SPIN Case Study

Automatic Greenhouse

Di Marco, Okwieka

University of Rome "La Sapienza"

July 15th, 2023

Why SPIN?

NuSMV is better suited for large but regular systems like hardware

SPIN only has explicit LTL model checking

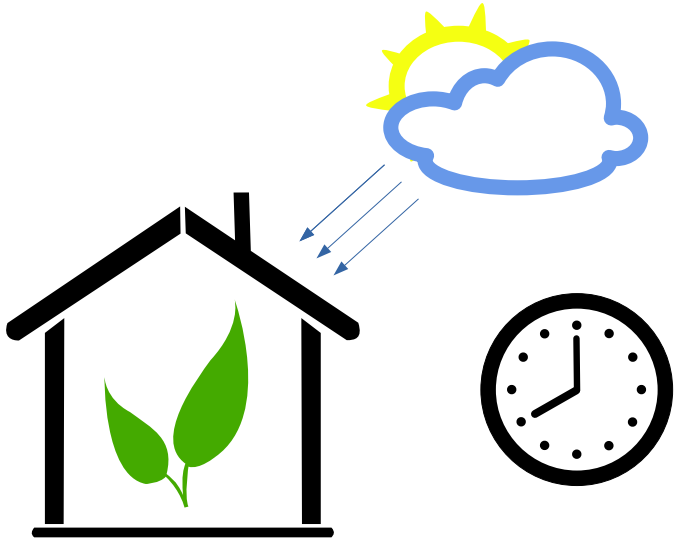
Promela has more language features (e.g. channels)

Promela has more similarities to classical programming (conditions, loops)

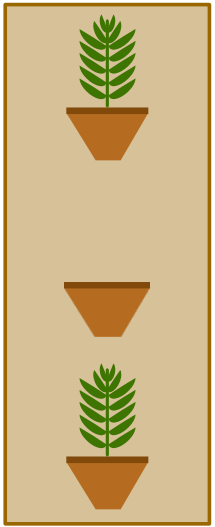
Promela has less focus on state transitions

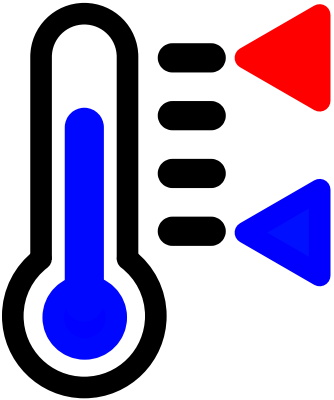
Greenhouse Model

Global overview: time and weather

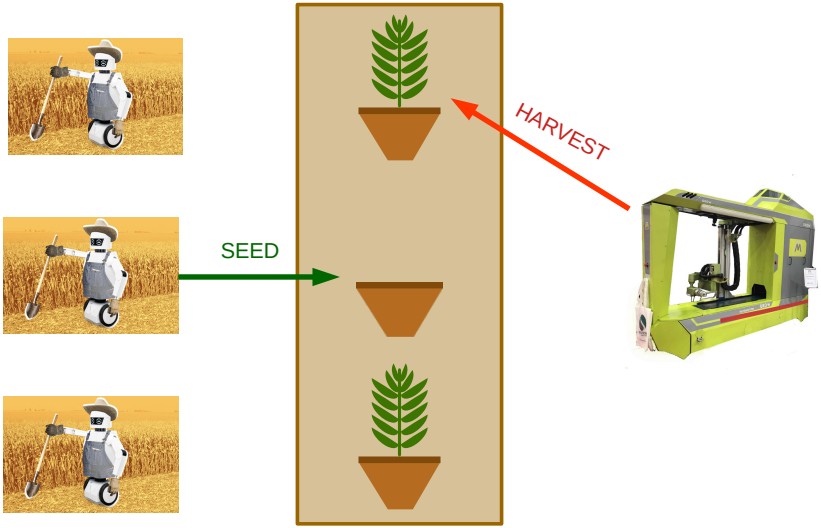


Plant Beds and Temperature





Seeder and Harvester Bots



SPIN Model

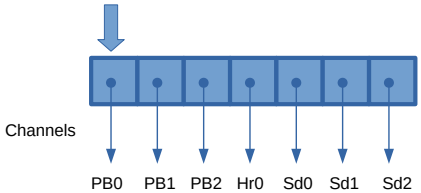
Global Clock

Need a global time, all processes must be in lockstep

Global array of unbuffered channels, one per process

Clock process cycles through and sends `TIME` messages

Reading the message and the process's action are atomic



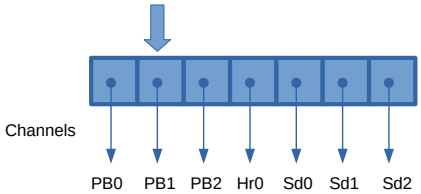
Global Clock

Need a global time, all processes must be in lockstep

Global array of unbuffered channels, one per process

Clock process cycles through and sends `TIME` messages

Reading the message and the process's action are atomic



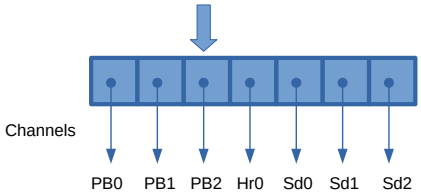
Global Clock

Need a global time, all processes must be in lockstep

Global array of unbuffered channels, one per process

Clock process cycles through and sends `TIME` messages

Reading the message and the process's action are atomic



Global Weather

Process with an internal **weather duration** decrementing each timestep

When this counter reaches 0, randomly choose (within parameters)

- **weather duration**
- constant **weather effect** on temperature (`weather_temp_rate`)

Currently only cold weather, as the greenhouse cannot counter heat (no cooling)

Plant Bed

Process that keeps the age of each plant. Can be empty (`AGE_NO_PLANT`).

The plant is grown each timestep if it has not **expired**.

If it is too **hot** or too **cold**, the plant expires immediately.

The plant state is published every timestep to a global array (plant sensors).

The plant bed checks whether it received a **plant action** message (`SEED`, `HARVEST`) and changes its state accordingly.

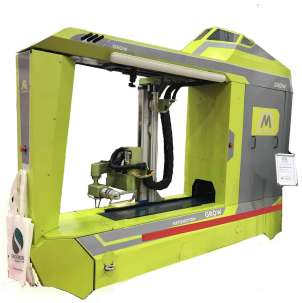
Harvester

Has a number of **Plant Beds** assigned to it based on index.

Reads its plant beds' sensors to find the age.

Policy: Harvests the plant with the **highest age** that is also **mature**.

Can only harvest **one plant per timestep**.



Seeder

Has a number of **Plant Beds** assigned to it by index.

Finds first empty plant bed based on plant sensor array.

Can plant **one seed per timestep**.

Heat Manager

Process that updates the greenhouse's **temperature** each timestep.

The temperature change depends on weather and whether the heater is turned on.

The heat manager also turns the **on** or **off** upon reaching safety thresholds.

Monitor and Properties

Expired

- At every time step the monitor checks the **number of expired plants** harvested.
- If (`n_expired_harvested` $>$ `LIMIT_EXPIRED`) the property has been violated.

Temperature

- At every time step the monitor checks the **temperature**.
- If `temp` $\in$ (`FREEZING_TEMP`, `BURNING_TEMP`) the property is satisfied.

Rate

- At every time step the **rate** of expired plants harvested.
- If $\left(\frac{\texttt{n\_mature\_harvested}+\texttt{n\_expired\_harvested}}{\texttt{n\_expired\_harvested}} > \texttt{RATE\_EXPIRED}\right)$ the property is satisfied.

Timed Properties

- Every mentioned property has a *timed* variant.
- The property is checked only for the fist `TIME_LIMIT` steps.
- After that point, it is set as satisfied.

Fairness

- Most violations happened because of continuous harsh temperature.
- SPIN only supports **weak fairness**.
- We can enforce fairness **inside the LTL formula**.
- Every mentioned property has a *fair* variant.
  - The property is checked only for fair execution.
  - The weather can't always be the *worst* and it can't always last for the *longest time*.

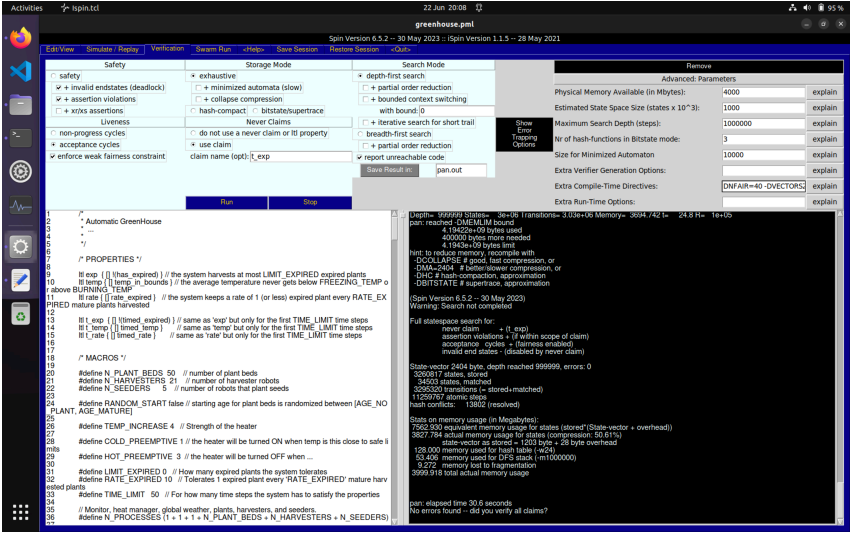
Results

Lightweight Simulation

Simulation time!

$\Rightarrow$

Heavyweight Simulation



Fin.