Proving in Coq (Pactole): A user perspective

David ILCINKAS
Joint work with Sébastien BOUCHARD

CNRS, Bordeaux, France

Descartes/Estate/BAD
November 9, 2021
# Big (french) projects about DC in Coq

## PADEC
- Dedicated to **self-stabilizing algorithms**
- **Complex algorithms**
- Precise analyses, including **time complexities**
- K. Altisen, P. Corbineau, S. Devismes

## PACTOLE
- Dedicated to distributed computing by **mobile robots**
- **Large variety of models** and parameters
- **Positive** and **negative** results
- T. Balabonski, P. Courtieu, L. Rieg, S. Tixeuil, X. Urbain, ...
Problem: Ring exploration with stop

Model/context
- Team of robots
  - sensing the environment by taking a snapshot of it
  - that do not communicate
  - that are anonymous and oblivious
- Anonymous unoriented rings.

Goal: exploration with stop
- Each node must be visited by at least one robot.
- All robots must stop after finite time.
The Look-Compute-Move cycle

Look
The robot takes an instantaneous egocentric snapshot of the network and its robots, with multiplicity detection ("zero", "one", or "more than one" robots).

Compute
Based on this observation, it decides to stay idle or to move to some neighbouring node.

Move
In the latter case it instantaneously moves towards its destination.
<table>
<thead>
<tr>
<th>Identical</th>
<th>Robots have <strong>no IDs</strong>. They execute the <strong>same program</strong>.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oblivious</td>
<td>The robots have <strong>no memory</strong> of observations, computations and moves made in previous cycles.</td>
</tr>
<tr>
<td>Asynchronous</td>
<td>The time between Look, Compute, and Move operations is <strong>finite but unbounded</strong>.</td>
</tr>
</tbody>
</table>
Some precisions
Some precisions
Problem definitions

Goal: exploration with stop

- Each node must be visited by at least one robot.
- All robots must stop after finite time.

Definition: \( \text{Explo}(k, n) \)

We say that exploration of a \( n \)-node ring is possible with \( k \) robots if there exists an algorithm enabling the robots to perform exploration with stop starting from any initial configuration of the \( k \) robots without multiplicity (at most one robot per node).

More formal definition: \( \text{Explo}(k, n) \)

\[
\exists \text{Algo}, \forall \text{Adv}, \text{Config}, \text{if is\_cycle}(n, \text{Config}) \land \text{has\_robots}(k, \text{Config}) \land \text{is\_flat}(\text{Config}) \text{ then exploStop(Algo, Adv, Config)}
\]
Results in PACTOLE

Already in PACTOLE

- not $\text{Explo}(1, n)$
- not $\text{Explo}(k, n)$ if $k$ divides $n$

Our “new” result

for every positive integer $m$,

$\text{not Explo}(k, n) \implies \text{not Explo}(k \cdot m, n \cdot m)$

Sébastien Bouchard and David Ilcinkas

Proving in Coq (Pactole): A user perspective
Results in PACTOLE

Already in PACTOLE

- not \( \text{Explo}(1, n) \)
- not \( \text{Explo}(k, n) \) if \( k \) divides \( n \)

Our “new” result

for every positive integer \( m \),
not \( \text{Explo}(k, n) \) \( \implies \) not \( \text{Explo}(k \times m, n \times m) \)
When $k$ divides $n$

Lemma
Impossible to stop (and sometimes to explore) when $k \mid n$. 

Sébastien Bouchard and David Ilcinkas
Proving in Coq (Pactole): A user perspective
When $k$ divides $n$

**Lemma**

Impossible to stop (and sometimes to explore) when $k \mid n$. 

\[
\text{Diagram Image}
\]
Sketch of proof

Proved via: $\text{Explo}(k \times m, n \times m) \iff \text{Explo}(k, n)$

Transformations from small $1$ to big $m$

- From $1$ to $m$: $i \rightarrow \{n \text{ or } k\} \times j + i$, for $0 \leq j < m$
- From $m$ to $1$: $i \rightarrow i \mod \{n \text{ or } k\}$