Solving the Dial-a-Ride Problem with the Firefly Metaheuristic

Fernando Bombardelli da Silva

Technische Universität Berlin
Fakultät IV Elektrotechnik und Informatik
Bachelorstudiengang Informatik

Bachelorarbeit
Advisor: Dr.-Ing. Axel Heßler

bombardelli.f@gmail.com

Berlin, 21st March 2016

Contents

Introduction & Motivation

- The DARP: A vehicle routing problem characterized by
  - User-orientation
  - Small vehicles, like vans, shared by users
  - Flexible routes and schedules
- In Germany: (An-)Rufbus or Taxibus. In Switzerland: Publicar
- Example of application: Patients of hemodialysis
- Their need: Door-to-door transport from home to the hospital

Introduction & Motivation

- The DARP: A vehicle routing problem characterized by
  - User-orientation
  - Small vehicles, like vans, shared by users
  - Flexible routes and schedules
- In Germany: (An-)Rufbus or Taxibus. In Switzerland: Publicar
- Example of application: Patients of hemodialysis
- Their need: Door-to-door transport from home to the hospital
- **Motivations:**
  - Improvement of urban mobility and public transportation
  - Reduction of costs for people who need this kind of transport

Approach & Contribution

- Problem of optimization of the operation costs
- Subject to a set of constraints that guarantee comfort

Approach & Contribution

- Problem of optimization of the operation costs
- Subject to a set of constraints that guarantee comfort
- We proceed with 2 approaches:
  - Exact approach with a generic solver
  - Near-optimal approach with the implementation of a metaheuristic

Approach & Contribution

- Problem of optimization of the operation costs
- Subject to a set of constraints that guarantee comfort
- We proceed with 2 approaches:
  - Exact approach with a generic solver
  - Near-optimal approach with the implementation of a metaheuristic
- Our contribution:
  - Comparison of the generic solver with the metaheuristic
  - Application of swarm intelligence to the DARP
  - New way to represent the solution

The Dial-a-Ride Problem (DARP)

- The DARP is a vehicle routing problem, proven to be $\mathcal{NP}$-hard
- It "extends" the *Pickup and Deliver Vehicle Routing Problem* and the *Vehicle Routing Problem with Time Windows*

The Dial-a-Ride Problem (DARP)

- The DARP is a vehicle routing problem, proven to be $\mathcal{NP}$-hard
- It “extends” the *Pickup and Deliver Vehicle Routing Problem* and the *Vehicle Routing Problem with Time Windows*
- There is a fleet of **homogeneous vehicles** with a **unique depot**
- Set of transport requests from passengers, known beforehand (**static** version)
- Maximal vehicle route duration and vehicle capacity
- Maximal user ride time

Request Characteristics

- Pick-up and drop-off locations (ordered pairs)
- Time windows for the pick-up and for the drop-off
- Time necessary for boarding or alighting
- Quantity of passengers

Request Characteristics

- Pick-up and drop-off locations (ordered pairs)
- Time windows for the pick-up and for the drop-off
- Time necessary for boarding or alighting
- Quantity of passengers
- The requests form a full graph G=(V,A), where
  - V is the set with every location of every request
  - A is the set of edges that tell the travel costs and travel times between any pair of locations

Request Characteristics

- Pick-up and drop-off locations (ordered pairs)
- Time windows for the pick-up and for the drop-off
- Time necessary for boarding or alighting
- Quantity of passengers
- The requests form a full graph G=(V,A), where
  - V is the set with every location of every request
  - A is the set of edges that tell the travel costs and travel times between any pair of locations
- The costs and time are assumed to be the *Euclidean distance* between the locations

Goal

Definition (Route)

The order of the locations through which a vehicle travels. It always starts and ends at the depot.

Goal

Definition (Route)

The order of the locations through which a vehicle travels. It always starts and ends at the depot.

Definition (Feasible Solution)

- Assignment of every request to one, and only one, vehicle
- Determination of the route of every vehicle
- Every constraint must be fulfilled

Goal

Definition (Route)

The order of the locations through which a vehicle travels. It always starts and ends at the depot.

Definition (Feasible Solution)

- Assignment of every request to one, and only one, vehicle
- Determination of the route of every vehicle
- Every constraint must be fulfilled

Optimal Solution: Feasible solution whose total duration time is less than or equal to any other feasible solution's total duration time

Exact Approach

- **Integer Linear Programming**
- Method commonly used in mathematics and operations research
- Represent the problem in a mathematical language (e.g. *AMPL*)
- Solve via generic solver (e.g. *GNU Linear Programming Kit*)
- Algorithms behind it: *Simplex*, *branch and bound*, *cutting plane*

Near-optimal Approach

- Implement a metaheuristic named **Firefly Algorithm (FA)**
- Other examples: *Simulated annealing*, *genetic algorithms*
- The FA is nature-inspired and applies the concept of swarm intelligence
- Similar to: *Ant colony*, *particle swarm optimization*

Near-optimal Approach

- Implement a metaheuristic named **Firefly Algorithm (FA)**
- Other examples: *Simulated annealing*, *genetic algorithms*
- The FA is nature-inspired and applies the concept of swarm intelligence
- Similar to: *Ant colony*, *particle swarm optimization*
- Represent the position of a firefly as a vector
- Represent a solution as a firefly (i.e. the vector of its position)

Near-optimal Approach

- Simulate the movements of the insects, based on 3 rules:

Near-optimal Approach

- Simulate the movements of the insects, based on 3 rules:
- Fireflies are attracted to each other regardless of their sex

Near-optimal Approach

- Simulate the movements of the insects, based on 3 rules:
- Fireflies are attracted to each other regardless of their sex
- The better the solution represented by a firefly, the brighter is its emitted light

Near-optimal Approach

- Simulate the movements of the insects, based on 3 rules:
- Fireflies are attracted to each other regardless of their sex
- The better the solution represented by a firefly, the brighter is its emitted light
- The attractiveness of a firefly **x** to another one **y** is directly proportional to the brightness of **y** and inversely proportional to the distance between them

Near-optimal Approach

- Simulate the movements of the insects, based on 3 rules:
- Fireflies are attracted to each other regardless of their sex
- The better the solution represented by a firefly, the brighter is its emitted light
- The attractiveness of a firefly **x** to another one **y** is directly proportional to the brightness of **y** and inversely proportional to the distance between them
- **Result:** Ideally, there will be a convergence of fireflies to the optimum

Solution Representation

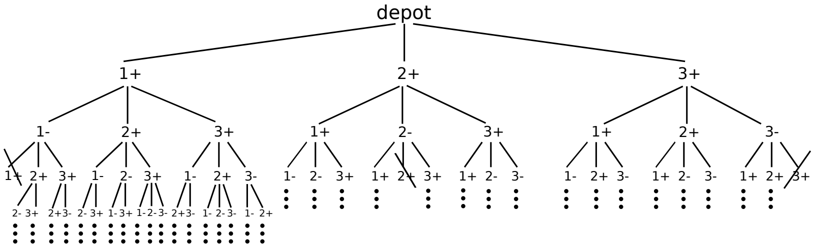
- $\mathbf{v} = (v_1, v_2, ..., v_{k+1})$, where $v_1, v_2, ..., v_{k+1} \in \mathbb{N}_{\geq 0}$ and k is the number of vehicles
- Initial solution easily generated, simply random numbers in an interval

Solution Representation

- $\mathbf{v} = (v_1, v_2, ..., v_{k+1})$, where $v_1, v_2, ..., v_{k+1} \in \mathbb{N}_{\geq 0}$ and k is the number of vehicles
- Initial solution easily generated, simply random numbers in an interval
- A value in v_1 represents a unique assignment from requests to vehicles, it may vary between 0 and k^n, where n is the number of requests

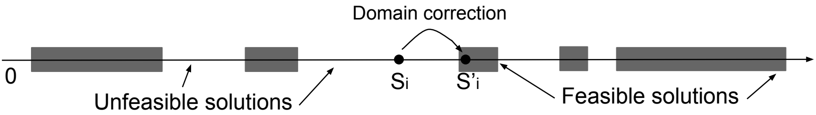
Solution Representation

- The values in $v_2, \ldots, v_{k+1}$ represent each vehicle's routes, analogously to v_1
- The possible arrangement of the requests can be seen as a tree



Correction of Unfeasibility

- Time constraints cause unfeasibilty of solutions
- Fireflies in these unfeasible regions should be moved into a feasible one
- Constraint satisfaction problem + Arc Consistency #3



Generic Solver vs. Metaheristic

- CPU: Intel Xeon 2.3 GHz 64bits, 15GB of RAM, Linux OS
- Generic solver: GNU Linear Programming Kit.
- Metaheuristic implemented in Python with SciPy
- Instances created from an instance of the literature

Generic Solver vs. Metaheristic

- Solution through GLPK is impracticable. Exponential growth of time
- Firefly implementation shows a steady growth of CPU time

Table: Results of the first evaluation

Inst.	Req.	Opt.	CPU-GLPK	Firefly Sol.	% Opt.	CPU-Firefly
Test-1	2	58.05	0.1	69.75	83.2%	7.4
Test-2	4	68.10	0.9	98.51	69.1%	11.7
Test-3	5	76.27	2.0	134.29	56.8%	2.5
Test-4	6	96.54	45.0	149.32	64.6%	3.3
Test-5	8	-	-	158.67	-	5.7

Metaheristic with Literature Instances

- 24 instance from the literature
- Smaller instances: 90% of optimality in average
- Deviation to the initial solution varies considerably

Table: Results of the second evaluation – Quality and progress

Instance	Optimum	Final Sol.	Optimality	Initial Sol.	Dev. to Initial Sol.
a2-16	294.25	312.96	94.02%	335.85	6.81%
a2-20	344.83	373.89	92.23%	427.48	12.54%
a2-24	431.12	442.85	97.35%	496.92	10.88%
a3-18	300.48	347.10	86.57%	374.20	7.24%
a3-24	344.83	409.73	84.16%	456.86	10.32%
a3-30	494.85	614.13	80.58%	615.99	0.30%
a3-36	583.19	-	-	-	-
a4-16	282.68	310.84	90.94%	354.54	12.33%

Metaheristic with Literature Instances

- Optimality sinks as instance sizes rise

Table: Results of the second evaluation – Quality and progress

Instance	Optimum	Final Sol.	Optimality	Initial Sol.	Dev. to Initial Sol.
a4-24	375.02	481.16	77.94%	567.63	15.23%
a4-32	485.50	639.73	75.89%	665.27	3.84%
a4-40	557.69	780.13	71.49%	841.92	7.34%
a4-48	668.82	956.90	69.89%	1032.38	7.31%
a5-40	498.41	779.32	63.95%	818.14	4.74%
a5-50	686.62	926.23	74.13%	1018.84	9.09%
a5-60	808.42	1195.94	67.60%	1257.38	4.89%
a6-48	604.12	993.08	60.83%	1025.55	3.17%

Metaheristic with Literature Instances

- Comparison of CPU times to another two approaches
- CPU^1: Parragh (2011). CPU^2: Parragh and Schmid (2013).

Table: Results of the second evaluation – Running time

Instance	CPU^1	CPU^2	CPU-Firefly
a2-16	68.2	0.12	**63.3**
a2-20	133.8	0.28	160.5
a2-24	187.8	0.35	419.3
a3-18	45.4	-	**29.5**
a3-24	86.8	0.29	**77.4**
a3-30	105.6	0.50	151.2
a3-36	162.6	0.83	370.6
a4-16	26.0	-	**14.5**
a4-24	50.8	-	**37.0**
a4-32	86.0	0.55	129.4
a4-40	130.6	0.78	233.5
a4-48	253.8	1.62	**222.3**

Conclusion

- Solution via generic solver is impracticable
- Solution via FA can solve much larger instances, though not the largest ones
- Main issue: The time constraints (time windows and ride time)
- How to overcome: Rethink solution representation, for example
- Improve efficiency: Reduce domain corrections and improve transformation functions

Bibliography

Braekers, Kris, An Caris, and Gerrit K. Janssens. 2014. "Exact and meta-heuristic approach for a general heterogeneous dial-a-ride problem with multiple depots." *Transportation Research Part B: Methodological* 67:166–186.

Cordeau, Jean-François, and Gilbert Laporte. 2007. "The dial-a-ride problem: models and algorithms." *Annals of Operations Research* 153 (1): 29–46.

Parragh, Sophie N. 2011. "Introducing heterogeneous users and vehicles into models and algorithms for the dial-a-ride problem." *Transportation Research Part C: Emerging Technologies* 19 (5): 912–930.

Parragh, Sophie N., and Verena Schmid. 2013. "Hybrid column generation and large neighborhood search for the dial-a-ride problem." *Computers & Operations Research* 40 (1): 490–497.

Ropke, Stefan, Jean-François Cordeau, and Gilbert Laporte. 2007. "Models and branch-and-cut algorithms for pickup and delivery problems with time windows." *Networks* 49 (4): 258–272.

Yang, Xin-She, and Xingshi He. 2013. "Firefly Algorithm: Recent Advances and Applications." *International Journal of Swarm Intelligence* 1 (1): 36–50.

Thank you for your attention

Questions...