New Challenges in Scheduling Theory
Aussois

April 4, 2014

Optimizing Supply Process in Charitable Organizations by Genetic Algorithm

Małgorzata Sterna
Mateusz Cicheński, Mateusz Jarus, Michał Miszkiewicz, Jarosław Szymczak

Institute of Computing Science
Poznań University of Technology
Poznań - Poland
Scope of the Talk

- Problem definition
- MP Formulation
- Heuristic Algorithm
- Genetic Algorithm
- Computational Experiments
- Conclusions
Problem Definition

- web service for charitable organizations
- gathering charitable organizations and donors
- allowing submitting requests and registering offers for various products
- managing database for registered users
- supporting supply process with optimization algorithms
Problem Definition

Charitable organization (customer)

Demand for \( m \) types of products in amount of \( d_j \) units

Products are offered by \( n \) depots (donors, warehouses, shops)

In amount of \( a_{ij} \) units at price \( c_{ij} \)

Unit transportation cost \( T \)

Distance \( t_{ir} \)

Amount of ordered units \( x_{ij} \) of particular products

Position of a depot in the route \( y_{ik} \)
MP Formulation

\[
\begin{align*}
\min \quad & \sum_{i=1}^{n} \sum_{j=1}^{m} x_{ij} c_{ij} + T \left( \sum_{j=1}^{n} y_{i1} t_{0i} + \sum_{i=1}^{n} \sum_{r=1}^{n} \sum_{k=1}^{n-1} \max\{0, y_{ik} + y_{r,k+1} - 1\} t_{ir} + \sum_{i=1}^{n} \sum_{r=1}^{n} (y_{ik} (1 - \sum_{r=1}^{n} y_{r,k+1})) \right) \\
\text{under constraints} \\
& \sum_{i=1}^{n} x_{ij} = d_{j} \quad j = 1 \ldots m \\
& x_{ij} \leq a_{ij} \quad i = 1 \ldots n, j = 1 \ldots m \\
& x_{ij} \geq 0 \text{ and integer} \quad i = 1 \ldots n, j = 1 \ldots m
\end{align*}
\]

order delivery subproblem

\[
\begin{align*}
& \sum_{k=1}^{n} y_{ik} = \min\{1, \sum_{j=1}^{m} x_{ij}\} \quad i = 1 \ldots n \\
& \sum_{k=1}^{n} y_{ik} \leq 1 \quad i = 1 \ldots n \\
& \sum_{i=1}^{n} y_{ik} \leq 1 \quad k = 1 \ldots n \\
& \sum_{i=1}^{n} y_{i,k+1} \leq \sum_{i=1}^{n} y_{ik} \quad k = 1 \ldots n - 1 \\
& y_{ik} \in \{0, 1\} \quad i = 1 \ldots n, k = 1 \ldots n + 1
\end{align*}
\]

order completion subproblem

\[
\begin{align*}
& \sum_{i=1}^{n} x_{ij} = d_{j} \quad j = 1 \ldots m \\
& x_{ij} \leq a_{ij} \quad i = 1 \ldots n, j = 1 \ldots m \\
& x_{ij} \geq 0 \text{ and integer} \quad i = 1 \ldots n, j = 1 \ldots m
\end{align*}
\]
Problem Formulation

- selecting depots offering demanded products at the lowest prices
  - order completion
  - „easy” problem – greedy solution is optimal
- determining the shortest route to pick up products from these depots
  - order delivery
  - „hard” problem – reduces to the shortest Hamiltonian cycle
- order completion and delivery is strongly NP-hard as a variant of Travelling Purchaser Problem (Ramesh 1981)
Heuristic Algorithm

- two-phase heuristic approach
  - selecting depots
  - constructing a tour
Heuristic Algorithm

- selecting depots - choosing depots until demand is satisfied
- ordering depots according to
  - product cost (greedy heuristic)
  - weighted priorities (priority heuristic), based on:
    - total distance to other locations
    - total cost of demanded products available at a depot
    - total cost of all demanded products
- various priority weights result in various list heuristics and various sets of selected depots
Heuristic Algorithm

- constructing a tour from selected depots
- Minimum Spanning Tree Heuristic (Hedl & Karp 1970)
  - constructing minimum spanning tree by Kruskal Algorithm
  - traversing the tree according to Depth First Search Strategy
  - converting DFS sequence to the Hamiltonian cycle

- MSTH is 2-approximation algorithm
Bounds

- heuristic solution determines upper bound (UB)
- lower bound

\[ LB = \sum_{j=1}^{m} \sum_{i=1}^{\tilde{n}} x_{\pi_i,j} c_{\pi_i,j} + T\left( \min_{i=1 \ldots n} \{t_{0i}\} + \min_{i=1 \ldots n} \{t_{i0}\} \right) \]

- reference bounds

\[ RB = \sum_{j=1}^{m} \sum_{i=1}^{\tilde{n}} x_{\pi_i,j} c_{\pi_i,j} + T\left( \min_{i=1 \ldots n} \{t_{0i}\} + \sum_{k=1}^{\tilde{n}-1} t_{[k]} + \min_{i=1 \ldots n} \{t_{i0}\} \right) \]

\[ t_{[k]} \] - k’th distance between depots
Genetic Algorithm

- solution is a sequence of assignments:
  - number of product units ordered from a depot
  - one product can be taken from more than one depot
  (to determine a complete solution a tour is constructed by MST heuristic)
- initial population
  - heuristic solutions corresponding to various priorities weights
  - random solutions
- new population replaces the previous one
Genetic Algorithm – Operators

- one-point crossover and two-point crossover
  - exchanging parts of assignments product-depot
  - infeasible offspring repair procedure:
    - exceeding the product availability at a certain depot
    - taking products from another depot in offspring
    - taking products from a depot in parental solution

- mutation
  - replacing a given number of assignments (product-depot) with random assignment
Genetic Algorithm - Selection

- selecting mating population according to crossover rate
  - roulette selection
    (according to criterion values)
  - ranking selection
    (according to the position in ranking)
  - tournament selection
    (the best solution from randomly chosen groups)

- mutation according to mutation rate
Genetic Algorithm – Termination Condition

- the maximum number of generations

- the maximum number of generations without improvement

- exceeding the satisfying ratio of the criterion value improvement
Computational Experiments

- random instances reflecting real world scenarios
- depots located in 48 Polish cities
- the charitable organization located in Poznań
- distances correspond to road distances (Bing Maps)
- unit transportation cost determined by government regulations
Computational Experiments

- a single order contains of 5 to 200 product types
  (from 1 to 10 units of each type)

- prices of products in depots are determined based on
  - basic price
  - modified by discount factor
    generated with normal distribution [-50%, +50%]
Computational Experiments

- availability of products in depots are generated according to 3 scenarios:
  - „round robin” distribution
    - demanded units of a product are placed in depots one by one
    - all depots have to be visited (order delivery is crucial)
  - „clone” distribution
    - all demanded products are available in all depots
    - prices of products are crucial (order completion is crucial)
  - „even” distribution
    - availability in all depots is increased by one unit at the time until demand is exceeded
    - prices and distances are crucial
Time per generation

![Graph showing the relationship between number of product types and time per generation for different strategies: clone strategy, even strategy, and round robin strategy.](image)

- **Clone strategy**
- **Even strategy**
- **Round robin strategy**

The graph illustrates the increase in time required for each strategy as the number of product types grows.
Transportation cost vs. Products cost

Number of product types

Transport cost ratio

- clone strategy
- even strategy
- round robin strategy
Criterion value vs. lower bound

- "Round Robin" distribution
- "Clone" distribution
- "Even" distribution
Conclusions

- web service devoted for charitable institutions
  - data base of offered/donated products and submitted requests
  - optimization tool supporting realizing orders by minimizing products cost and transportation cost
- solving a variant of travelling purchaser problem by
  - heuristic list algorithm
  - genetic algorithm
- validation algorithms in computational experiments
  - solution quality close to the lower bound
  - short computational time acceptable by web service users