Ant Simulator:
Swarm Intelligence using Goal-based Agents & A*

Christopher D. Canfield
Project Overview

- Ant colony simulator: Swarm intelligence, goal-based agents, A* pathfinding

- Swarm intelligence: Many independent agents appear to coordinate actions, as if controlled by a higher-level coordinating AI

- Ants are a good example of this: they use pheromones to notify other ants about the location of food\(^1\) and “to alert nestmates for colony defense.”\(^2\)

- *ants, termites and stingless bees mark the route between their nest and discovered food sources with a chemical (pheromone), thus indirectly leading nestmates to the food.*\(^1\)


• World Creator: Random map generator

• Navigation Graph

• Ants: Goal-based agents
  • The most complex component of the simulation

• A*

• World has ants, navigation nodes & edges, pheromones, obstacles, an ant hill, and piles of green food

• Technical:
  • C++
  • SFML, Boost, POCO libraries for graphics, input processing, UUID, circular array...
  • Microsoft Unit Testing Framework
First: Did It Work?
• Generates random maps with up to 900 navigation nodes and thousands of edges

• Places the ant hill, 10-50 obstacles (rocks), and 5-15 food piles throughout the map

• Places 30-60 ants in the ant hill

• Accepts a seed from the user; billions+ possibilities
Navigation Graph

- **Node**
  - Used for A* start and end points
  - Static objects (rocks, ant hill, food piles) must be on a node

- **Edge**
  - A* g cost
  - Ants release pheromones onto edges
• Ants have goals and subgoals
• Ants need to eat, and will die if unable to eat for a long period of time

Gray ants have died

• Ants have a desire to bring as much food to the colony as possible
• Ants can sense the following:

  • The current node, if any, plus connected edges (or the current edge if not on a node)

  • Pheromones, on edges
    • Pheromones are released on edges when ants pass them while carrying food
    • Pheromones degrade over time if not replenished, and eventually wear away completely. This is graphically simulated using red->orange->yellow lines

  • Food pile, if one exists on the current node
    • The ant cannot sense food piles that are not on the current node
    • Food piles have a limited amount of food, and do not regenerate
Ants: Knowledge Base

- Ants:
  - Do not know ahead of time where food is located.
  - Remember the location of up to 3 food piles. Certain goals pop off the most recently found food pile from the knowledge base.
  - Know where the ant hill is located.
  - Implicitly know where obstacles are located, because obstacles eliminate the navigation nodes and edges that are blocked.
Ants: Goals & Subgoals

• The current goal may be set by the ant, when the previous top-level goal has completed, or by other goals and subgoals.

• Ants have 7 goals and subgoals (listed from simplest subgoals to most complex top-level goals):

1. AntMoveToNode
   • Instructs an ant to move from one node to an adjacent node
   • The simplest subgoal
   • Used by many other goals

2. AntFollowPath
   • Takes a target node, generates a path using A*, and moves the ant along the path
   • Uses AntMoveToNode to move between individual nodes
3. AntFindFood
   - Instructs the ant to attempt to find food by doing the following:
     - Check the locations of the three most recently found food piles, starting with the most recent location
     - If that does not work, select a series of random nodes in an attempt to find food or pheromones
     - If at any time pheromones are encountered, follow that path
   - Uses the AntMoveToNode subgoal

4. AntGoHome
   - Instructs the ant to return to the ant hill
   - Uses the AntFollowPath subgoal
5. **AntExplore**
   - Instructs the ant to explore random paths
   - If food is encountered, its location is noted and then the ant continues
   - The ant does not follow pheromones when it is pursuing this goal
   - Uses the AntMoveToNode subgoal
   - Top-level goal: Selected 1% of the time when a new top-level goal is needed, if the ant is not hungry

6. **AntForage**
   - Instructs the ant to search for food
   - If food is found, it is brought back to the ant hill, where it is deposited
   - A pheromone trail is released while the ant is holding food
   - Uses the AntFindFood subgoal
   - Top-level goal: Selected 99% of the time when a new top-level goal is needed, if the ant is not hungry
7. AntEat

- Instructs the ant to find food for consumption
- First goes to the ant hill, using the AntGoHome goal; if food is available, it is eaten and the goal is completed
  - Pheromones are ignored in this state
- If no food is available there, it uses the AntFindFood subgoal to search for food in the locations of the three most recently encountered food piles.
  - Pheromones are followed in this state
- If no food is found in any of the last three food pile locations, and no pheromones have been encountered, the ant will randomly search locations for food and pheromones
  - Pheromones are followed in this state
- Top-level goal: Selected if the ant is hungry
## Ants: PEAS

<table>
<thead>
<tr>
<th>Performance</th>
<th>Environment</th>
<th>Actuators</th>
<th>Sensors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Is alive</td>
<td>Partially viewable (knows nav graph, but does not know food or pheromone locations)</td>
<td>Move</td>
<td>View current navigation node or edge</td>
</tr>
<tr>
<td>Is not hungry</td>
<td>Partially dynamic (nav graph is static, but food disappears and pheromones appear and wear out)</td>
<td>Pick up &amp; drop food</td>
<td>Smell pheromones on current edge, or on all edges connected to current node</td>
</tr>
<tr>
<td>Colony has food</td>
<td>Navigation graph with nodes and up to 8 edges per node</td>
<td>Lay pheromone trail</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Obstructions, food piles, ant hill, other ants (no direct interaction), pheromones</td>
<td>Eat food</td>
<td></td>
</tr>
</tbody>
</table>

Christopher D. Canfield
A* method takes a heuristic function pointer

Implemented Manhattan and Straight-line heuristic functions

The simulation uses Straight-line, because the nav graph allows 8 directions of movement

Despite extensive testing, initial implementation had flaws:
- Paths could contain a loop (clearly not the optimal path)
- Rarely, paths weren’t fully connected (big problem!)
- These issues were uncovered when I ported this to the Wumpus World

Problem: I wasn’t properly accounting for g cost (cumulative distance from start). Fixing that resolved the issues.
Light-blue lines are paths calculated by A*
The user sets the random map seed, and can set the following to display or not:

- Nav graph
- Pheromones
- Path for each ant
- Dead ants
- Background
The Simulator

- An ant can be clicked to receive additional information, including:
  - Current goal & all current subgoals
  - Hunger
  - Whether the ant is alive
  - The ant’s path

- Colony-wide information is printed to the console every 10 seconds, including:
  - Total number of ants (alive + dead)
  - Total number of living ants
  - Amount of food in ant hill
• Heavily test-driven: 123 unit tests in Visual Studio + 23 related JUnit tests
  • Includes 26 unit tests related to search, plus another 23 JUnit tests for
    the A* implementation used in my Wumpus Explorer project (different
    language, algorithm ported from Ant Simulator)
  • Microsoft Unit Testing Framework

• Visual Studio static code analysis
The Simulator: Verification & Testing

- 3 test applications:
  - GUI Tester: Tests gui implementation & related aspects, including various user input events, observer pattern implementation, drawing various items to screen, etc
  - A* Tester: Text-based output of paths generated by A*. Separate from the unit tests.
  - Ant Goal Tester: Tests all ant goals using 7 test agents
Demo