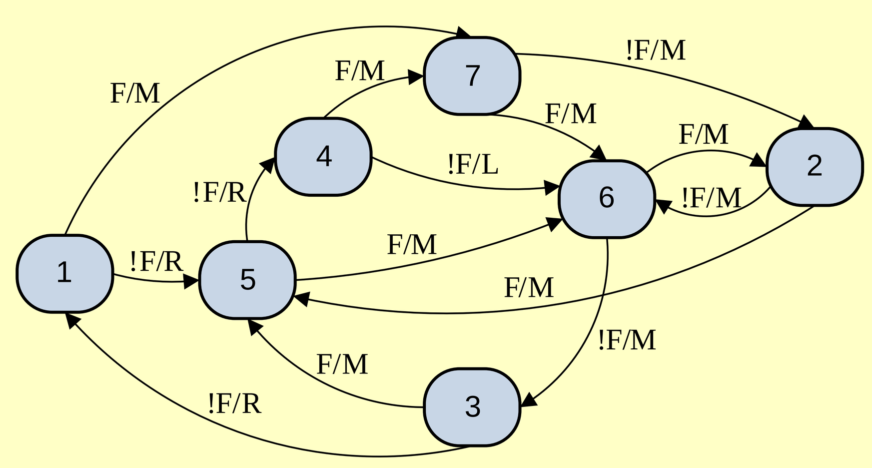
Learning Finite-State Machines with Ant Colony Optimization

Daniil Chivilikhin and Vladimir Ulyantsev
St. Petersburg National Research University of Information Technologies, Mechanics and Optics
Computer Technologies Department

Problem Definition

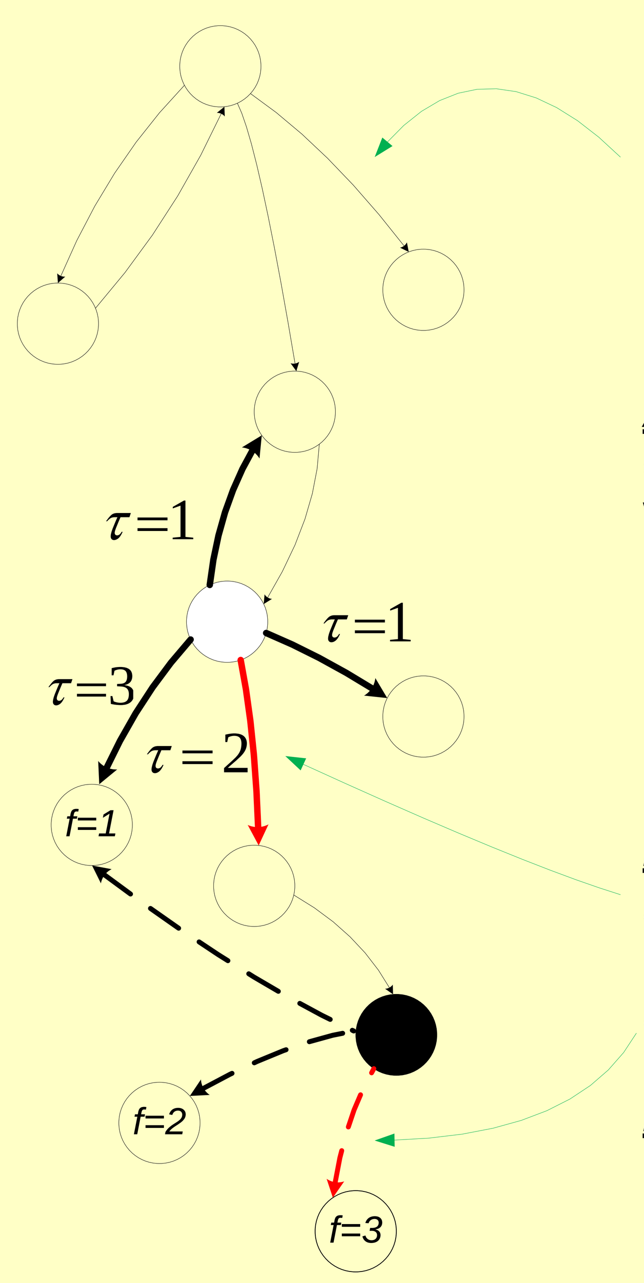
A **finite-state machine** (FSM) is a sextuple <S, Σ, Δ, δ, λ, s_0>, where:

- S – set of states
- Σ – set of input events
- Δ – set of output actions
- $\delta : S \times \Sigma \to S$ – transition function
- $\lambda : S \times \Sigma \to \Delta$ – actions function
- $s_0 \in S$ – initial state



- The input data consists of the number of states N, a set of events Σ, and set of actions Δ of the target FSM. Input data also specifies the **fitness function** (FF) f defined for any FSM and a boundary value of this function f_0
- The **goal** is to build an FSM with a value of $f \geq f_0$

Proposed algorithm



Search space representation

- Directed graph G
- Nodes – FSMs
- Edges – mutations of FSMs:
  - Change transition end state
  - Change transition action

Algorithm

graph G = {generate random FSM}
While (True)
    Launch colony of ants on G
    Update pheromone values
    Check stop conditions

- Each ant has a limited number of steps

Next node selection

$P = (1 - P_0)$ – select next node with roulette method
$P = P_0$ – generate mutated FSMs, select best

Pheromone update

- Path quality – maximum fitness value of nodes
- Ants deposit pheromone along sub-path from start to best node
- Pheromone evaporation

Experiments: Inducting FSMs from tests

Input data:

- number N and sets Σ and Δ
- set of test examples T

Each test example consists of an input sequence of events I_j and the corresponding reference output sequence of actions O_j

Set of tests → Induction Algorithm → FSM

Goal: build an FSM which complies with all tests.

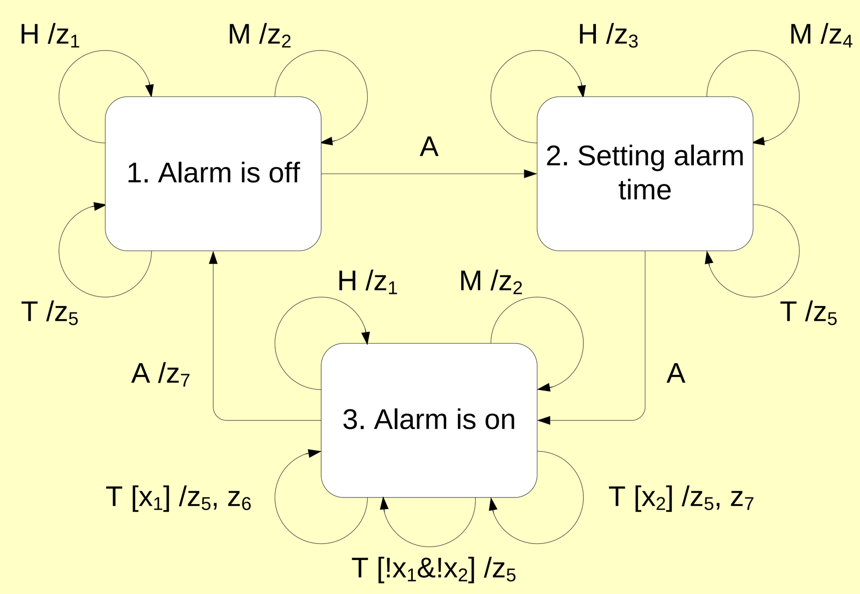
$$f' = \frac{1}{|T|}\sum_{j=1}^{|T|}\left(1 - \frac{ED(O_j, A_j)}{\max\left(len(O_j), len(A_j)\right)}\right) \qquad f = 100 \cdot f' + \frac{1}{100} \cdot (100 - n_{trans})$$

Input sequences → FSM → Output sequences → **Fitness function** ← Reference output sequences

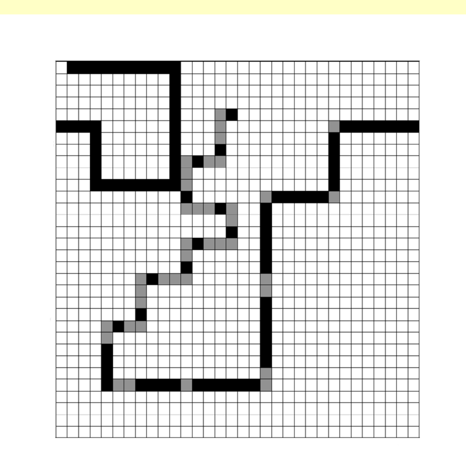
Alarm clock control system induction

- input data: 38 tests for alarm, total length of input sequences 242, total length of answer sequences 195
- comparison with GA and GA+HC
- 1000 runs of each algorithm

Algorithm	Min	Max	Avg	Median
GA	32830	599022	117977	83787
GA+HC	26740	**188509**	**53706**	48106
ACO	**2440**	210971	53944	**46293**



Experiments: Inducting FSMs for John Muir Food Trail Problem



- An "ant" is placed in a two-dimensional toroidal field 32×32
- Some cells contain "food" (apples), a total of 89 pieces
- The ant can "see" if the next cell contains food (events F and !F)
- There are 200 steps, on each step the ant can turn left, turn right or move forward, possibly "eating" a piece of food (actions L, R, M)
- **Goal:** build an FSM controlling the ant so that it can eat all food in 200 steps

Classical FF: $f_1(A) = n + \frac{200 - n_{steps}}{200}$

Modified – variable number of states: $f_2(A) = n + \frac{200 - n_{steps}}{200} + 0.1 \cdot (U - N)$

n – number of eaten apples
n_{steps} – elapsed steps
N – number of states in FSM
U – number of used states

First experiment

Setup:

- Using fitness function f_1
- Searching among FSMs with seven states
- Comparing with GA

Results:

- GA result – 160 and 250 million FF calculations
- ACO result – 143 and 221 million FF calculations

Second experiment

Setup:

- Using fitness function f_2
- Searching among FSMs with 12 states
- 30 runs of ACO algorithm

Results:

- An average of 37 million FF calculations

Publications

- Chivilikhin D., Ulyantsev V., Tsarev F. Test-Based Extended Finite-State Machines Induction with Evolutionary Algorithms and Ant Colony Optimization / Proceedings of the 2012 GECCO Conference Companion on Genetic and Evolutionary Computation. NY.: ACM. 2012, pp. 603 – 606.
- Ulyantsev V., Tsarev F. Extended Finite-State Machine Induction using SAT-Solver / Proceedings of the Tenth International Conference on Machine Learning and Applications, ICMLA 2011, Honolulu, HI, USA, 18-21 December 2011. IEEE Computer Society, 2011. Vol. 2. P. 346–349.

Summary

- Intoduced an ACO-based method of FSM induction
- ACO is either better then GA or works just as well
- ACO does not use problem-specific data, only FSM structure

chivilikhin.daniil@gmail.com
ulyantsev@rain.ifmo.ru