Using Artificial Life Techniques to Generate Test Cases for Combinatorial Testing

Presentation: TheinLai Wong

Authors: T. Shiba, T. Tsuchiya, T. Kikuno
Osaka University

Backgrounds

- Testing is an important but cost-consuming process.
  - An efficient and cost-effective methodology is required.
- **Combinatorial Testing** has received recent attention.

Combinatorial Testing

- Covers all combinations of values for any ***k*** of the input parameters.
  - Rationale: Many faults are caused by interactions of a small number of parameters.

Example

k =2, All parameters have three values.

	P1	P2	P3	P4
Test 1	1	1	1	1
Test 2	1	2	2	2
Test 3	1	3	3	3
Test 4	2	1	3	2
Test 5	2	2	1	3
Test 6	2	3	2	1
Test 7	3	1	2	3
Test 8	3	2	3	1
Test 9	3	3	1	2

Number of Tests

$3^4 = 81$ → 9

Exhaustive Testing → **Combinatorial Testing**

Objective and Results

- Problem with Combinatorial Testing
  - Constructing a minimum test set is computationally infeasible (NP-Complete).

- Our Results
  - Developing new heuristic algorithms based on artificial life techniques; namely,
    - **Genetic Algorithm (GA)**
    - **Ant Colony Algorithm (ACA)**

The AETG Algorithm

- An algorithm used in AETG, a commercial testing tool of *Telcordia* (http://aetgweb.argreenhouse.com/).
- Our algorithms are modifications of this algorithm.

TestSet := Empty;
repeat {
  Construct a new test.
  Add the test to *TestSet*.
}
until (all combinations are
        covered by *TestSet*)

ϕ

Test 1	1	1	1	1

Test 1	1	1	1	1
Test 2	1	2	2	2

⋮

Sub-Problem Tackled

- Input
  - A test set.

Test 1	1	1	1	1
Test 2	1	2	2	2

- Output
  - A single test

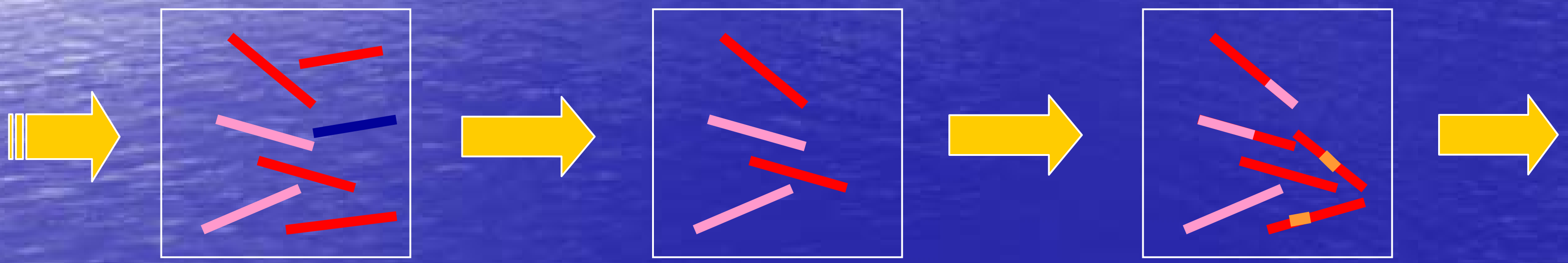
Test 3	2	1	3	2

- Objective
  - To cover as many new combinations as possible

- AETG
  - A random greedy algorithm
- **Ours**
  - **Genetic Algorithm (GA)**
  - **Ant Colony Algorithm (ACA)**

Genetic Algorithm (GA)

- A metaheuristic algorithm that mimics the natural selection process
  - Solution = Chromosome



Selection Selects good solutions

Crossover or **Mutation** Creates a new generation

Proposed Genetic Algorithm (GA)

- Chromosome
  - A Test Case

=

1	3	2	1	2	3

- Crossover operator
  - Exchanges the values for each parameter with probability 0.5 (Uniform Crossover)

1	3	2	1	2	3
2	1	3	2	3	1

→

1	1	3	1	3	3
2	3	2	2	2	1

- Mutation Operator
  - Randomly changes the value of each parameter

1	3	2	1	2	3

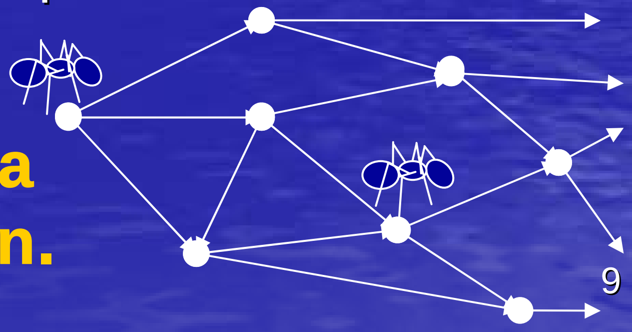
1	2	2	1	2	1

Ant Colony Algorithm (ACO)

- A metaheuristic algorithm that mimics the pheromone-trail building of an ant swarm

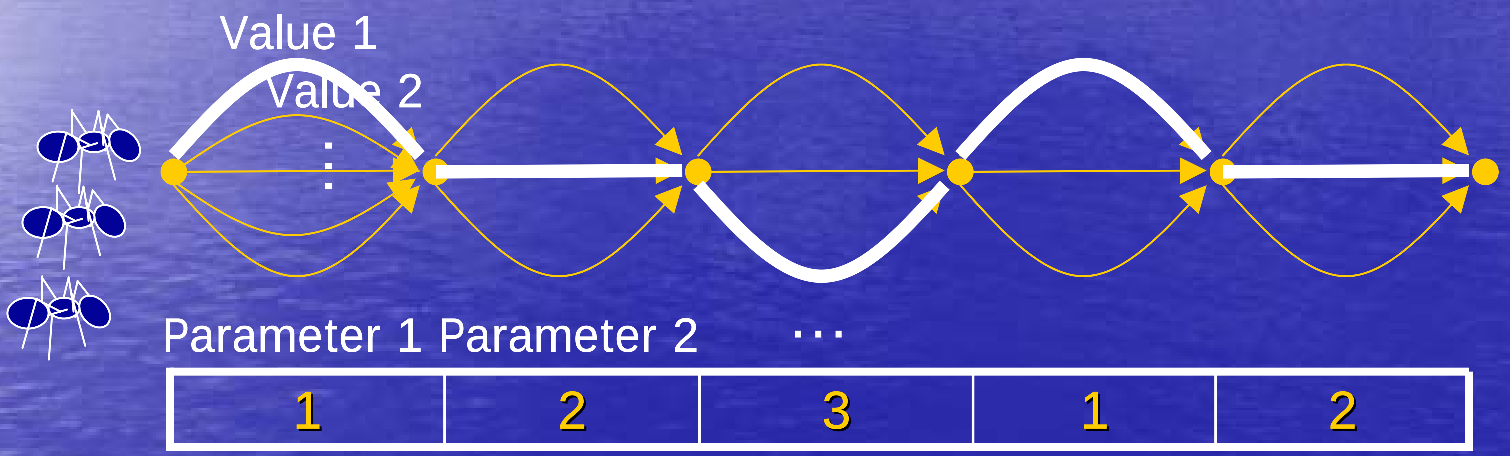
- Ants randomly travel in the graph representing the solution space.
  - A path followed by an ant represents a solution.
  - An arc with a high amount of pheromone is chosen with high probability.
- Pheromone is deposited on the path followed.
  - The amount of pheromone deposited depends on the quality of the corresponding solution.

Ants will eventually converge to a path representing a good solution.



Proposed Ant Colony Algorithm (ACA)

- Graph representation of the solution space
  – Each path from the leftmost node to the rightmost node represents a test case.



- The amount of pheromone deposited
  – Proportional to the number of the newly covered combinations

Experiments

- Algorithms compared
  - The original AETG algorithm
  - The two proposed algorithm
    - Genetic Algorithm (GA)
    - Ant Colony Algorithm (ACA)
- Termination Condition
  - No more than 1000 candidate solutions were evaluated in creating a single test.
- Problem Instances
  - 16 instances were taken from the literature.
  - 300 runs were executed for each instance.

Results (Pairwise Combinations, k=2)

Number of Tests Required to Cover All Combinations

Problem	AETG*	GA	ACA
3^{4}	11	9	9
3^{13}	17	18	18
2^{100}	12	14	14
10^{20}	197	227	232
$5^{1}3^{8}2^{2}$	19	17	17
$7^{1}6^{1}5^{1}4^{5}3^{8}2^{3}$	45	43	43
$5^{1}4^{4}3^{11}2^{5}$	30	26	27
$6^{1}5^{1}4^{6}3^{8}2^{3}$	34	33	34
$4^{15}3^{17}2^{29}$	35	38	38
$4^{1}3^{39}2^{35}$	25	29	28

1 5-valued parameter
8 3-valued parameters
2 2-valued parameters

*The figures are taken from Cohen et al. "New techniques for designing qualitatively independent systems," ICSE 2003.

Results (3-Way Combinations, $k=3$)

Number of Tests Required to Cover All Combinations

Problem	AETG*	GA	ACA
3^6	47	38	36
4^6	105	64	64
6^6	343	335	335
10^6	1508	1491	1490
5^7	229	221	221

*The figures are taken from Cohen et al. "New techniques for designing qualitatively independent systems," ICSE 2003.

Conclusions

- Two algorithms were proposed for combinatorial testing.
  - Genetic Algorithm
  - Ant Colony Algorithm
- Experiments were conducted.
  - For the case k =2 (pairwise combinations), the two algorithms exhibited comparable performance to the AETG algorithm.
  - For the case k =3 (3-way combinations), the two algorithms outperformed the AETG algorithm.