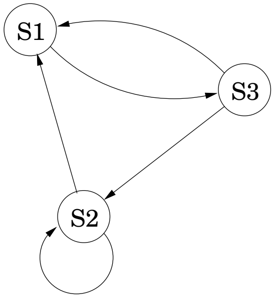
$$\forall X, Y.\ p(X,Y) \ \Rightarrow\ \bigcirc \exists Z.\ q(Z,Y)$$ monodic

$$\forall X, Y.\ p(X,Y) \ \Rightarrow\ \bigcirc q(X,Y)$$ not monodic

Finite-State Processes and Temporal Logic

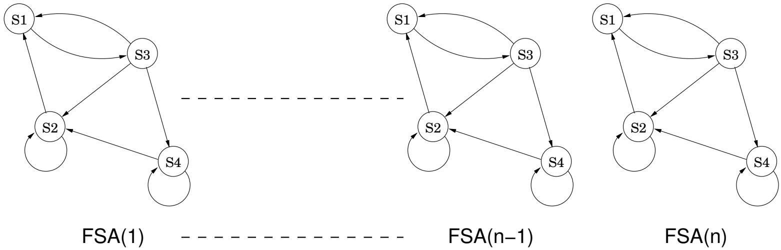
Recall that simple finite state automata can often be concisely described using propositional temporal logic.

So, we can use a propositional temporal formula to represent a single agent, e.g:



$$\longrightarrow \quad \begin{array}{lcl} In_{S1} & \Rightarrow & \bigcirc In_{S3} \\ In_{S2} & \Rightarrow & \bigcirc(In_{S1} \lor In_{S2}) \\ In_{S3} & \Rightarrow & \bigcirc(In_{S1} \lor In_{S2}) \end{array}$$

Swarms and First-Order TL



$$\downarrow$$

$$
\begin{array}{rcl}
\forall X.\ In_{S1}(X) & \Rightarrow & \bigcirc In_{S3}(X) \\
\forall X.\ In_{S2}(X) & \Rightarrow & \bigcirc (In_{S1}(X) \vee In_{S2}(X)) \\
\forall X.\ In_{S3}(X) & \Rightarrow & \bigcirc (In_{S1}(X) \vee In_{S2}(X) \vee \ldots\ldots
\end{array}
$$

Automated Verification

We can specify correctness conditions in FOTL, eg:

$\exists X.\ \Diamond In_{S4}(X)$ $\qquad\qquad$ $\forall Y.\ \Diamond\Box In_{S2}(Y)$

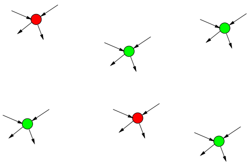
∞-state system $\searrow$

monodic FOTL $\xrightarrow[\text{resolution}]{\text{clausal}}$ correctness/ counter-example

correctness condition $\nearrow$

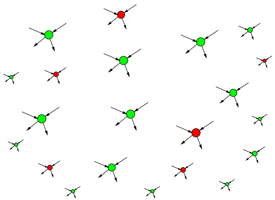
Failing Robots

What if some robots *fail*?



With a finite number of robots,
and a fixed number of failures,
then verification is still feasible.

But what if we don't have a *fixed* number of failures?



How about a *finite*, but *unknown*, number of failures?

Finite Failures

Provide FOTL axiomatisation of the finiteness we desire, then add this to the FOTL specification of the system.

For example, consider:

$$\forall X.\ \textit{failed}(X) \ \Rightarrow\ \Box \textit{failed}(X)$$

Now if Y ranges over finite set of *failing* processes, then add

$$\forall Y.\ \Diamond \textit{failed}(Y) \ \Rightarrow\ \Diamond(\forall Y.\ \textit{failed}(Y))$$

Intuitively:

> *since we have a finite number of failures, there will be a future moment when all will have occurred.*

Current Work: Representation Issues

- In Social Robotics, *trust* is key
- Disobeying orders will erode trust!
- Privacy violations will erode trust!!
- What if person orders robot to do something robot thinks is *detrimental* to person?

  Or orders robot not to tell anyone?

Trustworthiness	*versus*	human dignity/privacy
	versus	direct human orders
	versus	robot confidence in diagnosis
	versus	legality
	etc	

Current Work: Complexity Issues

We have been developing a *temporal logic* with improved complexity properties.

- Essentially this combines classical XOR restrictions with standard temporal logic.

  The outcome is a temporal logic with (if we are careful!) a polynomial decision procedure.
- Extended to *first-order temporal logic*, giving FOTLX.

How far can we push this in large-scale swarm verification?

Spatial aspects still a problem!

Current Work: Applications

- Verifying Autonomous Systems.
- Certification Evidence for Autonomous Unmanned Aircraft.
- Trustworthy Human-Robot Interactions.
- Formal Verificaiton of Ethical Choices in Autonomous Systems.
- BSI Standard **BS 8611**: *Guide to the Ethical Design and Application of Robots and Robotic Systems*.
- Analysis of Social Networks and Digital Crowds.

Sample Relevant Publications

- Dennis, Fisher, Lincoln, Lisitsa, Veres. Practical Verification of Decision-Making in Agent-Based Autonomous Systems. *Journal of Automated Software Engineering 23(3):305-359*, 2016.
- Dennis, Fisher, Slavkovik, Webster. Formal Verification of Ethical Choices in Autonomous Systems. *Robotics & Autonomous Systems 77:1-14*, 2016.
- Dennis, Fisher, Webster. Verifying Autonomous Systems. *Communications of the ACM 56(9):84–93*, 2013
- Dixon, Konev, Fisher, Nietiadi. Deductive Temporal Reasoning with Constraints. *Journal of Applied Logic 11(1):30-51*, 2013.
- Konur, Dixon, Fisher. Analysing Robot Swarm Behaviour via Probabilistic Model Checking. *Robotics & Autonomous Systems 60(2):199-213*, 2012.
- Slavkovik, Dennis, Fisher. An Abstract Formal Basis for Digital Crowds. *Distributed and Parallel Databases 33(1):3-31*, 2015.
- Webster, Cameron, Fisher, Jump. Generating Certification Evidence for Autonomous Unmanned Aircraft Using Model Checking and Simulation. *Journal of Aerospace Information Systems 11(5):258–279*, 2014.
- Webster, Dixon, Fisher, Salem, Saunders, Koay, Dautenhahn, Saez-Pons. Toward Reliable Autonomous Robotic Assistants Through Formal Verification: A Case Study. *IEEE Trans. Human-Machine Systems 46(2):186-196*, 2016.

Verification & Validation of Autonomous Systems Network

EPSRC funded Academic Network ...... `http://www.vavas.org`

Start of funding: 1st Sept 2015, for 3 years.

> *Aims:* to stimulate, coordinate, promote, and disseminate academic research on the verification and validation of autonomous systems

Activities so far:

- **Sep 2015:** *Agent Verification Workshop*, Liverpool
- **Dec 2015:** *Winter School on Verification of Mobile and Autonomous Robots*, York
- **Feb 2016:** *Workshop on Autonomous Systems: Legal/Regulatory Aspects and V&V*, London
- **Jul 2016:** *Workshop on Industrial Perspectives on the V&V of Autonomous Systems*, Sheffield
- **Nov 2016:** *Workshop on V&V for Autonomous Road Vehicles*, London