# Genetic algorithm with iterated local search for solving a location-routing problem (2012)

## by Houda Derbel, Bassem Jarboui, Saïd Hanafi, Habib Chabchoub

L.T. van Binsbergen S. Fafianie J.P. Pizani Flor

Department of Information and Computing Sciences
Utrecht University

Monday 7$^{th}$ January, 2013



Universiteit Utrecht

# Table of Contents

# Problem Description

# Problem Definition

- Location Routing Problem (LRP)
- set of costumers $I = \{1, \ldots, n\}$
- set of potential depots $J = \{1, \ldots, m\}$
- limited capacity $b_j$ and fixed cost $f_j$
- non-negative demand $d_i$
- travelling cost $c_{ij}$



Universiteit Utrecht

# Problem Definition

- each depot has a single incapacitated vehicle
- vehicle begins and ends its route at its depot
- find a subset of depots to be opened
- elaborate vehicle tours to meet customer demands
- minimize total cost of location and delivery

# Related Work

- combination of Vehicle Routing Problem(VRP) and Facility Location Problem(FLP)
- branch and bound method - Laporore and Norbert(1981)
  - single-facility LRP
  - no tour length restrictions
- branch and cut method - Laport, Norbert and Arpin(1986)
  - capacitated vehicles and depots (CLRP)
  - fixed number of vehicles
- heuristic approaches
  - simulated annealing - Wu, Low and Bai (2002)
  - greedy randomized adaptive procedure (GRASP)
  - tabu search - Albreda-Sambola et al. (2005)

# Hybrid Approach

- Genetic Algorithm
  - population of solutions may lead to global optimum
  - sub-optimal solutions are not improved fast enough
- Iterated Local Search
  - find local optimum quickly
  - may not find global optimum
- hybrid approach maximizes the chance of convergence to an optimal solution by using various search spaces

# Hybrid Approach

- generate and evaluate random population of solutions
- in each cycle:
  - select parents $x_1$ and $x_2$
  - apply crossover to $x_1$ and $x_2$ to generate child $x_{new}$
  - apply mutation to $x_{new}$
  - apply ILS to $x_{new}$ if $\text{fitness}(x_{new}) < (1 + \delta) \cdot fitness_{best}$
  - select fittest

Genetic Algorithm

# Solution Representation

- solution $x$ is represented by:
  - $A(x) = \{a_1, \ldots, a_n\}$ assignment configuration
  - $a_i = j$ means costumer $i$ is assigned to depot $j$
  - $P(x) = \{p_1, \ldots, p_n\}$ rank of a costumer on a given route
  - customer $p_i$ is served before $p_{i'}$ if $i < i'$



Universiteit Utrecht

# Solution Representation

$$A = \begin{array}{|c|c|c|c|c|c|c|c|}\hline 3 & 1 & 2 & 3 & 2 & 1 & 2 & 3 \\ \hline\end{array}$$

$$P = \begin{array}{|c|c|c|c|c|c|c|c|}\hline 6 & 2 & 3 & 1 & 5 & 8 & 7 & 4 \\ \hline\end{array}$$

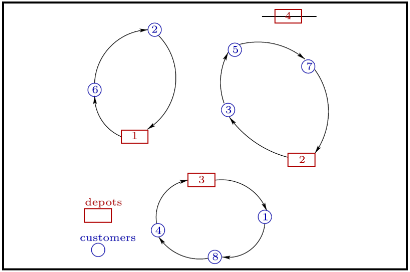



**Fig. 1.** An example of LRP solution representation.

# Parent Selection

- $\mathbb{P}([k]) = \frac{2k}{M(M+1)}$
- $[k]$ is the $k$th chromosome in descending order
- $M$ is the population size

# Crossover operator

- 1-point crossover for the assignment configuration $A$
- 1-point order crossover for the permutation configuration $P$:

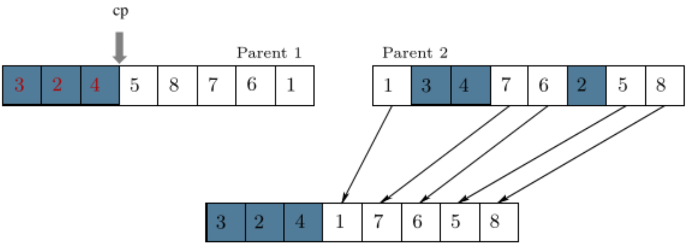



**Fig. 2.** Crossover operation for the permutation vector.



Universiteit Utrecht

# Mutation

## Assignment configuration

- Mutating $A$ by randomly changing an assignment to any other depot
- Possibly introducing a new depot, or removing one
- Performed according to a probability distribution $\mathbb{P}_a$

## Permutation configuration

- Mutation on $P$ is performed by taking a random customer and inserting it at a random position
- Shifting other customers towards the old location of the customer
- Performed according to probability distribution $\mathbb{P}_p$

# Fitness function

- $fitness(x) = cost(x) + penalty(x)$
- $cost(x)$ is the sum of all the driving and depot costs
- $penalty(x) = \sum_{j \in J} \alpha \, max\{0, D_j(x) - b_j\}$

# Replacement

- The newly created child is compared to the worst in the current population



Universiteit Utrecht

# Iterated Local Search

# ILS structure

**Algorithm 1** General structure of the used ILS

**Require:** $x_0$ is an initial solution
  $\hat{x} \leftarrow localsearch(x_0)$
  **repeat**
    $x \leftarrow perturbation(\hat{x})$
    $\tilde{x} \leftarrow localsearch(x)$
    **if** $fitness(\hat{x}) < fitness(\tilde{x})$ **then** $\hat{x} \leftarrow \tilde{x}$
    **end if**
  **until** Termination condition is met

# Local search method used

**Algorithm 2** General structure of the local search method used

**Require:** an initial solution $x$

$x_1 \leftarrow$ first improvement on $x$ using neighbourhood $\mathcal{N}1$
$x_2 \leftarrow$ first improvement on $x_1$ using neighbourhood $\mathcal{N}2$
$x_3 \leftarrow$ first improvement on $x_2$ using neighbourhood $\mathcal{N}3$
$x_4 \leftarrow$ first improvement on $x_3$ using neighbourhood $\mathcal{N}4$
**if** $\mathit{fitness}(x_4) < \mathit{fitness}(x_1)$ **then**
  $x \leftarrow x_4$
  Go to line 1
**end if**

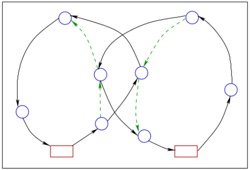


Universiteit Utrecht

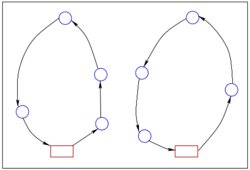
# Neighbourhood structures

Four structures were used:

- N1 and N2: involving 2 routes
  - N1: swap two customers

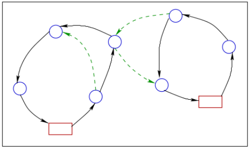


(a) initial solution $x$

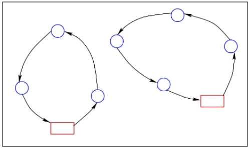


(b) neighboring solution in $\mathcal{N}1(x)$

  - N2: move customer from one route to another



(a) initial solution $x$



(b) neighboring solution in $\mathcal{N}2(x)$

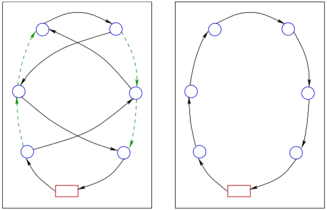


Universiteit Utrecht

# Neighbourhood structures

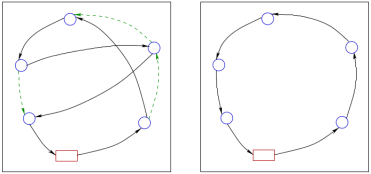
Four structures were used:

- N3 and N4: intra-route
  - N3: swap two customers



(a) initial solution $x$ (b) neighboring solution in $\mathcal{N}3(x)$

  - N4: move customer to another position in the route



(a) initial solution $x$ (b) neighboring solution in $\mathcal{N}4(x)$



Universiteit Utrecht

# Perturbation criterion

- Local moves concern only open depots
- Perturbation opens new depots, preserving variability
- Perturbation criterion:
  - Select a random *open* depot
  - Move the customer assigned from the original depot to another (open or closed) one.
  - Affects only configuration A of each chromossome (assignment)



Universiteit Utrecht

# Conclusions and Comparison

# Test instances

- Benchmarks proposed by Albreda-Sambola et al. (2005)
- Five sets of instances: S1, S2, S3, M2, M3
  - S1, S2 and S3: 5 facilities, 10, 20 and 30 customers
  - M2 and M3: 10 facilities, 20 and 30 customers
- Instances further parameterized by 2 other variables:
  - $R_1$: Ratio between total customer demand and total depot capacity
  - $R_2$: Value proportional to the fixed cost of opening a depot



Universiteit Utrecht

# Parameter setting

- Generic parameters:
  - Population size (M): 40
  - Mutation probability on configuration A ($\mathbb{P}_p$): 0.7
  - Mutation probability on configuration P ($\mathbb{P}_p$): 0.9
  - *Penalty* constant used in fitness evaluation ($\alpha$): 1000
- ILS parameters:
  - $\delta$ coefficient: 0.1 (ILS used rarely)
  - Termination condition: 100 sucessive iterations with no improvement

# Comparative study

- Execution results compared with best-known solutions
- Best-known solutions: Albreda-Sambola et al. (2005), using *tabu search*
- Two dimensions were measured in the experiment:
  - *%gap*: average deviaton of found solution to the a-priori lower bound (global optimum)
  - *Time*: running time over ten instances
- *t-test* done over %gap to verify the divergence between the two scenarios

# Experimental results

Some notable results from the comparative study:

- **S1:** GA&ILS finds all optima and beats TS in running time, but pure ILS comes close (*%gap*) in less time.

# Experimental results

Some notable results from the comparative study:

- **S1:** GA&ILS finds all optima and beats TS in running time, but pure ILS comes close (*%gap*) in less time.
- **S2:** GA&ILS has slightly smaller *%gap* than pure ILS, both much better than TS

# Experimental results

Some notable results from the comparative study:

- **S1:** GA&ILS finds all optima and beats TS in running time, but pure ILS comes close (*%gap*) in less time.
- **S2:** GA&ILS has slightly smaller *%gap* than pure ILS, both much better than TS
- **M3 (largest):** ILS beats TS completely and GA&ILS slightly in terms of *%gap*, TS has around 10x larger running time than both others.

# Experimental results

Some notable results from the comparative study:

- **S1:** GA&ILS finds all optima and beats TS in running time, but pure ILS comes close (*%gap*) in less time.
- **S2:** GA&ILS has slightly smaller *%gap* than pure ILS, both much better than TS
- **M3 (largest):** ILS beats TS completely and GA&ILS slightly in terms of *%gap*, TS has around 10x larger running time than both others.
- **t-test (%gap):** ILS and GA&ILS beat TS with error risk close to 0. GA&ILS beats pure ILS with error risk of 15%.



Universiteit Utrecht

# Conclusions

- Hybridization between GA and ILS to solve the LRP efficiently
  - ILS improves each generation outputted by the GA
  - Genetic operators AND neighbourhood structures take into account location and routing *simultaneously*

# Conclusions

- Hybridization between GA and ILS to solve the LRP efficiently
  - ILS improves each generation outputted by the GA
  - Genetic operators AND neighbourhood structures take into account location and routing *simultaneously*
- Proposed algorithm was compared to five problem sets from the literature
  - Improves over best-known approach (TS) both in quality of solutions *and* in computational requirements

# Conclusions

- Hybridization between GA and ILS to solve the LRP efficiently
  - ILS improves each generation outputted by the GA
  - Genetic operators AND neighbourhood structures take into account location and routing *simultaneously*
- Proposed algorithm was compared to five problem sets from the literature
  - Improves over best-known approach (TS) both in quality of solutions *and* in computational requirements
- Authors suggest applying VNS (Variable Neighbourhood Search) combined with GA as future study