EVOLVING ANTS

Enrique Areyan
School of Informatics and Computing
Indiana University
January 24, 2012

About Me & Evolving Ants (EVA)

- Enrique Areyan – Master Student (CS) at SoIC. You can contact me at **eareyan@umail.iu.edu**. Also, check out my web page for this project!

  http://www.enriqueareyan.com/evolvingants

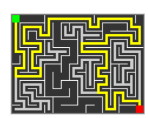
- EVA - Final Project for *I585 BionInspired Computing course*, under the supervision of Dr. Luis Rocha.
- EVA was built on top of AntFramework, a project I did as my undergraduate thesis in UCV (Universidad Central de Venezuela)

Agenda

- Preliminaries
- Motivation
- What is EVA?
- The algorithm
- Evolutionary Strategy
  - Simple Genetic Encoding
  - L-System Genetic Encoding
  - Genetic Operators
- Experiments
- Results
- Conclusion and Future Work

Preliminaries

- For the purpose of this project, EVA works in a perfect Maze (with some effort it can be made to work in other types of environments)
- A perfect Maze has one and only one path from any cell in the maze to any other cell. A cell is a node in a graph.



Motivation

- Is it true that by providing genetic information to each individual ant in an Ant Colony Optimization algorithm solutions may be found faster in environments that do not change often (e.g. Maze, the topology in a commercial network, etc)?
- How do we go about encoding such genetic information?
- How would such an strategy compare with traditional ant colony optimization and genetic algorithms?

What is EVA?

- EVA is a novel combination of traditional ant colony optimization (ACO) algorithm with genetic algorithm (GA).
- An ant is provided with genetic information (individual “memory”) that it uses together with *stigmergic* information (shared “memory”) to construct solutions to a path in a maze.
- The genes of an ant **indirectly** encode the series of steps it took to reach the source of food from its nest.

EVA

- Population of ants will be evolved using a GA, whose fitness function will test the "effort" ants made (i.e., number of steps) and cost function of the path found.
- The best or more “fit” ants will be reproduced using roulette wheel selection, random crossover point, fair probability of crossover and low probability of mutation.
- In latter runs of the algorithm, an ant will use both its internal and share memory to make the decision about which nodes to visit.

EVA – Decision Rule

Probability of ant *k* to visit node *j* when visiting node *i*

α and β control the influence of the genetic and pheromone trail on the ant's decision

$$p(k)_{i,j} = \alpha \cdot G(k) + \beta \cdot T(i,j)$$

Genetic information of ant *k*
Different encodings are possible

Stigmergic information on node *i* going to node *j*

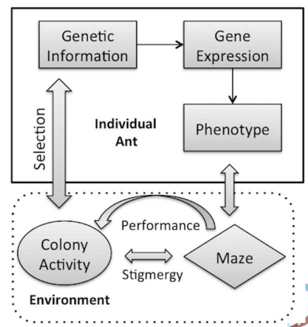
Note that EVA generalizes ACO:
By setting α=0, we have traditional ACO
By setting β=0, we have traditional GA

The algorithm

```
Initialize Parameters (ACO,GA,EvolvingAnts)
Initialize Random Population of Ants
While stop-condition-not-reached do
       Node = start node
For each Ant in Population do
       While node-is-not-goal do
              Node = selectNextNode(node,ant)
Run Genetic Algorithm with solutions found
```

```
Function selectNextNode(node n,ant a)
       For each node reachable from n do
              Individual-Decision = a.develop()
              Collective-Decision = getPheromoneTrail(n)
              R = uniformRandomNumber(0,1)
              Choose node n' such that R > (Individual-
Decision + Collective-Decision) / normalizer factor
       Return n'
```

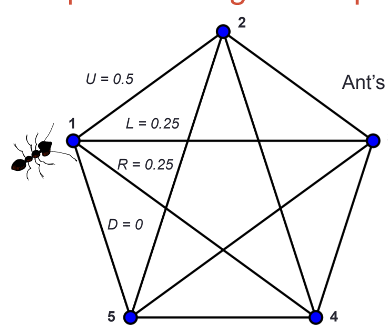
Evolutionary Strategy



Simple Encoding

- An ant's gene is a string of fix length from the four-letter alphabet $\Sigma = \{U, D, L, R\}$. A random string of a fixed length is generated and assigned as the ant's genes.
- When making a decision, the function *G* is just the count of each letter in the string, e.g., if the ant's gene is the string "UUDL", then:
  - G("U", "UUDL") = 0.5, G("D", "UUDL") =G("L", "UUDL") = 0.25, and G("R", "UUDL") = 0
- This encoding is mainly used as a control case to test against more sophisticated strategies such as L-System.

Simple Encoding - Example



Ant's Gene = "UULR"

$p(k)_{i,j} = \alpha \cdot G(k) + \beta \cdot T(i,j)$

$p(k)_{1,2} = \alpha \cdot 0.5 + \beta \cdot T(i,j)$

$p(k)_{1,3} = \alpha \cdot 0.25 + \beta \cdot T(i,j)$

$p(k)_{1,4} = \alpha \cdot 0.25 + \beta \cdot T(i,j)$

$p(k)_{1,5} = \alpha \cdot 0 + \beta \cdot T(i,j)$

L-System Encoding

- Ant's genes G=(Σ, σ,F), where
  - Σ={U,D,L,R}
  - σ∈Σ* , is the axiom, which is initialized to be a random string from Σ* the set of all possible strings of alphabet Σ
  - F = $\{F_1,F_2,F_3,F_4\}$, four rules that map from each letter in Σ to Σ*. Each rule is initialized randomly
- This is your typical L-System but with four production rules, each one mapping from an action in the maze to a string.

L-System Expression

- Unlike the simple gene, before the ant uses the information stored in its genes this must go trough a "developmental" phase:
- Iteration of the L-System a fix number of times resulting in string S
- The weights used as G(k) are now the proportion of each letter in the string S, similar to the simple gene.

Why use an L-System?

- There is a clear genotype/phenotype mapping between the system's axiom and final string, through a developmental phase.
- Encode of self-similar information which can be useful in a maze structure where a lot of the decision are similar
- Compact a lot of information effectively.
- The L-System stands as a suitable metaphor for an ant's genetic information.

Genetic Operations

- Those ants that constructed the best solution (so far) are selected for reproduction. How does this work?
- Simple encoding: the best ants' genes are reproduced taking a random crossover point and allowing for mutation
- L-System encoding: The crossover operator takes two genes G_u,G_v, and a random number *r* uniformly distributed between 0 and 3, and replaces the rules (F_0,F_r) of gene G_u with rules (F_r,F_3) of gene G_v. A single gene can also be mutated, meaning that a rule F_j will be changed randomly.

Experiments

- EVA was tested on mazes of different sizes, i.e., from 20 cells to 300 cells, in increment of 10 cells.
  - 20,30,40, …, 290 and 300.
- For each maze, different topologies of the mazes were tested. For instance, for the maze of dimension 40, three different topologies were created having 5, 10 and 20 cells per row.
- For each of these topologies, 10 different mazes were tested. For convenience, node *0* is the nest and node *n* is the food source.

Experiments

Parameters used to test EVA

Simple gene-EVA (A-D) L-System-EVA (E-H)

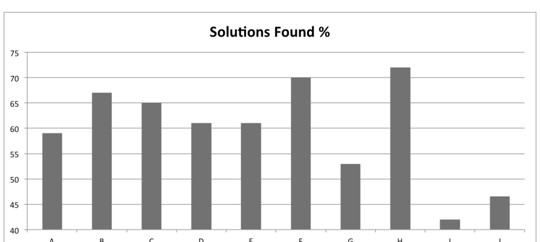
Parameter Set	A	Parameter Set	B	Parameter Set	E	Parameter Set	F
ACO		ACO		ACO		ACO	
numberOfAnts	15	numberOfAnts	20	numberOfAnts	15	numberOfAnts	20
GA		GA		GA		GA	
maxNumIterations	35	maxNumIterations	50	maxNumIterations	35	maxNumIterations	50
EVA		EVA		EVA		EVA	
maxNumIterations	50	maxNumIterations	100	maxNumIterations	50	maxNumIterations	100
agImportance	0.5	agImportance	0.5	agImportance	0.5	agImportance	0.5
acoImportance	0.5	acoImportance	0.5	acoImportance	0.5	acoImportance	0.5
Parameter Set	C	Parameter Set	D	Parameter Set	G	Parameter Set	H
ACO		ACO		ACO		ACO	
numberOfAnts	15	numberOfAnts	20	numberOfAnts	15	numberOfAnts	20
GA		GA		GA		GA	
maxNumIterations	35	maxNumIterations	50	maxNumIterations	35	maxNumIterations	50
EVA		EVA		EVA		EVA	
maxNumIterations	50	maxNumIterations	100	maxNumIterations	50	maxNumIterations	100
agImportance	0.75	agImportance	0.25	agImportance	0.25	agImportance	0.75
acoImportance	0.25	acoImportance	0.75	acoImportance	0.75	acoImportance	0.25

EVA (parameters sets A through H)

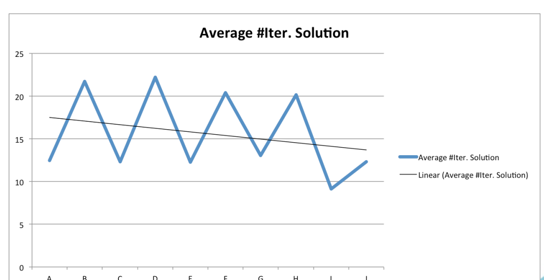
Parameter Set	I	Parameter Set	J
ACO		ACO	
numberOfAnts	15	numberOfAnts	20
EVA		EVA	
maxNumIterations	35	maxNumIterations	50
agImportance	0	agImportance	0
acoImportance	1	acoImportance	1

ACO (parameters I and J)

Results



Results



Conclusion

- EVA is a viable optimization algorithm.
- Results of the experiments performed using EVA suggests that this algorithm outperforms a pure ACO algorithm when tested under the conditions previously described.
- The performance of EVA varies drastically with the choice of the genetic encoding of the evolutionary algorithm.
  - An direct genotype/phenotype mapping (Simple Gene) is outperformed by another strategy that uses an intermediary, growth phase.

Conclusion – Future Work

- Other evolutionary strategies can be tested:
  - Cellular automata
  - Boolean Networks
  - etc
- Other functions mapping from the string to actions in the environment can be created and tested.
- Other kind of environments can also be tested.
- **Thank you for your attention!**