Development of an automatically configurable ant colony optimization framework. State of art.

Aldar Saranov

March 12, 2017

Contents

Abstract

Some animal species show an extreme degree of social organization. Such species (e.g. ants) have pheromone production and detection body parts and therefore seize an ability to communicate between each other in indirect way. This concept has inspired the development of algorithms which are based on social behavior of population called ant colony optimization algorithms (ACO). These algorithms allow to solve NP-hard problems in a very efficient manner. Since these algorithms are considered metaheuristic the development of a ACO framework is the next step of formalizing of this area is to provide tools for resolving general optimization problems. This article gives the brief overview of the current ACO research area state, existing framework description and some tools which can be used for the framework automatic configuration.

1 Introduction (1 page)

Section descriptions. Pheromones. Constructive heuristics. Solution components. Problem models.

2 Combinatorial Optimization Problems and Constructive Heuristics (17 pages)

Definition

Optimization Problem is a tuple (Φ, ω, f), where

- Φ is a search space consisting of all possible assignments of discrete variables x_i, with $i = 1, ..., n$
- ω is a set of constraints for the decision variables
- $f : \Phi \to R$ is an objective function which has to be optimized

TSP and QAP description. Solution components. Feasible solution. Permutation space.

3 The ACO Algorithmic Framework

Algorithm

```
procedure ACO-Metaheuristic
    repeat
        for each ant do
            repeat
                ExtendPartialSolutionProbabilistically()
            until solution is complete
        for each ant $\in$ SelectAntsForLocalSearch() do
            ApplyLocalSearch(ant)
        EvaporatePheromones()
        DepositPheromones()
    until termination criteria met
end
```

3.1 Choice of pheromone trails and heuristic information

C - Solution components.
$\tau_c \in T$ - pheromones of choosing.
$\tau'_c \in T'$ - pheromones of considering order.
π - candidate solution.
$\eta_c \in H$ - heuristic information (constant in time).

3.2 Solution construction

A solution is constructed by an ant.
Probabilistic rules:

- Classic
- Maniezzo
- Dorigo

α, β - choice parameters.
Extensions:

- Lookahead - pick several components at once[94]
- Candidate list - restriction of component choice at each step[33,34]
- Iterated greedy (partial deconstruction)[110]
- With external memory[1]
- Iterated ants[129]
- Cunning ants[128]
- Enhanced ACO[47]

3.3 Global pheromone update

Evaporation
$\tau_{new} = evaporation(\tau_{old}, \rho, S^{eva})$
ρ - evaporation rate
S^{eva} - chosen solutions for evaporation

Deposition
w_k - weight of solution s_k.
$F(S_k)$ - non-decreasing solution quality scaling function.

Update selection

1. Ant system (update all)
2. Single update selections:
   (a) iteration-based update
   (b) global-based update
   (c) restart-based update

3.4 Pheromone update schedule

3.5 Initialization of pheromones

3.6 Pheromone reinitialization

3.7 Local pheromone update

3.8 Pheromone limits

3.9 Local search

3.10 ACO algorithms as instantiations of the ACO Metaheuristc

4 ACOTSP/ACOQAP: A unified framework of ACO algorithms for the TSP and QAP

4.1 Finding a better ACO configuration for the TSP

4.2 Finding a better ACO configuration for the QAP

5 Applications of ACO to other problem types

5.1 Continuous Optimization Problems

5.2 Multi-objective problems

5.3 Dynamic problems

5.4 Stochastic problems

6 ACO in combination with other methods

6.1 ACO and tree search methods

6.2 ACO and exact methods

6.3 ACO and surrogate models

6.4 Parameter adaptation

7 Existing ACO framework (5 pages)

8 IRACE automatic configuration (3 pages)

9 Conclusions