Intuitive Modelling and Formal Analysis of Collective Behaviour in Foraging Ants

Rocco De Nicola[1] **Luca Di Stefano**[2] Omar Inverso[3]
Serenella Valiani[1]

[1]IMT, Lucca, Italy
[2]University of Gothenburg, Sweden
[3]Gran Sasso Science Institute, L'Aquila, Italy

CMSB'23
13 Sep 2023
Luxembourg

IMT LUCCA
UNIVERSITY OF GOTHENBURG
CHALMERS UNIVERSITY OF TECHNOLOGY
GSSI
erc European Research Council
Established by the European Commission

Goal

- Describe/design/reason about collective systems

How?

- Formulate hypotheses about
  - Individual behaviour
  - Interaction mechanisms (agent-agent, agent-environment)
- Check if collective features emerge with time + interactions

Wishlist

- Modelling languages that are
  - Agent-based
  - High-level
  - Intuitive (close to the domain of interest)
  - Formally defined

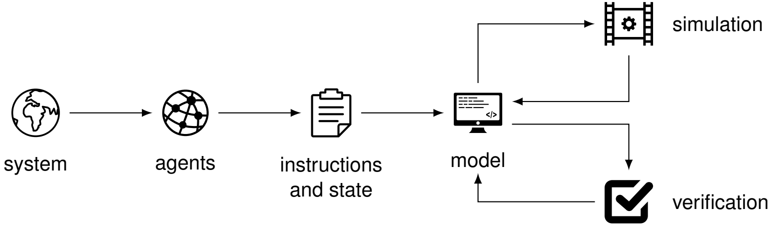
Wishlist

- Modelling languages that are
  - Agent-based
  - High-level
  - Intuitive (close to the domain of interest)
  - Formally defined
- Analysis tools and workflows that are
  - Automated
  - Intuitive (easy to use)
  - Built on top of mature off-the-shelf solutions
  - Extensible

Wishlist

- Modelling languages that are
  - Agent-based
  - High-level
  - Intuitive (close to the domain of interest)
  - Formally defined
- Analysis tools and workflows that are
  - Automated
  - Intuitive (easy to use)
  - Built on top of mature off-the-shelf solutions
  - Extensible
- Effective methodologies to put all this at work

Our methodology



- Isolate features of agents & environment
- Come up with a high-level behavioural skeleton
- Flesh out the skeleton into a model
- Get feedback from simulation/verification
- Refine the model

Why?

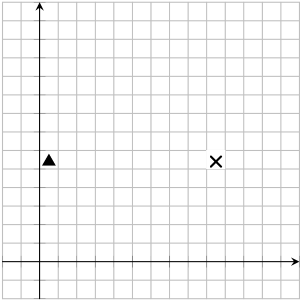
- Well-known, extensively studied
- Several interesting mechanisms at play
  - Stigmergic (pheromone-based) interaction
  - Path integration

Why?

- Well-known, extensively studied
- Several interesting mechanisms at play
  - Stigmergic (pheromone-based) interaction
  - Path integration

Our setting

- Arena: square grid of cells
- One cell contains food (×)
- One cell contains the nest (▲)
- Cells may be marked with pheromone



LAbS: System description

LAbS = simple, formal language for agent-based models

Parameters

size: Length of the sides of the arena

n: Number of ants (see line 4)

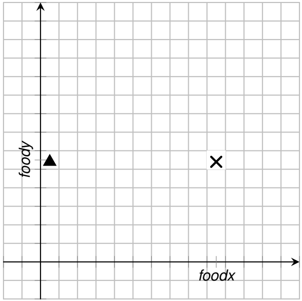
foodx, foody: Food cell coordinates

m, k: Related to ants' behaviour, initial state (coming soon)

Shared state

ph: 2-D array, tracks whether a cell is marked with pheromone

```
1 system {
2   extern = size, n, foodx, foody, m, k
3   environment = ph[size, size]: 0
4   spawn = Ant: n
5 }
```



Behaviour

- Explore surroundings for food
  - Exploration is random
  - But may be influenced by pheromone trail-following
- Bring found food to the nest
  - Dead reckoning (go back to the nest along a straight line)
  - Release pheromone along the way

Pheromone sensing

1. Sample two random cells within range m
2. If either cell is marked, move there;

   Otherwise move to a random cell within range

LAbS: Ant description

```
agent Ant {
  interface = x: 0..size; y: 0..size;
              nextX: 0; nextY: 0

  Behavior = Explore; GoHome; Behavior

  Explore =
    x ≠ foodx or y ≠ foody ⇒ (
      SmellPheromone; Move; Explore)

  Move =
    (nextX = x and nextY = y ⇒ {
      dX, dY := [−m..m+1], [−m..m+1];
      nextX ← x+dX;
      nextY ← y+dY;
      nextX ← max(nextX, 0);
      nextY ← max(nextY, 0);
      nextX ← min(nextX, size−1);
      nextY ← min(nextY, size−1)
    });
    x, y ← nextX, nextY

  SmellPheromone = {
    dX := [1..m+1];
    dY := [1..m+1];
    testx1, testy1 := min(x+dX, size−1), min(y+dY, size−1);
    testx2, testy2 := max(x−dX, 0), max(y−dY, 0);

    nextX ← if ph[testx1, testy1] then testx1 else
            if ph[testx2, testy2] then testx2 else x;
    nextY ← if ph[testx1, testy1] then testy1 else
            if ph[testx2, testy2] then testy2 else y
  }

  GoHome =
    x ≠ 0 or y ≠ foody ⇒ ({
      ph[x,y] ⇐1;
      x ← max(0, x−1)
    }; GoHome)

}
```

Assumptions

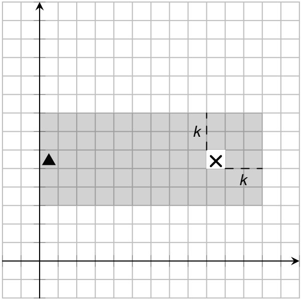
Additional constraints on the initial state

- At least one ant starts at the food location
- All the others start “far” from the shortest path (shaded area) between food and nest

LAbS: Quantified predicate in a separate section of the model

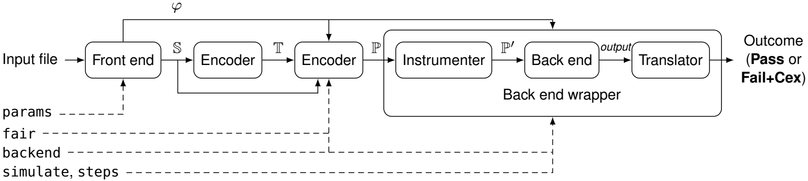
```
assume {
  FoodAnt = exists Ant a,
    (x of a = foodx) and (y of a = foody)

  FarFromThePath = forall Ant a,
    ((x of a = foodx) and (y of a = foody)) or
    (x of a > foodx + k) or
    (y of a > foody + k) or
    (y of a < foody − k)
}
```



A tool to verify/simulate LAbS models[1]

- Converts model into a symbolic intermediate representation
- Converts IR into imperative programs (here, sequential C)
- Reuses off-the-shelf analysis tools (here, SAT-based BMC[2])

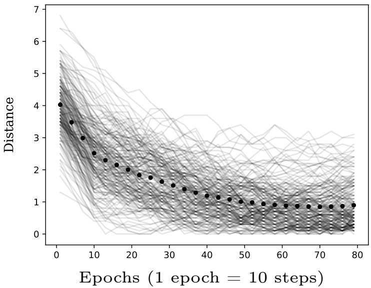


[1] https://github.com/labs-lang/sliver
[2] https://www.cprover.org/cbmc

Parameter values

size	Lenght of the arena's sides	20
foodx	Food *x*-coordinate	10
foody	Food *y*-coordinate	10
k	Initial distance from trail	2
n	Number of ants	10
m	Ants' movement range	1
B	Simulation bound	800
	Number of simulations	200

Average ant-trail distance

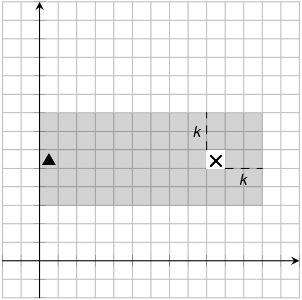


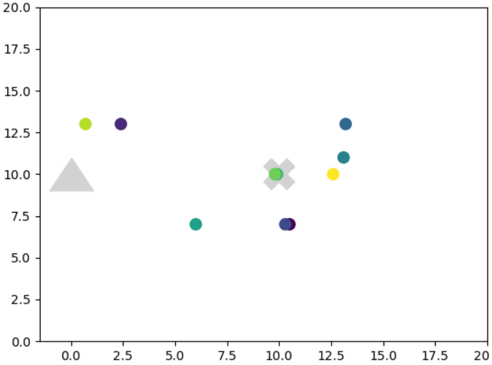
- Ants end up close to the pheromone trail in most simulations
- …even though pheromone sensing is rather simple (nondeterministic, memoryless)

Verification

Now, let us specify that we would like *every* ant to be *within* the shaded region after a certain number of steps B

```
check {
  ShortestPath =
    after B forall Ant a,
    (x of a ≤ foodx + k) and
    (y of a ≥ foody − k) and
    (y of a ≤ foody + k)
}
```

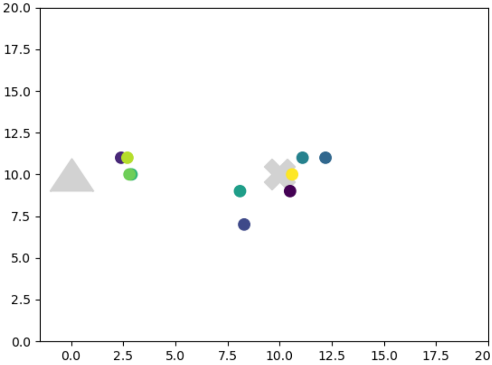




(1 frame = 10 epochs = 100 steps)

Initial state: ant ● finds food

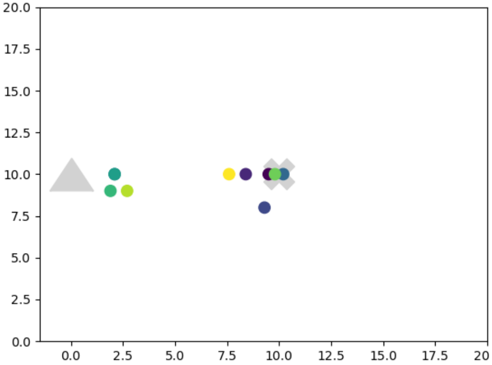
Verification results



(1 frame = 10 epochs = 100 steps)

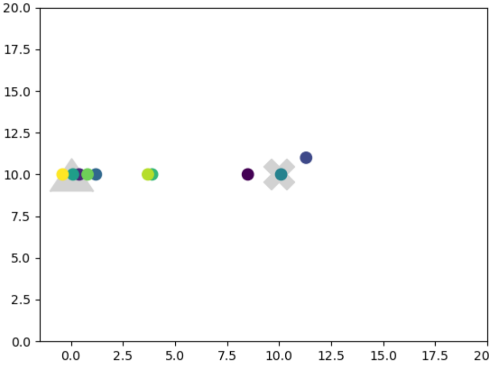
Ant ● goes from × towards ▲, leaves trail

Verification results



(1 frame = 10 epochs = 100 steps)

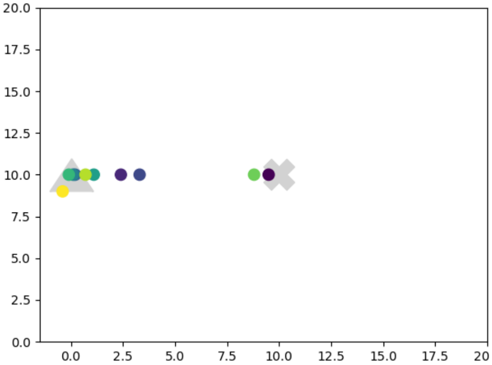
Pheromone trail affects other ants



(1 frame = 10 epochs = 100 steps)

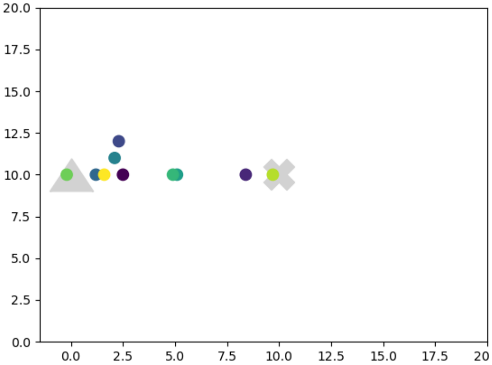
Several ants find food, go back to nest

Verification results



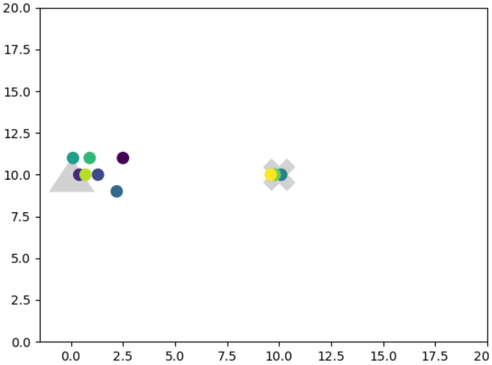
(1 frame = 10 epochs = 100 steps)

Ants (more or less) stay on track



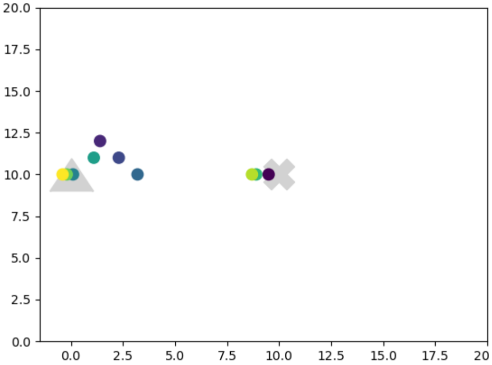
(1 frame = 10 epochs = 100 steps)

Ants (more or less) stay on track



(1 frame = 10 epochs = 100 steps)

Ants (more or less) stay on track

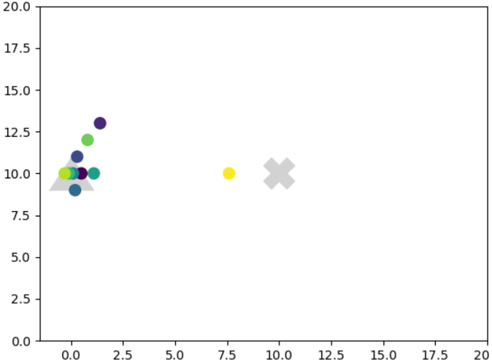


(1 frame = 10 epochs = 100 steps)

Ant ● starts straying from shortest path

Verification results

CHALMERS UNIVERSITY OF TECHNOLOGY | UNIVERSITY OF GOTHENBURG



(1 frame = 10 epochs = 100 steps)

Final state: ● is too far away

We can also use verification to generate “interesting” traces

Example. If *exactly one ant* starts at ×, can *every* ant end up close to the trail (after B steps)?

Verification (part 2)

We can also use verification to generate “interesting” traces

Example. If *exactly one ant* starts at ×, can *every* ant end up close to the trail (after *B* steps)?

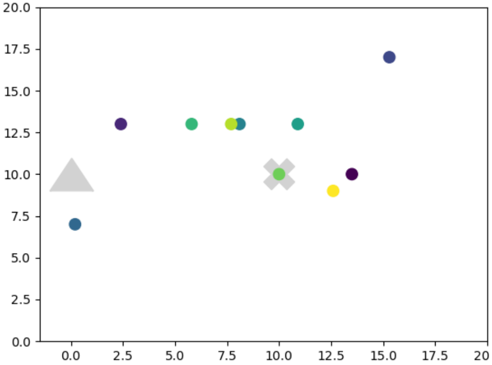
Verify against the negation of the property:

```
assume {
  FoodAnt =
    exists–unique Ant a,
      (x of a = foodx) and
      (y of a = foody)

  FarFromThePath = ...
}

check {
  NegShortestPath =
  after B exists Ant a,
    (x of a > foodx + k) or
    (y of a < foody − k) or
    (y of a > foody + k)
}
```

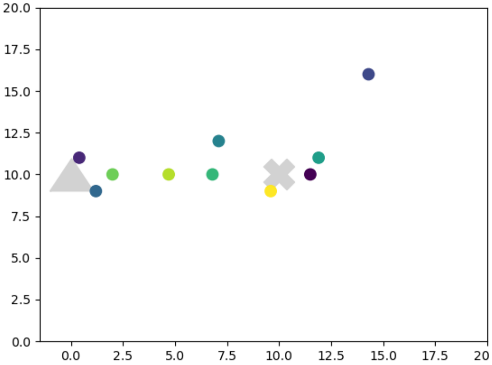
Verification (part 2): Results



(1 frame = 10 epochs = 100 steps)

Initial state: ant ● finds food

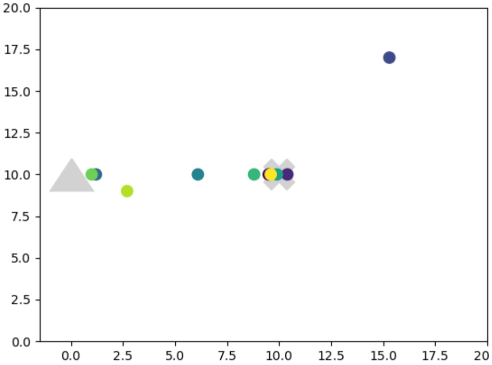
Verification (part 2): Results



(1 frame = 10 epochs = 100 steps)

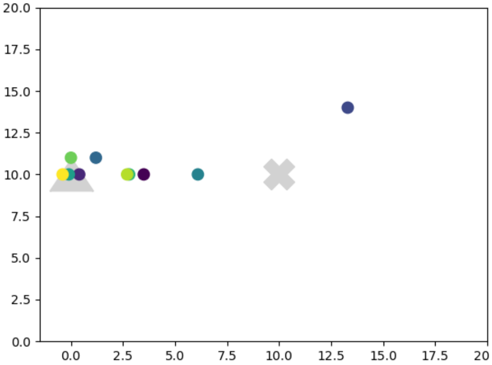
Ant ● goes from ✕ towards ▲, leaves trail

Verification (part 2): Results



(1 frame = 10 epochs = 100 steps)

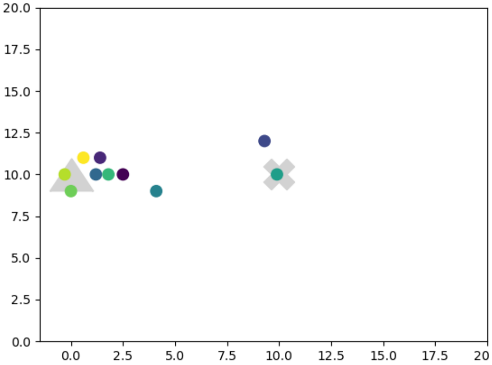
Other ants explore arena, get on the trail



(1 frame = 10 epochs = 100 steps)

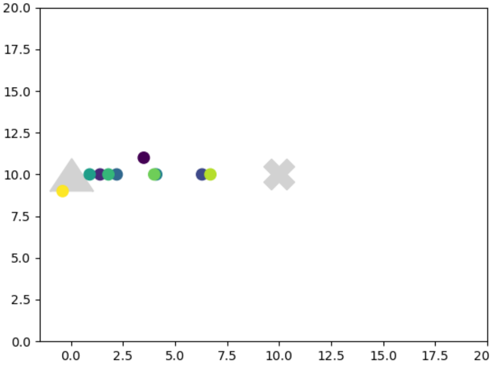
Other ants explore arena, get on the trail

Verification (part 2): Results



(1 frame = 10 epochs = 100 steps)

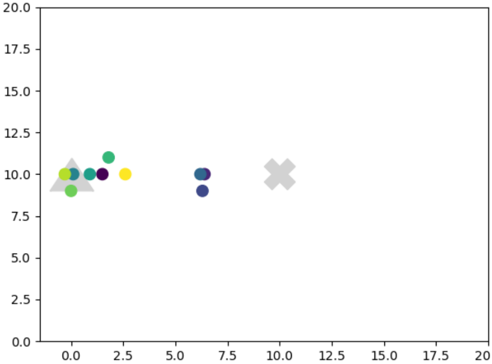
Ant ● starts getting closer



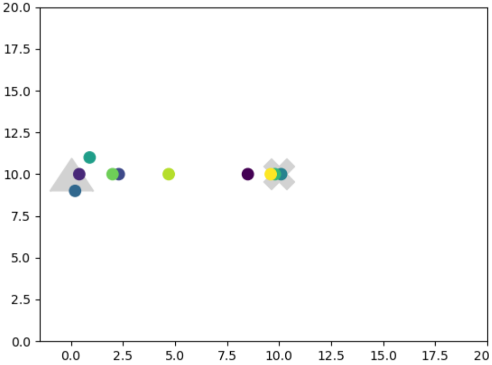
(1 frame = 10 epochs = 100 steps)

Ant ● starts getting closer

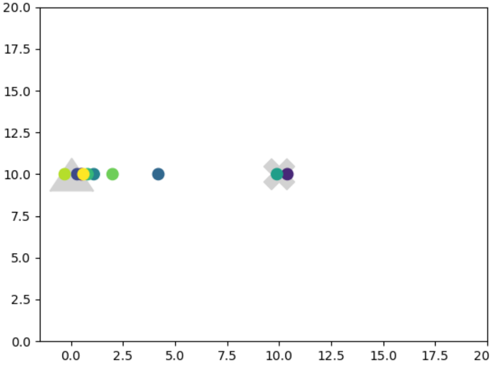
Verification (part 2): Results



(1 frame = 10 epochs = 100 steps)



(1 frame = 10 epochs = 100 steps)



(1 frame = 10 epochs = 100 steps)

Conclusion

- Agent-based modelling of collective systems requires appropriate languages and tools
- These need to be supported by an adequate methodology
  - Gradual refinement of informal descriptions into formal models
  - Analysis-driven, iterative improvements to the model
- Simulation and exhaustive techniques complement each other

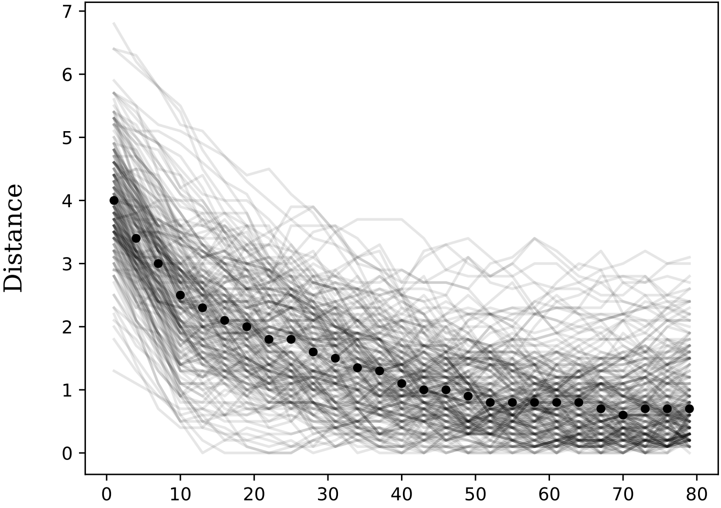
Future work

- Support more expressive properties (e.g., full LTL)
- Improve simulation/verification performance
- Implement runtime verification, statistical model checking, ...
- Look for new case studies

Backup slides

Simulation results: Median distance

(Omitted from the paper)



Simulation results: Box plot
(Omitted from the paper)