Formal methods for mobile anonymous robots

joint work with L. Millet, M. Potop-Butucaru, A. Sangnier, L. Tible and S. Tixeuil

LIP6, UPMC
Coalition of robots with weak capacities

- Anonymous
- No memory
- No communication
- No common handedness
- Only one sense: sight

Restriction to the ring
Execution models

- Each robot follows a cycle
  - Look/Compute
  - Move
- Either atomic and synchronous (FSYNC or SSYNC)
- Or asynchronous (ASYNC)
Execution models

- Each robot follows a cycle
  - Look/Compute
  - Move
- Either atomic and synchronous (FSYNC or SSYNC)
- Or asynchronous (ASYNC)
Execution models

• Each robot follows a cycle
  Look/Compute  Move
• Either atomic and synchronous (FSYNC or SSYNC)
• Or asynchronous (ASYNC)
• Each robot follows a cycle
  
  Look/Compute
  Move

• Either atomic and synchronous (FSYNC or SSYNC)

• Or asynchronous (ASYNC)
Execution models

- Each robot follows a cycle **Look/Compute** Move

- Either atomic and synchronous (FSYNC or SSYNC)

- Or asynchronous (ASYNC)
Execution models

• Each robot follows a cycle
  
  Look/Compute  
  Move

• Either atomic and synchronous (FSYNC or SSYNC)

• Or asynchronous (ASYNC)
Execution models

• Each robot follows a cycle
  Look/Compute
  Move

• Either atomic and synchronous (FSYNC or SSYNC)

• Or asynchronous (ASYNC)
Execution models

• Each robot follows a cycle
  Look/Compute ➔ Move

• Either atomic and synchronous (FSYNC or SSYNC)

• Or asynchronous (ASYNC)
Execution models

• Each robot follows a cycle
  Look/Compute
  Move

• Either atomic and synchronous (FSYNC or SSYNC)

• Or asynchronous (ASYNC)
Execution models

- Each robot follows a cycle
  - Look/Compute
  - Move
- Either atomic and synchronous (FSYNC or SSYNC)
- Or asynchronous (ASYNC)
Objectives of the robots

- Perpetual exploration
- Exploration with stop
- Gathering
State of the art

• Algorithms parameterized by the number of nodes (and sometimes number of robots)

• Autostabilizing algorithms

• Difficult to prove! (hence sometimes incorrect…)

• Open cases
  
  Use of formal methods to
  • verify existing protocols
  • automatically synthesize protocols (or prove impossibility)
Outline

1. Parameterized verification of algorithms for robots
   1. Model
   2. Results

2. Synthesis of algorithms for robots
   1. Model
   2. Results
1. Parameterized verification of algorithms for robots

- Model-Checking of such algorithms for fixed number of robots and nodes [BLMPT16]

- Parameterized verification undecidable in general but here?
  
  Verification for a fixed number of robots, but parameterized number of nodes
(Quantifier-free) Presburger Arithmetics

\[ \tau := x \mid \tau + \tau \mid a \cdot \tau \mid \tau \mod a \]

\[ \Phi := \tau \bowtie b \mid \Phi \lor \Phi \mid \Phi \land \Phi \mid \neg \Phi \]

\[ a, b \in \mathbb{N} \]

\[ \bowtie \in \{<,>,=,\le,\ge\} \]

Satisfiability of a Presburger formula is decidable.

It is NP-complete for the quantifier-free fragment.

[P30]
View of a robot

- The view of a robot is the sequence of distances between all the robots on the ring.
- All the robots in a tower share the same view of the ring.
- Robots can be disoriented (symmetry).
View of a robot

• The view of a robot is the sequence of distances between all the robots on the ring

• All the robots in a tower share the same view of the ring

• Robots can be disoriented (symmetry)

\[ \langle 2,0,0,2,0,1 \rangle \]
\[ \langle 1,0,2,0,0,2 \rangle \]
View of a robot

• The view of a robot is the sequence of distances between all the robots on the ring

• All the robots in a tower share the same view of the ring

• Robots can be disoriented (symmetry)
• The view of a robot is the sequence of distances between all the robots on the ring

• All the robots in a tower share the same view of the ring

• Robots can be disoriented (symmetry)
View of a robot

• The view of a robot is the sequence of distances between all the robots on the ring

• All the robots in a tower share the same view of the ring

• Robots can be disoriented (symmetry)
View of a robot

• The view of a robot is the sequence of distances between all the robots on the ring

• All the robots in a tower share the same view of the ring

• Robots can be disoriented (symmetry)
Encoding of protocols

• Protocols will be QFP formulae interpreted over views.

• For a protocol $\phi$ and a view $V$ of robot $i$, if $V \models \phi$ then $i$ will move in the same direction than its view.

• A protocol is a QFP formula $\phi$ such that

$$\text{if } V \models \phi \text{ and } V \neq \vec{V} \text{ then } \vec{V} \not\models \phi$$
Encoding of protocols - example with 3 robots

\[(d_1 > 4 \land d_2 = 3 \land d_3 = 1)\]
\[\lor (d_1 = 2 \land d_2 > 4 \land d_3 = 3)\]
\[\lor (d_1 = 4 \land d_2 = 2 \land d_3 > 4)\]
\[\lor (d_1 > 4 \land d_3 > d_1 \land d_2 = 1)\]
\[\lor (d_1 = d_3 \land d_1 > 0 \land d_2 > 0)\]
\[\lor (d_2 > d_3 \land d_3 > d_1 \land d_1 > 1)\]
\[\lor (d_2 = d_1 \land d_3 = 1)\]
\[\lor (d_2 = 1 \land d_3 = 2)\]
Verification problems

SAFE$_m$ problem (m ∈ \{s, ss, as\}):

• Given: $\Phi$ a protocol, Bad undesirable configurations two QFP formulae

• Question: $\exists \exists$ a size of the ring and a configuration of the robots $p$ such that $p \notin \lbrack \text{Bad} \rbrack$ and $\text{Post}_m^*(\Phi, p) \cap \lbrack \text{Bad} \rbrack \neq \emptyset$
Verification problems

REACH_m problem (m ∈ {s, ss, as}):

• Given: Φ a protocol, Goal desirable configurations two QFP formulae

• Question: ∃? a size of the ring and a configuration of the robots p such that Post_m*(Φ, p) ∩ [Goal] = ∅
Outline

1. Parameterized verification of algorithms for robots
   1. Model
   2. Results

2. Synthesis of algorithms for robots
   1. Model
   2. Results
Bad news

Theorem
REACH_m undecidable for m∈{s,ss,a}.
SAFE_{as} undecidable.
Bad news

Theorem
REACH$_m$ undecidable for $m \in \{s, ss, a\}$. SAFE$_{as}$ undecidable.

Robots can simulate a 2 counter-machine
Simulation of a 2CM

- Relative positions of the robots on the ring encode values of counters and current instruction
- Need to avoid symmetry
Simulation of a 2CM

- Robots have no memory!
- To simulate a transition:
  
  Move corresponding robot left or right
  
  Move corresponding robot to the next instruction
Simulation of a 2CM

• Robots have no memory!

• To simulate a transition:
  
  Copy the current value of the counter to modify
  Move corresponding robot left or right
  Copy current instruction
  Move corresponding robot to the next instruction
Simulation of a 2CM

- stable configuration:

- indicates when the position of the robots encode an actual configuration of the 2CM

- if the 2CM stops, then on a big enough ring, robots can simulate the run

- Conversely? How to ensure that robots start in the initial configuration?
Simulation of a 2CM

- Two special robots $R_{C0}$ and $R_{Ch}$
Simulation of a 2CM

- Two special robots $R_{C0}$ and $R_{Ch}$
Simulation of a 2CM

- Two special robots $R_{C0}$ and $R_{Ch}$
Simulation of a 2CM

- If robots are not machine-like, nobody moves
- If robots are machine-like, they move, one at a time, and simulate the transitions
  - From a machine-like, stable configuration, encoding $C_0$, $R_{C0}$ moves.
  - From a machine-like, stable configuration encoding $C_h$, $R_{Ch}$ moves.
Reduction of the halting problem

- **Bad** encodes existence of a collision

- If the 2CM halts, then there is an execution where $R_{C0}$ sees $C_0$, and later $R_{Ch}$ sees $C_h$ and they move at the end $\rightarrow$ collision!

- If there is an execution with a collision, comes from $R_{C0}$ and $R_{Ch}$ $\rightarrow$ robots have simulated a halting run of the 2CM.
Source of undecidability

• For SAFE, problem comes from **asynchrony**!

• For REACH, similar idea but reduction works also in synchronous semantics
Positive results

Theorem
SAFE_{s} decidable.

• Reachability of a bad state equivalent to reachability in one step in an autostabilizing protocol

• Post expressible in Presburger arithmetics
Positive results

A protocol \( \Phi \) is \textit{pseudo-synchronous} if for all configuration \( p \), at most one robot \( i \) is such that \( \text{move}_i(\Phi, p) \neq \{0\} \).

Ex [KMP06]

Theorem
\( \text{SAFE}_{\text{as}} \) decidable for pseudo-synchronous protocols.

If the protocol is pseudo-synchronous, \( \text{SAFE}_{\text{as}} \) and \( \text{SAFE}_{\text{s}} \) are equivalent.
Positive results

A protocol $\Phi$ is pending-bounded if, for each configuration $p$ where $A(p) = \{i | \text{move}_i(\Phi, p) \neq \{0\}\}$, then for all configuration $p'$ such that $p \xrightarrow{J \subseteq A(p)} p'$, $A(p') \subseteq A(p) \setminus J$

Ex: [KKN10, DSN14]

**Theorem**

$\text{SAFE}_{\text{as}}$ decidable for pending-bounded protocols.

In pending-bounded protocols, we can bound the length of a bad run. $\text{Post}^n(\Phi, p)$ is Presburger expressible.
Positive results

Some properties of protocols are decidable (e.g. pseudo-synchronous or pending-bounded).

Use of SMT-solver to verify safety properties as well as properties of the protocol in the proofs of correctness.
Outline

1. Parameterized verification of algorithms for robots
   1. Model
   2. Results

2. Synthesis of algorithms for robots
   1. Model
   2. Results
2-Player Game
2-Player Game
2-Player Game
2-Player Game
2-Player Game
2-Player Game
2-Player Game
2-Player Game

Objectives: Reachability, Büchi, Parity, ...

**Problem**: Compute a strategy for Player 1, that is winning against any play of Player 2.
2-Player Game

Objectives: Reachability, Büchi, Parity, …

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

Strategy: $f : Pos^* \rightarrow Act$
2-Player Game

Objectives: Reachability, Büchi, Parity, …

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

Strategy:
\[ f : \text{Pos}^* \rightarrow \text{Act} \]
2-Player Game

Objectives: Reachability, Büchi, Parity, …

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

Strategy: 
\[ f : Pos^* \rightarrow Act \]
2-Player Game

Objectives: Reachability, Büchi, Parity, …

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

Strategy: \( f : Pos^* \rightarrow Act \)
2-Player Game

Objectives: Reachability, Büchi, Parity, …

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

\[ f : Pos^* \rightarrow Act \]
2-Player Game

Objectives: Reachability, Büchi, Parity, ...

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

Strategy:

\[ f : Pos^* \rightarrow Act \]
2-Player Game

Objectives: Reachability, Büchi, Parity, ...

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

Strategy:
\[ f : Pos^* \rightarrow Act \]
2-Player Game

Objectives: Reachability, Büchi, Parity, …

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

Strategy: \( f : \text{Pos}^* \rightarrow \text{Act} \)
2-Player Game

Objectives: Reachability, Büchi, Parity, ...

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

Strategy: $f: Pos^* \rightarrow Act$

Memoryless strategy: $f: Pos \rightarrow Act$
2-Player Game

Objectives: Reachability, Büchi, Parity, …

Problem: Compute a strategy for Player 1, that is winning against any play of Player 2.

Memoryless strategy:

- In 2-player games, if there is a winning strategy, there is a memoryless winning strategy.
- Algorithms for solving reachability games run in linear time in the size of the arena.
1. Model for synthesis of protocols

Configurations are equivalent by rotation and symmetry (a protocol should provide the same set of actions).
1. Model for synthesis of protocols

Two robots having the same view should take the same decision.
1. Model for synthesis of protocols

A protocol is a function $f: \text{View} \rightarrow \{\text{Front, Back, Doubt, Idle}\}$ such that if $f(V) = \text{Doubt}$ then $|V| = 1$ and if $|V| = 1$ then $f(V) \in \{\text{Doubt, Idle}\}$.

$f(\{<4,1,4>\}) = \text{Doubt}$

$f(\{<4,4,1>,<1,4,4>\}) = \text{Back}$
The case of gathering - FSYNC and SSYNC

Player 1

\[ f_1 : \{(3, 7, 0, 5, 1), (1, 5, 0, 7, 3)\} \mapsto \sim \]
\[ \{(0, 5, 1, 3, 7), (7, 3, 1, 5, 0)\} \mapsto \sim \]
\[ \cdots \]

\[ f_2 \]

\[ f_3 \]
The case of gathering - FSYNC and SSYNC

Player 2

\[ f : \{(6, 3, 0, 3), (3, 0, 3, 6)\} \mapsto \bigcap \{(0, 3, 6, 3)\} \mapsto ? \]
The case of gathering - FSYNC and SSYNC
Reachability objective
The case of gathering - FSYNC and SSYNC

Reachability objective

• Efficient encoding un UPPAAL TiGa

• Can synthesize synchronous strategy for any size of ring and any number of robots.
The general case?

- What about other objectives, like perpetual exploration?
- What about asynchronous executions?
Asynchronous executions

If **pending moves** then next configurations will be different.

But if **pending moves** other robots must not depend on them.
Asynchronous executions

• We need to encode internal state and/or next move of each robot, to compute executions correctly.

• Classical 2-player games are not enough: Need for partial observation
Partial observation game

We look for a strategy $f : \text{Obs}^* \rightarrow \text{Act}$
Partial observation game

s={L,-1,L,1,0}
t={0,1,0,1,1}

s={1,-1,L,0,L}
t={1,1,0,1,0}

same configuration
Partial observation game

\[ s = \{1, 1, L, -1\} \]
\[ t = \{1, 1, 0, 1\} \]

\[ s = \{L, L, 1, 0\} \]
\[ t = \{0, 0, 1, 1\} \]

\[ s = \{L, L, 1, L\} \]
\[ t = \{1, 1, 1, 1\} \]
Partial observation games

• Gathering is a reachability objective

• Need to select only fair schedulers: coBüchi condition
Partial observation games

• In a partial observation game, memory may be needed!!

• For reachability objectives, deciding existence of a memoryless winning strategy is NP-complete

⇒ SAT-solver can help
Conclusion

• Parameterized verification can be an efficient tool to assist the design of protocols for coalition of robots

• New algorithms can be discovered through synthesis, and also partial proofs of impossibility