technische universität dortmund

Computational Intelligence

Winter Term 2010/11

Prof. Dr. Günter Rudolph

Lehrstuhl für Algorithm Engineering (LS 11)

Fakultät für Informatik

TU Dortmund

Contents

- Ant algorithms (combinatorial optimization)
- Particle swarm algorithms (optimization in $\mathbb{R}^n$)

metaphor

swarms of bird or fish seeking for food ↓

concepts:

- evaluation of own current situation
- comparison with other conspecific
- imitation of behavior of successful conspecifics

⇒ audio-visual communication

ants or termites seeking for food ↓

concepts:

- communication / coordination by means of „stigmergy“
- reinforcement learning → positive feedback

⇒ olfactoric communication

ant algorithms (ACO: Ant Colony Optimization)

paradigm for design of metaheuristics for combinatorial optimization

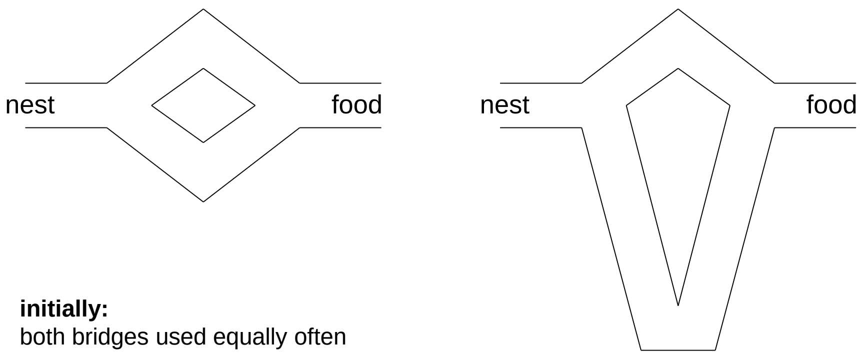
stigmergy = indirect communication through modification of environment

∼ 1991 Colorni / Dorigo / Maniezzo: Ant System (also: 1. ECAL, Paris 1991)
Dorigo (1992): collective behavor of social insects (PhD)

some facts:

- about 2% of all insects are social
- about 50% of all social insects are ants
- total weight of all ants = total weight of all humans
- ants populate earth since 100 millions years
- humans populate earth since 50.000 years

double bridge experiment (Deneubourg et al. 1990, Goss et al. 1989)



initially:
both bridges used equally often

finally:
all ants run over single bridge only!

finally:
all ants use the shorter bridge!

How does it work?

- ants place pheromons on their way
- routing depends on concentration of pheromons

more detailed:

ants that use shorter bridge return faster
⇒ pheromone concentration higher on shorter bridge
⇒ ants choose shorter bridge more frequently than longer bridge
⇒ pheromon concentration on shorter bridge even higher
⇒ even more ants choose shorter bridge
⇒ a.s.f.

positive feedback loop

Ant System (AS) 1991

combinatorial problem:

- components $C = \{ c_1, c_2, ..., c_n \}$
- feasible set $F \subseteq 2^C$
- objective function $f: 2^C \rightarrow \mathbb{R}$

ants = set of concurrent (or parallel) asynchronous agents move through <u>state of problems</u>

↓

partial solutions of problems

⇒ caused by movement of ants the final solution is compiled incrementally

movement: stochastic local decision

(2 parameters)

‘trails‘	‘attractiveness‘
paths	excitement, stimulus

while constructing the solution (if possible), otherwise at the end:

1. evaluation of solutions
2. modification of ‘trail value‘ of components on the path

$\underbrace{\text{1. evaluation of solutions} \quad \text{2. modification}}_{\text{feedback}}$

ant k in state i

- determine all possible continuations of current state i
- choice of continuation according to probability distribution p_{ij}

p_{ij} = q(attractivity, amount of pheromone)

heuristic is based on *a priori* desirability of the move

a posteriori desirability of the move „how rewarding was the move in the past?“

- update of pheromone amount on the paths: as soon as all ants have compiled their solutions
  good solution ↗ increase amount of pheromone, otherwise decrease ↘

Combinatorial Problems (Example TSP)

<u>TSP:</u>

- ant starts in arbitrary city i
- pheromone on edges (i, j): τ_{ij}
- probability to move from i to j: $p_{ij}^{(t)} = \dfrac{\tau_{ij}^{\alpha}\,\eta_{ij}^{\beta}}{\sum\limits_{k \in \mathcal{N}_i(t)} \tau_{ik}^{\alpha}\,\eta_{ik}^{\beta}}$ for $j \in \mathcal{N}_i(t)$
- $\eta_{ij} = 1/d_{ij}$; d_{ij} = distance between city i and j
- $\alpha = 1$ and $\beta \in [2, 5]$ (empirical), $\rho \in (0,1)$ “evaporation rate“
- $\mathcal{N}_i(t)$ = neighborhood of i at time step t (without cities already visited)
- update of pheromone after μ journeys of ants: $\tau_{ij} := \rho\,\tau_{ij} + \sum\limits_{k=1}^{\mu} \Delta\tau_{ij}(k)$
- $\Delta\tau_{ij}(k)$ = 1 / (tour length of ant k), if (i,j) belongs to tour

two additional mechanisms:

1. *trail evaporation*
2. *demon actions* (for centralized actions; not executable in general)

Ant System (AS) is prototype

tested on TSP-Benchmark → not competitive

⇒ but: <u>works in principle</u>!

subsequent: 2 targets

1. increase efficiency (→ competitiveness with *state-of-the-art* method)
2. better explanation of behavior

1995 ANT-Q (Gambardella & Dorigo), simplified: 1996 ACS *ant colony system*

Particle Swarm Optimization (PSO)

↓

abstraction from fish / bird / bee swarm

paradigm for design of metaheuristics for <u>continuous</u> optimization

developed by Russel Eberhard & James Kennedy (~1995)

concepts:

- particle (x, v) consists of position $x \in \mathbb{R}^n$ and “velocity” (i.e. direction) $v \in \mathbb{R}^n$
- PSO maintains multiple potential solutions at one time
- during each iteration, each solution/position is evaluated by an objective function
- particles “fly” or “swarm” through the search space
  to find position of an extremal value returned by the objective function

PSO update of particle (x_i, v_i) at iteration t

1st step:

$$v_i(t+1) = \omega\, v_i(t) + \gamma_1\, R_1\, (x_b^*(t) - x_i(t)) + \gamma_2\, R_2\, (x^*(t) - x_i(t))$$

ω: const.

γ_1: const.

R_1: random variable

$x_b^*(t)$: best solution among all solutions of iteration t ≥ 0

γ_2: const.

R_2: random variable

$x^*(t)$: best solution among all solutions up to iteration t ≥ 0

$$x_b^*(t) = \underset{i=1,\dots,\mu}{\operatorname{argmin}}\{f(x_i(t))\} \qquad x^*(t) = \underset{\tau=0,\dots,t}{\operatorname{argmin}}\{f(x_b^*(\tau))\}$$

PSO update of particle (x_i, v_i) at iteration t

1st step:

$$\underbrace{v_i(t+1)}_{\text{new direction}} = \omega\, \underbrace{v_i(t)}_{\text{old direction}} + \gamma_1\, R_1\, \underbrace{(x_b^*(t) - x_i(t))}_{\text{direction from } x_i(t) \text{ to } x_b^*(t)} + \gamma_2\, R_2\, \underbrace{(x^*(t) - x_i(t))}_{\text{direction from } x_i(t) \text{ to } x^*(t)}$$

ω : inertia factor, often $\in [0.8, 1.2]$

γ_1 : cognitive factor, often $\in [1.7, 2.0]$

γ_2 : social factor, often $\in [1.7, 2.0]$

R_1 : positive r.v., often $r_1 \sim U[0,1]$

R_2 : positive r.v., often $r_2 \sim U[0,1]$

PSO update of particle (x_i, v_i) at iteration t

2nd step:

$$\underbrace{x_i(t+1)}_{\text{new position}} = \underbrace{x_i(t)}_{\text{old position}} + \underbrace{v_i(t+1)}_{\text{new direction}}$$

Note the similarity to the concept of mutative step size control in EAs: first change the step size (direction), then use changed step size (direction) for changing position.