Design and Optimization of Supply Chain Management

As. Prof. Yannis Marinakis and Dr. Magdalene Marinaki, Technical University of Crete,
School of Production Engineering and Management, magda@dssl.tuc.gr

Chania, 11/5/2017

Supply Chain Management I

A complete logistics system covers the entire process of shipping raw materials and input requirements from suppliers to plants, the conversion of the inputs into products at certain plants, the shipping of the products to various warehouses or depots, and the eventual delivery of these products to the final customers. The distribution activities of a firm comprise all shipping and storage of goods downstream from the plants.

We classify the decisions for supply chain management into two broad categories — strategic and operational. Strategic decisions are made typically over a longer time horizon. On the other hand, operational decisions are short term, and focus on activities over a day–to–day basis. There are four major decision areas in supply chain management, and there are both strategic and operational elements in each of these areas:

Supply Chain Management II

1. Location.
2. Production.
3. Inventory.
4. Transportation — Distribution.

Transport decisions can involve mode selection, shipment size, and routing and scheduling. These decisions are influenced by the proximity of warehouses to customers and plants, which, in turn, influence warehouse location. Inventory levels also respond to transport decisions through shipment size.

Supply Chain Management III

- Transportation often represents the most important single element in logistics costs for most firms.
- Transportation is a key decision area within the logistics mix. Except for the cost of purchased goods, transportation absorbs, on the average, a higher percentage of logistics costs than any other relevant activity.
- Because transportation costs typically range between one third and two thirds of total logistics costs, improving efficiency through the maximum utilization of transportation equipment and personnel is of a major interest.
- The time that goods are transit reflects on the number of shipments that can be made with a vehicle within a given period of time and on the total transportation costs for all shipments.
- To reduce transportation costs and also to improve customer services finding the best — in terms of time or distance minimization — routes that a vehicle should follow through a network of roads, rail and other shipping lines is frequently an important decision problem.

Capacitated Vehicle Routing Problem I

- The **Vehicle Routing Problem (VRP)** or the **Capacitated Vehicle Routing Problem (CVRP)** is often described as the problem in which vehicles based on a central depot are required to visit geographically dispersed customers in order to fulfill known customer demands.
- The problem is to construct a low cost, feasible set of routes - one for each vehicle.
- A route is a sequence of locations that a vehicle must visit along with the indication of the service it provides. The vehicle must start and finish its tour at the depot.
- The objective function is to minimized the total distance traveled

Capacitated Vehicle Routing Problem I

- Let $G = (V, E)$ be a graph where V is the vertex set and E is the arc set.
- The customers are indexed $i = 2, \cdots, n$ and $i = 1$ refers to the depot. The vehicles are indexed $k = 1, \cdots, K$. A customer i has a demand of q_i.
- The capacity of vehicle k is Q_k. If the vehicles are homogeneous, the capacity for all vehicles is equal and denoted by Q.
- A demand q_i and a service time st_i are associated with each customer node i.
- The travel cost and the travel time between customers i and j is c_{ij} and t_{ij}, respectively.

Capacitated Vehicle Routing Problem I

The formulations of the Capacitated Vehicle Routing Problem are divided in three different categories

- *vehicle flow models,*
- commodity flow models and
- *set partitioning models*

.

Capacitated Vehicle Routing Problem I

- The **vehicle flow models** use integer variables associated with each arc. They can, easily, be used when the cost of the solution can be expressed as the sum of the costs associated with each arc.
- In the **commodity flow models**, additional integer variables are associated with the arcs and they express the flow of the commodities along the paths traveled by the vehicles.
- Finally, in the **set partitioning formulation**, a collection of circuits with minimum cost is determined which serves each customer once and, possibly, satisfies additional constraints. One of the main drawbacks of the last models is the huge number of variables.

Capacitated Vehicle Routing Problem I
Formulation 1 by Fisher & Jaikumar

Let

$$x_{ijk} = \begin{cases} 1, & \text{if vehicle } k \text{ visits customer } j \text{ immediately after} \\ & \text{customer } i \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

and

$$y_{ik} = \begin{cases} 1, & \text{if customer } i \text{ is visited by vehicle } k \\ 0, & \text{otherwise.} \end{cases} \quad (2)$$

Capacitated Vehicle Routing Problem I
Formulation 1 by Fisher & Jaikumar

The Vehicle Routing Problem can then be stated as:

$$\min \sum_{i,j} c_{ij} \sum_k x_{ijk} \tag{3}$$

s.t.

$$\sum_k y_{ik} = \begin{cases} 1, & i = 2, \cdots, n, \\ K, & i = 1 \end{cases} \tag{4}$$

$$\sum_i q_i y_{ik} \leq Q_k \qquad k = 1, \cdots, K \tag{5}$$

$$\sum_j x_{ijk} = \sum_j x_{jik} = y_{ik} \qquad i = 1, \cdots, n \tag{6}$$

$$k = 1, \cdots, K$$

$$\sum_{i,j \in S} x_{ijk} \leq \mid S \mid -1, \qquad \forall S \subseteq \{2, \cdots, n\}, \tag{7}$$

$$k = 1, \cdots, K$$

$$y_{ik} \in \{0, 1\}, \qquad i = 1, \cdots, n, \tag{8}$$

$$k = 1, \cdots, K$$

$$x_{ijk} \in \{0, 1\}, \qquad i, j = 1, \cdots, n, \tag{9}$$

$$k = 1, \cdots, K.$$

Capacitated Vehicle Routing Problem II

Formulation 1 by Fisher & Jaikumar

- Constraints (4) ensure that each customer is assigned to one vehicle, except of the depot which is visited by all vehicles,
- Constraints (5) are the capacity constraints of the vehicles,
- Constraints (6) ensure that a vehicle which visits a customer leaves immediately from the customer and
- Constraints (7) are the subtour elimination constraints for the TSP.

Capacitated Vehicle Routing Problem I
Formulation 2 by Golden

$$\min \sum_{i=1}^{n} \sum_{j=1}^{n} \sum_{k=1}^{K} c_{ij} x_{ij}^{k} \tag{10}$$

s.t.

$$\sum_{i=1}^{n} \sum_{k=1}^{K} x_{ij}^{k} = 1, \qquad j = 2, \cdots, n \tag{11}$$

$$\sum_{j=1}^{n} \sum_{k=1}^{K} x_{ij}^{k} = 1, \qquad i = 2, \cdots, n \tag{12}$$

$$\sum_{i=1}^{n} x_{ii_1}^{k} - \sum_{j=1}^{n} x_{i_1 j}^{k} = 0, \qquad k = 1, \cdots, K \tag{13}$$

$$i_1 = 1, \cdots, n$$

$$\sum_{i=1}^{n} q_i \sum_{j=1}^{n} x_{ij}^{k} \le Q_k, \qquad k = 1, \cdots, K \tag{14}$$

Capacitated Vehicle Routing Problem I

Formulation 2 by Golden

$$\sum_{i=1}^{n} t_i^k \sum_{j=1}^{n} x_{ij}^k + \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij}^k x_{ij}^k \le Tm_k, \qquad k = 1, \cdots, K \tag{15}$$

$$\sum_{j=2}^{n} x_{1j}^k \le 1, \qquad k = 1, \cdots, K \tag{16}$$

$$\sum_{i=2}^{n} x_{i1}^k \le 1, \qquad k = 1, \cdots, K \tag{17}$$

$$X \in S \tag{18}$$

$$x_{ij}^k = 0 \text{ or } 1, \qquad \text{for all } i, j, k \tag{19}$$

Capacitated Vehicle Routing Problem I
Formulation 2 by Golden

In the previous equations:

- n is the number of nodes,
- K the number of vehicles,
- Q_k the capacity of vehicle k,
- Tm_k the maximum time allowed for a route of vehicle k,
- q_i the demand of node i ($q_1 = 0$),
- t_i^k the time required for vehicle k to deliver or to collect at node i ($t_1^k = 0$),
- t_{ij}^k the travel time for vehicle k from node i to node $j (t_{ii}^k = \infty)$,
- c_{ij}^k the cost of travel from node i to node j,
- $x_{ij}^k = 1$ if arc $i - j$ is traversed by vehicle k and 0 otherwise.

Capacitated Vehicle Routing Problem I
Formulation 2 by Golden

- Objective function (10) states that the total distance is to be minimized.
- Equations (11) and (12) ensure that each demand node is served by exactly one vehicle.
- Route continuity is represented by (13), i.e. if a vehicle enters in a demand node, it must exit from that node.
- Equations (14) are the vehicle capacity constraints and
- (15) are the total elapsed route time constraints.
- Equations (16) and (17) guarantee that vehicle availability is not exceeded.

Basic Variants of the Vehicle Routing Problem I

Open Vehicle Routing Problem

- The **Open Vehicle Routing Problem (OVRP)** is the variant of the classic vehicle routing problem where the vehicles do not return to the depot after the service of the customers.
- The main real life application of the Open Vehicle Routing Problem concerns the case where either the company does not have vehicles at all or the vehicles owned by the company are not enough in order to use them for the distribution of the products to the customers and, thus, a number of vehicles has to be hired by the company in order to realize the distribution of the products.

Basic Variants of the Vehicle Routing Problem I

Open Vehicle Routing Problem

- The purpose is to construct a low cost, feasible set of routes - one for each vehicle.
- A route is a sequence of locations that a vehicle must visit along with the indication of the service it provides.
- Each vehicle starts at the depot but it doesn't return to the depot.
- The total traveling cost of each route can not exceed the restriction L

Basic Variants of the Vehicle Routing Problem I
Vehicle Routing Problem with Time Windows

- The **Vehicle Routing Problem with Time Windows (VRPTW)** specifies that customers must be serviced within some time window.
- Vehicles may, also, be associated with time windows within which they are allowed to operate.
- Usually, in most publications concerning the Vehicle Routing Problem with Time Windows, the **objective is to design least cost routes for a fleet of identical capacitated vehicles to service geographically scattered customers within pre-specified time windows**.

Basic Variants of the Vehicle Routing Problem I

Vehicle Routing Problem with Time Windows

- Each customer must be serviced once by a vehicle and the total demands of the customers serviced by the vehicle must not exceed the capacity of the vehicle.
- Moreover, each customer must be serviced within a specified time window.
- If a vehicle arrives at a customer earlier than the lower bound of the customer's time window, the vehicle must wait until the service is possible.
- The depot has, also, a time window and all the vehicles must return by the closing time of the depot.
- The objective is to minimize, initially, the number of tours or routes and, then, for the same number of routes to minimize the total traveled distance.

Basic Variants of the Vehicle Routing Problem I

MultiDepot Vehicle Routing Problem

- The **MultiDepot Vehicle Routing Problem (MDVRP)** is the variant of Vehicle Routing Problem where more than one depots are used for the customers' service.
- The MDVRP is one of the most challenging variants of the VRP where there is a very large number of published papers.
- In this problem, there is a possibility for each customer to be clustered and served from only one depot or the customers have to be served from any of the depots using the available fleet of vehicles.
- In the classic formulation of the problem, each vehicle route starts and ends at the same depot, each customer is serviced exactly once by a vehicle, the total demand of each route does not exceed the vehicle capacity and the total cost of the distribution is minimized.

Pickup and Delivery I

- The pickup and deliveries problem is a field of the Vehicle Routing Problem that it has gained much attention in the last years.
- The problem can be divided in **pickup and delivery of goods and of people**.
- Also, other different formulations are the formulations as a problem with one origin and destination, as a problem with different origins and destinations, as a problem with one single vehicle or as a problem with more than one vehicles, as a problem where all the deliveries are made and, then, all the pickups are made or mixed and so on.

Pickup and Delivery II

- The most known formulation of the pickup and delivery problem is the one denoted as 1-M-1 Pickup and delivery problem (One-to-many-to-one) in which deliveries and pickups concern two distinct sets of commodities: some are shipped from the depot to the customers and others are picked up at the customers and delivered to the depot

Pickup and Delivery I
Vehicle Routing Problem with Simultaneous Pickup and Delivery

- In the **Vehicle Routing Problem with Simultaneous Pickup and Delivery (VRPSPD)** there is a set of customers where they do not only require the delivery of products but, also, a simultaneous pick up of products from them is needed.
- Thus, in this problem, each customer i has two different quantities, the quantity d_i that represents the demand of homogeneous commodities to be delivered and the quantity p_i that represents the picked up quantity of customer i.
- It should be defined the node that plays the role of the origin (i_O) of the vehicles and the node that plays the role of the destination (i_D) of the vehicles.

Pickup and Delivery II

Vehicle Routing Problem with Simultaneous Pickup and Delivery

- It is assumed that the delivery is performed before the pickup and that the vehicle load should never be negative or larger than the vehicle capacity. Usually, the origin and destination nodes are the same (the depot) and all the other constraints are the same with the constraints of the CVRP.
- The problem is to construct K simple routes with minimum cost where each route visits the depot node, each vehicle is associated with only one route, each customer node is visited by exactly one vehicle, the current load of the vehicle along the route must be nonnegative and should never exceed the vehicle capacity Q.

Pickup and Delivery I

Vehicle Routing Problem with Backhauls

- In this problem, the vehicles are not only required to deliver goods to (linehaul) customers but also to pick up goods at (backhaul) customer locations.
- **Linehaul customers** require the delivery of a given quantity of product from the depot, whereas a given quantity of product must be picked up from **Backhaul customers** and transported to the depot.
- The customers are partitioned into two subsets, the n_1 Linehaul Customers, each requiring a quantity of product to be delivered.

Pickup and Delivery I

Vehicle Routing Problem with Backhauls

- The second subset contains the n_2 Backhaul Customers, where a given quantity of inbound product must be picked up.
- The objective of the problem is to find a set of vehicle routes that serves all the linehaul and backhaul customers, where each route visits the depot node, each vehicle is associated with only one route, each customer node is visited by exactly one vehicle, for each route the total loads associated with linehaul and backhaul customers do not exceed, separately, the vehicle capacity, the distance of each route does not exceed the maximum distance that the associated vehicle can travel, on each route the backhaul customers, if any, are served after all linehauls customers, and, the total distance traveled by the fleet is minimized.

Pickup and Delivery I
Dial - A - Ride Vehicle Routing Problem

- In this problem, a vehicle picks up customers at their places of origin and takes them to their destination, without exceeding vehicle capacity, and respecting time windows constraints at both the origins and the destinations.
- The objective is to find the itinerary which minimizes the total distance traveled.
- This is one of the classic pickup and delivery problems focusing on people and not on goods.

Other Variants of the Vehicle Routing Problem I

Split Delivery Vehicle Routing Problem

- In the classic Vehicle Routing Problem, a customer is not allowed to be serviced by more than one vehicle but in the **Split Delivery Vehicle Routing Problem** (**SDVRP**) a customer is allowed to be serviced by several vehicles if this reduces the overall cost.
- Thus, the latter addresses the situation where a fleet of homogeneous vehicles must serve a set of customers, the demand of which can take any integer number, possibly greater than the vehicle capacity.
- A customer may need to be served more than once (multiple customer visits), contrary to what is usually assumed in Vehicle Routing Problems.
- Each time a vehicle visits a customer it collects an integer quantity.
- No constraint on the number of available vehicles is considered. Each vehicle starts from and returns to the depot at the end of each tour.

Other Variants of the Vehicle Routing Problem II
Split Delivery Vehicle Routing Problem

- The objective is to minimize the total distance traveled by the vehicles to serve all the customers.

Other Variants of the Vehicle Routing Problem I

Heterogeneous Fleet Vehicle Routing Problem

- In the **Heterogeneous Fleet Vehicle Routing Problem (HFVRP)** there is a set of vehicles each having a possibly different capacity and cost. A set of n customers is given, each one having a demand q_i. There is, also, a fleet of P different vehicle types.
- Each vehicle type $p = 1, \cdots, P$ has a capacity Q_p, a fixed cost FC_p and a traveling cost c_{ij}^p.
- Many specific variants of the problem were studied in the literature.
- The main differences between them are: the vehicle fleet may be either limited or unlimited, the fixed cost of the vehicles may be either considered or ignored and the routing costs on arcs may be vehicle-dependent or vehicle-independent.

Other Variants of the Vehicle Routing Problem I

Multi-Trip Vehicle Routing Problem

The **Multi-Trip Vehicle Routing Problem** has a number of additional constraints of the CVRP. The most important of them is that a vehicle can make more than one trip.

Other Variants of the Vehicle Routing Problem I

Periodic Vehicle Routing Problem

- One of the most interesting generalization of the capacitated vehicle routing problem is the **Periodic Vehicle Routing Problem (PVRP)**.
- In this problem, vehicle routes must be constructed over multiple days where during each day within the planning period, a fleet of capacitated vehicles travels along routes that begin and end at a single depot.
- The objective of the PVRP is to find a set of tours for each vehicle that minimizes total travel cost while satisfying vehicle capacity and visit requirements.
- The PVRP require three types of decisions: initially, the selection of visiting patterns for each customer, afterwards, the assignment of the chosen day-customer combinations to tours, and, finally, the routing of vehicles for each day of the planning horizon.

Other Variants of the Vehicle Routing Problem I

Green Vehicle Routing Problem

- In recent years, a significant growth of publications in **Green Vehicle Routing Problems (GVRP)** has been realized.
- In these problems, the main target is the reduction of fuel and energy consumption or the reduction of the pollution caused by the CO_2 emissions.
- Thus, a number of different publications with different objectives and different constraints have been made. In these publications, the authors did not use a transformation of the CVRP into GVRP but they used anyone of the variants of the VRP depending on the problem that they have to solve.

Other Variants of the Vehicle Routing Problem I

Vehicle Routing Problem with Profits

- In the Vehicle Routing Problems with profits (VRPPs) the set of customers to be served is not given in advance.
- Two different decisions have to be taken:
  - which customers to serve, and
  - how to cluster the customers to be served in different routes (if more than one) and order the visits in each route.
- The basic characteristic is that a profit is associated with each customer that makes such a customer more or less attractive.
- Thus, any route or set of routes, starting and ending at a given depot, can be measured both in terms of cost and in terms of profit.
- There is a large number of applications that corresponds to a Vehicle Routing Problem with Profits.

Other Variants of the Vehicle Routing Problem I
Team Orienteering Problem

- The **Team Orienteering Problem (TOP)** is a variant of the vehicle routing problem with profits or pricing. In the team orienteering problem (TOP) a set of locations is given, each with a score.
- The goal is to determine a fixed number of routes, limited in length, that visit some locations and maximize the sum of the collected scores.
- The **objective of the TOP** is to construct a certain number of paths starting at an origin and ending at a destination that maximize the total profit without violating predefined limits.

Vehicle Routing Problem with Stochastic Demands

- The **Vehicle Routing Problem with Stochastic Demands (VRPSDs)** is a well known NP-hard problem in which a vehicle with finite capacity leaves from the depot with full load and has to serve a set of customers whose demands are known only when the vehicle arrives to them (this is the main difference of the VRPSDs from the Capacitated VRP where all the customers' demands are known beforehand).
- A route begins from the depot and visits each customer exactly once and returns to the depot. This is called an a priori tour and it can be seen as a template for the visiting sequence of all customers.
- In a given instance, the customers should be visited based on the sequence of the a priori tour but the final route includes, also, returns to the depot when the vehicle needs replenishment. The nodes from which the vehicle returns to the depot are stochastic points.

Stochastic and Dynamic Vehicle Routing Problems I

Vehicle Routing Problem with Stochastic Demands and Customers

- In the **Vehicle Routing Problem with Stochastic Demands and Customers** the customers' demands are stochastic variables and we are, also, not aware before the route if every customer is present or not in the route, i.e. if he has a demand in this specific route.
- This problem is more difficult than the previous problem. The route is constructed as in previous case where an a priori route with all customers present is constructed.
- Then, when the route begins, we check if a customer is present in the route, and, thus, if its demand is different than zero and when the vehicle arrives to the customer how much is the demand of the customer.

- There are two ways to deal with the route failure in this problem as in the case of the Vehicle Routing Problem with Stochastic Demands, either by using the preventive restocking strategy and return to the depot before the route failure occurs or to return immediately to the depot when the route failure occurs.

Vehicle Routing Problem with Stochastic Travel Times

- The **Vehicle Routing Problem with Stochastic Travel Times (VRPSTT)** is one of the most interesting and challenging variants of the Vehicle Routing Problem.
- In this variant, the travel time between every node pair is a random variable related to traffic jam, road maintenance or weather conditions.
- This problem belongs to the category of the **Stochastic Vehicle Routing Problems**, where the most known of them are the Vehicle Routing Problem with Stochastic Demands, the Vehicle Routing Problem with Stochastic Customers, the Vehicle Routing Problem with Stochastic Demands and Customers and the Vehicle Routing Problem with Stochastic Service Times.

Stochastic and Dynamic Vehicle Routing Problems II

Vehicle Routing Problem with Stochastic Travel Times

- However, the main difference of the Vehicle Routing Problem with Stochastic Travel Times from all the other problems is that the conditions of the customers (demands, service time or existence of the current customer in the route) are deterministic and the stochasticity appears in the arcs (road) of the network due to unexpected conditions.

Stochastic and Dynamic Vehicle Routing Problems I
Vehicle Routing Problem with Stochastic Travel Times

- The VRPSTT is a **stochastic vehicle routing problem (SVRP)** that consists of planning routes to service a number of customers in the presence of stochastic travel and service times.
- The arcs have nonnegative stochastic travel times and the nodes have nonnegative stochastic service times with known distributions.
- Both stochastic variables have a discrete probability distribution.
- The real travel time of an arc is known only when the vehicle arrives to the customer having used this arc and the real service time of each node is known only when the vehicle leaves from the node. Vehicles are capacitated and routes for each vehicle begin and end at the depot.
- The demand of the customers is not a stochastic parameter. A priori routes are calculated. The vehicles must follow their a priori routes; no route reoptimizations are permitted.

Stochastic and Dynamic Vehicle Routing Problems I

Vehicle Routing Problem with Stochastic Travel Times

- The time at which the vehicles return to the depot, after all nodes have been serviced, is called the completion time. More analytically, the completion time is the summation of the travel times between the selected arcs for all vehicles and of the service times of the nodes.
- The objective function of the problem is the minimization of the expected completion time.
- The operation is completed when all the vehicles have returned to the depot after finishing their routes.
- The constraints are the classic constraints of a vehicle routing problem, meaning a vehicle that arrives to a node should leave from this node, each node is serviced exactly once, each vehicle begins and ends its route in the depot, the capacity of the vehicles should not be violated, the subtour elimination constraint exist.

Stochastic and Dynamic Vehicle Routing Problems I

Dynamic Vehicle Routing Problem

- A number of problems with uncertainty in one or more basic parameters have been studied in the framework of the Vehicle Routing Problem.
- In dynamic problems, part or all of the input is unknown and is revealed dynamically during the design or execution of the routes. These problems are denoted as **Dynamic Vehicle Routing Problems (DVRPs)**.
- For these problems, vehicle routes are redefined in an ongoing fashion, requiring technological support for real-time communication between the vehicles and the decision maker.

In the dynamic vehicle routing problems new data arrive or are revealed in real time. Thus, the decisions must be made in a changing environment and the main goal is the decision maker to react to the new events as well as to anticipate future events, particularly if exploitable stochastic information is available or can be derived from past data.

Location Routing Problem I

In the **most location models**, it is assumed that the customers are served directly from the facilities being located.

Each customer is served on his or her own route.

In many cases, however, customers are not served individually from the facilities. Rather, customers are consolidated into routes which may contain many customers.

One of the reasons for the added difficulty in solving these problems is that there are far more decisions that need to be made by the model.

Location Routing Problem I

- how many facilities to locate,
- where the facilities should be,
- which customers to assign to which warehouses,
- which customers to assign to which routes,
- in what order customers should be served on each route.

Location Routing Problem I

In the Location Routing Problem a number of facilities are located among candidate sites and delivery routes are established to a set of users in such a way that the total system cost is minimized.

As Perl and Daskin pointed out, location-routing problems involve three inter-related, fundamental decisions: where to locate the facilities, how to allocate customers to facilities, and how to route the vehicles to serve customers.

Location Routing Problem I

The difference of the **Location Routing Problem (LRP)** from the classic vehicle routing problem is that not only routing must be designed but also, the optimal depot location must be simultaneously determined.

The main difference between the location routing problem (LRP) and the classical location - allocation problem is that, once the facility is located, the former requires a visitation of customers through tours while the later assumes that the customer will be visited from the vehicle directly and, then, it will return to the facility without serving any other customer.

In general terms, the **combined location - routing model** solves the joint problem of determining the optimal number, capacity and location of facilities serving more than one customer and finding the optimal set of vehicle routes.

Location Routing Problem II

In the **location routing problem** the distribution cost is decreased due to the assignment of the customers to vehicles while the main objective is the design of the appropriate routes of the vehicles.

Heuristic Algorithms I

Constructive methods *Nearest Neighbor Procedure* In this heuristic, the salesman starts at some city and then visits the city nearest to the starting one. From there, he visits the nearest unvisited city, until all cities are visited, and then returns to the starting city.

Step 1. Start with any node as the beginning of a path.

Step 2. Find a node, not already on the path, which is closest to the last added node. Add it to the path.

Step 3. Repeat Step 2 until all nodes belong to the path. Then, join the first and the last nodes of the path.

For symmetric, complete graphs, the worst case behavior of the algorithm is

$$\frac{\text{length of nearest neighbor tour}}{\text{length of optimal tour}} \leq \frac{1}{2}\lceil \log_2(n) \rceil + \frac{1}{2},$$

where $\lceil x \rceil$ is the smallest integer $\geq x$, and n is the number of nodes in the network. The nearest neighbor algorithm is of $O(n^2)$ time complexity.

Heuristic Algorithms II

Insertion Procedures The insertion procedures takes a subtour of k nodes and attempts to determine which node (not already in the subtour) should join the subtour next (the *selection step*) and then determines where in the subtour it should be inserted (the *insertion step*). The most known of these algorithms is the *nearest insertion algorithm*:

Step 1 Start with a subgraph consisting of only one node, say i.

Step 2 Find node k such that c_{ik} is minimal and form the subtour $i - k - i$.

Step 3 (Selection) Given a subtour, find node k, not already in the subtour, closest to any subtour node.

Step 4 (Insertion) Find the arc (i, j) in the subtour which minimizes $c_{ik} + c_{kj} - c_{ij}$. Insert k between i and j.

Step 5 Go to Step 3 unless a Hamiltonian cycle has been formed.

The worst case behavior of the algorithm is

$$\frac{\text{length of nearest insertion tour}}{\text{length of optimal tour}} \leq 2,$$

and its time complexity $O(n^2)$.

Heuristic Algorithms III

Clarke and Wright algorithm

This savings algorithm is an exchange procedure, that was originally developed for the VRP. It can, however, be applied to the TSP as well:

Step 1. Select any node as the central depot and denote it as node 1.

Step 2. Compute the savings $s_{ij} = c_{1j} + c_{1i} - c_{ij}$ for $i, j = 2, 3, \cdots, n$.

Step 3. Order the savings from largest to smallest.

Step 4. Starting at the top of the savings list and moving downwards, form larger subtours by linking appropriate nodes i and j. Repeat until a tour is formed.

Next the Clarke and Wright algorithm for the solution of the VRP is given:

Step 1. Calculate the savings $s_{ij} = c_{1j} + c_{1i} - c_{ij}$ for all pairs of customers i and j. Note that s_{ij} is the saving in cost that would result if the link (i, j) is made to produce the route $(1, i, j, 1)$ instead of supplying i and j on two routes $(1, i, 1)$ and $(1, j, 1)$.

Step 2. Order the savings in descending order.

Step 3. Starting at the top of the list do the following.

Parallel version

Heuristic Algorithms IV

Step 4. If a given link results in a feasible route according to the constraints of the VRP, then append this link to the solution, if not reject the link.

Step 5. Try the next link in the list and repeat Step 4 until no more links can be chosen.

Sequential version

Step 4. Find the first feasible link in the list which can be used to extend one of the two ends of the currently constructed route.

Step 5. If the route can not be expanded further, terminate the route. Choose the first feasible link in the list to start a new route.

Step 6. Repeat Steps 4 and 5 until no more links can be chosen.

The worst case behavior for this approach is bounded by a linear function in $log_2(n)$. The calculation of the matrix s_{ij} in Step 2 requires about cn^2 computations for some constant c. Next, in Step 3 savings can be sorted into nonincreasing order via Heapsort method in a maximum of $cn^2 lg(n)$ computations. Step 4 involves at most n^2 computations. Thus, the Clark and Wright savings procedure requires an order of $n^2 \log_2(n)$ computations.

Local Search Algorithms I

A local search algorithm is built around a neighborhood search procedure. Given a tour the algorithm examines all the tours that are "neighboring" to it and tries to find a shorter one. The definition of "neighbor" varies with the details of the particular local search heuristic. The overall process goes as follows: Starting with some initial tour, chosen arbitrary or generated by some other heuristic, if there is no neighboring tour which is shorter than the original one, the process halts. The initial tour is a local optimum with respect to the chosen neighborhood. Otherwise, a shorter neighbor of the original tour is used as a new starting point, and the process is repeated. The method must eventually terminate as there is only a finite number of possible tours.

A neighborhood N for a problem instance (S, f), where S is the set of feasible points and f the objective function, can be defined as a mapping from S to its power set, i.e., $N : S \rightarrow 2^S$. $N(s)$ is called the *neighborhood* of $s \in S$, and contains all the solutions that can be reached from s by a *single move*. Here, the meaning of a move is that of an operator which transforms one solution to another with small modifications. A solution x is called a *local minimum* of f with respect to the neighborhood N if $f(x) \leq f(y), \forall y \in N(x)$.

A local search or improvement algorithm begins with an arbitrary solution and ends up in a local minimum where no further improvement is possible. There are many different ways to implement local search. The best known improvement heuristics for the TSP and VRP are the edge exchange heuristics. The 2–opt and 3–opt heuristics were introduced by Lin and the LK heuristic was introduced by Lin and Kernigham. For a given k, if we define a k–exchange to be the deletion of k edges from a tour and their replacement by k other edges, not already in the tour, then we say that a tour is k–optimal if it is not possible to improve it further via such k–exchanges.

Local Search Algorithms II

2–opt method

A 2–opt procedure, in general, consists of eliminating two edges and reconnecting the two resulting paths in order to obtain a new tour. Note that there is only one way to reconnect the paths. The 2–opt procedure was introduced by Lin (1965) for the TSP:

Step 1. Let T be the current tour.

Step 2. For every node $i = 1, 2, \cdots, n$: Examine all 2–opt moves involving the edge between i and its successor in the tour. If it is possible to decrease the tour length this way, then choose the best such 2–opt move and update T

Step 3. If no improving move could be found, then stop.

Metaheuristic Algorithms I

Simulated Annealing

Simulated annealing (SA) belongs to a class of local search algorithms that are known as *threshold algorithms*. These algorithms play a special role within local search for two reasons. First, they appear to be quite successful when applied to a broad range of practical problems. Second, some threshold algorithms such as SA have a stochastic component, which facilitates a theoretical analysis of their asymptotic convergence. The approach of SA originates from theoretical physics, where Monte–Carlo methods are employed to simulate phenomena in statistical mechanics. Its predecessor is the so-called Metropolis filter. This simulation method can be motivated as follows. Consider a huge number of particles of fixed volume at some temperature ϑ. Since the particles move, the system can be in various states. The probability that the system is in a state of certain energy E is given by the Boltzmann distribution $f(E) = \frac{\exp(\frac{-E}{\kappa_B \vartheta})}{z(\vartheta)}$, where $z(\vartheta)$ is a normalization factor and κ_B is the Boltzmann constant. This distribution characterizes the statistical equilibrium of the system at temperature ϑ. In the following we present a simulated annealing procedure for the TSP:

- **Step 1.** Compute an initial tour T and choose an initial temperature $\vartheta > 0$ and a repetition factor r.
- **Step 2.** If the stopping criterion is not satisfied:
- **Step 2.1.** Do the following r times.

Metaheuristic Algorithms II

- **Step 2.1.1.** Perform a random modification of the current tour to obtain the tour (T') and let $\Delta = c(T') - c(T)$.
- **Step 2.1.2.** Compute a random number x, $0 \leq x \leq 1$.
- **Step 2.1.3.** If $\Delta < 0$ or $x < \exp(\frac{-\Delta}{\vartheta})$.
- **Step 2.2.** Update ϑ and r.
- **Step 3.** Output the current tour T as solution.

Metaheuristic Algorithms III

Tabu Search

- Tabu search (TS) was introduced by Glover. Computational experience has shown that TS is a well established approximation technique, which can compete with almost all known techniques and which, by its flexibility, can beat many classic procedures.
- TS combines the deterministic iterative improvement algorithm with a possibility to accept cost–increasing solutions. Thus, the search is directed away from local minima so that other parts of the search space can be explored.
- The next solution visited is always chosen to be a legal neighbor of the current solution with the best cost, even if that cost is worse than that of the current solution.
- The set of legal neighbors is restricted by a tabu list designed to prevent the search from cycling.
- The tabu list is dynamically updated during the execution of the algorithm.
- The tabu defines solutions that are not acceptable in the next few iterations.
- However, a solution in the tabu list may be accepted if its quality is in some sense good enough, in which case it is said to attain a certain aspiration level.

A general description of TS is the following:

Metaheuristic Algorithms IV

Table 1 : Tabu Search

```
*Initialization*
Create an initial solution $S_0$
Calculate the fitness function of the initial solution
$S^* = S_0$ ! $S^*$ the best known solution
$f(S^*) = f(S_0)$ ! $f$ is the overall error of the solution
*Main Phase*
**Do while** termination criteria are not satisfied
    Calculate neighbor solution $S'$
    **if** $f(S') < f(S^*)$ **then**
        $S^* = S'$, $f^* = f(S')$
    **endif**
    Record the current move in the tabu list (delete the oldest entry)
    Call every $k_1$ iterations the intensification strategy
    **if** $f(S_{intensification}) < f(S^*)$ **then**
        $S^* = S_{intensification}$, $f^* = f(S_{intensification})$
    **endif**
    Call every $k_2$ iterations the diversification strategy
    **if** $f(S_{diversification}) < f(S^*)$ **then**
        $S^* = S_{diversification}$, $f^* = f(S_{diversification})$
    **endif**
**Enddo**
**Return** the best solution.
```

Metaheuristic Algorithms V

Adaptive Memory Procedure

One of the most interesting developments to have occurred in the area of TS in recent years is the concept of Adaptive Memory developed by Rochat and Taillard. It is mostly used in TS, but its applicability is not limited to this type of metaheuristic. An adaptive memory is a pool of good solutions that is dynamically updated throughout the search process. Periodically, some elements of these solutions are extracted from the pool and combined differently to produce new good solutions. In the VRP, vehicle routes selected from several solutions will be used as a starting point. The extraction process gives a larger weight to those routes belonging to the best solutions.

GRASP

The Greedy Randomized Adaptive Search Procedure (GRASP) is a two phase local search. This randomized technique provides a feasible solution within every iteration. The final result is simply the best solution found over all iteration (multi-start local search). Each iteration consists of two phases, a construction phase and a local search procedure. In the construction phase a randomized greedy function is used to build up an initial solution. This solution is then exposed for improvement attempts in the local search phase.

The construction phase can be described as stepwise adding one element at a time to the partial (incomplete) solution. The choice of the next element to be added is determined by ordering all elements in a candidate list with respect to a greedy function. The heuristic is adaptive because

Metaheuristic Algorithms VI

the benefits associated with every element are updated at each iteration of the construction phase to reflect the changes brought on by the selection of the previous element. The probabilistic component of a GRASP is characterized by randomly choosing one of the best candidate in the list but not necessary the top candidate.

A generic GRASP algorithm is given below:

Step 1. Input the problem instance.

Step 2. Do until some GRASP stopping criterion is satisfied:

Step 2.1. Execute the construction phase of GRASP in order to obtain a greedy random solution.

Step 2.2. Apply the local search phase to the obtained greedy solution.

Step 2.3. Update the best solution if needed.

Step 3. Return the best solution found.

Evolutionary and Swarm Intelligence Algorithms I

Genetic Algorithms This search strategy uses concepts from population genetics and evolution theory to construct algorithms that try to optimize the fitness of a population of elements through recombination and mutation of their genes. The general idea of *genetic local search* is give by the following procedure:

Step 1 (Initialize): Construct an initial population of n solutions.

Step 2 (Improve): Use local search to replace the n solutions in the population by n local optima.

Step 3 (Recombine): Augment the population by adding m offspring solutions, the population size now equals $n + m$.

Step 4 (Improve): Use local search to replace the m offspring solutions by m local optima.

Step 5 (Select): Reduce the population to its original size by selecting n solutions from the current population.

Step 6 (Evolute): Repeat Step 3 through 5 until a stop criterion is satisfied.

Ant Algorithms

The ant system, introduced by Colorni, Dorigo and Maniezzo, is a new distributed metaheuristic for hard combinatorial optimization problems and was first used on the traveling salesman problem. Observations on real ants searching for food were the inspiration to imitate the behavior of ant colonies.

Real ants are able to communicate information concerning food sources via an aromatic essence, called pheromone. They mark the path they walk on by laying down pheromone in a quantity that depends on the length of the path and the quality of the discovered food source. Other ants can observe the pheromone trail and are attracted to follow it.

The described behavior of real ant colonies can be used to solve combinatorial optimization problems by simulation: artificial ants searching the solution space simulate real ants searching their environment, the objective values correspond to the quality of the food sources and an adaptive memory corresponds to the pheromone trails. In addition, the artificial ants are equipped with a local heuristic function to guide their search through the set of feasible solutions.

The initial quantity of the pheromone τ_{ij} for each arc (i,j) is calculated from the formula:

$$\tau_{ij} = \frac{ant\_size}{init\_opt} \tag{20}$$

Evolutionary and Swarm Intelligence Algorithms III

Probability of selection of arc (i,j)

$$p_{ij} = \frac{[\tau_{ij}]^{\alpha}[n_i]^{\beta}}{\sum\limits_{l=1}^{M}[\tau_{lj}]^{\alpha}[n_l]^{\beta}} \tag{21}$$

Update pheromone:

$$\tau_{ij} \leftarrow \begin{cases} (1-q)\tau_{ij} + \frac{1}{ant\_opt}, & \text{if arc } \textit{(i,j)} \text{ is selected} \\ (1-q)\tau_i, & \text{otherwise} \end{cases} \tag{22}$$

Evolutionary and Swarm Intelligence Algorithms IV

Table 2 : Ant Colony Optimization

```
Initialization
Creation of the initial population of Ants
Calculation of heuristic function $n_i$
Calculation of the initial pheromone $\tau_{ij}$ for each arc $(i, j)$
Selection of the maximum number of generations
Main Phase
Do until the maximum number of generations has been reached:
    Each ant begins its own tour from a different node
    For each ant in the population:
        Do while stopping criteria are not satisfied:
            Select the next node based on the pheromone
                and the heuristic function
            Calculate the fitness function of each ant
                                Apply a Local Search Procedure
        Enddo
    EndFor
    Update the pheromone based on the solution of the best ant
        in the population
    (Elitist Strategy for ACO)
Enddo
Return the best ant (the best solution).
```

Evolutionary and Swarm Intelligence Algorithms V

Particle Swarm Optimization

Particle Swarm Optimization (PSO) is a very popular global optimization method that was originally proposed by Kennedy and Eberhart as a simulation of the social behavior of social organisms such as bird flocking and fish schooling.

PSO uses the physical movements of the individuals in the swarm and has a flexible and well-balanced mechanism to enhance and adapt to the global and local exploration abilities.

In general, in a PSO algorithm a set of solutions are used where each solution is denoted as a particle. These solutions create the swarm. In many algorithms more than one swarms exist (a Multiswarm version of the algorithm). Initially, the solutions are randomly initialized in the solution space. Two are the main vectors that describe a particle, the position vector (x_{ij}, where i denotes the particle ($i = 1, \cdots, N$, N is the swarm size) and j denotes the corresponding dimension of the particle ($j = 1, \cdots, d$, d is the dimension of the problem)) and the velocities vector (v_{ij}). The performance of its particle is evaluated on the predefined fitness function ($f(x)$).

Thus, each particle is randomly placed in the d-dimensional space as a candidate solution. It is very important to have a suitable initialization of the particles. If the particles are initialized completely at random, then, there is a possibility to leave most of the solution space without particles and this will lead to a slower convergence to the global optimum or to a good local optimum or there is a possibility for the algorithm to be trapped in a bad solution. One very

Evolutionary and Swarm Intelligence Algorithms VI

simple and effective way to initialize the particles is when the minimum value of the solution space is denoted as $x_{min,j}$ and the maximum value is denoted as $x_{max,j}$, then, the values of the solution vector for each particle is calculated from the following equation:

$$x_{ij}(0) = x_{min,j} + r_j(x_{max,j} - x_{min,j}) \tag{23}$$

where r_j is a random number between $(0, 1)$.

The velocity of the i-th particle v_{ij} is defined as the change of its position. The flying direction of each particle is the dynamical interaction of the individual and the social flying experience. Thus, the particles are initialized with a velocity value either equal to zero or using a random number in the solution space. The algorithm completes the optimization through following the personal best solution of each particle and the global best value of the whole swarm. Thus, in each iteration, except of the current position, another vector is used which is the personal best position of each particle through the iterations. The best member of the personal best position vector is the global best position of the whole swarm. Each particle adjusts its trajectory towards its own previous best position and the global best position, namely $pbest_{ij}$ and $gbest_j$, respectively. The velocities and positions of particles are updated using the following formulas:

Evolutionary and Swarm Intelligence Algorithms VII

$$v_{ij}(t+1) = v_{ij}(t) + c_1 rand_1 (pbest_{ij} - x_{ij}(t)) + c_2 rand_2 (gbest_j - x_{ij}(t)) \tag{24}$$

$$x_{ij}(t+1) = x_{ij}(t) + v_{ij}(t+1) \tag{25}$$

where t is the iterations counter, c_1 and c_2 are the acceleration coefficients, $rand_1$ and $rand_2$ are two random variables in the interval (0, 1). The values of c_1 and c_2 could be constant or they could be adapted during the iterations. Usually in the most published papers concerning an application of PSO in a continuous or discrete problem, these values were set equal to 2. However, there is a possibility to take different values or to adjust their values during the iterations, for example using the following equations:

$$c_1 = c_{1,min} + \frac{c_{1,max} - c_{1,min}}{iter_{max}} \times t \tag{26}$$

Evolutionary and Swarm Intelligence Algorithms VIII

$$c_2 = c_{2,min} + \frac{c_{2,max} - c_{2,min}}{iter_{max}} \times t \tag{27}$$

where $iter_{max}$ is the maximum number of iterations and $c_{1,min}$, $c_{1,max}$, $c_{2,min}$, $c_{2,max}$ are the minimum and maximum values that c_1, c_2 can take, respectively. In the beginning of the procedure the values of c_1 and c_2 are small and, then, are increasing until they reach to their maximum values. By doing this, on the first iterations there is a great freedom of movement in the particles' solution space in order to find the optimum quickly.

A number of different velocities equations have been proposed during the last years. Nowadays, other researchers use one of the velocities equations proposed in the past that have been proved to perform well (mainly the inertia equation or the constriction equation) or less frequently they propose a new one which either is a variant of a previously published equation or it simulates something from the nature in which the specific researcher focuses in his study. The most important and known velocities equations are the following:

Evolutionary and Swarm Intelligence Algorithms IX

```
algorithm Particle Swarm Optimization
Initialization
Select the number of Swarms
Select the number of Particles for each swarm
Initialization of the position and velocity of each particle
Calculation of the initial cost function (fitness function) value of each particle
Keep Global Best Particle (Solution) of the whole swarm
Keep Personal Best of each particle
Main Phase
Do until the maximum number of iterations has not been reached:
    Calculate the velocity of each particle
    Calculate the new position of each particle
    Evaluate the new fitness function of each particle
    Update the best solution of each particle
    Update the best particle of the whole swarm
Enddo
Return the best particle (the global best solution)
```