EnviroInfo 2011: Innovations in Sharing Environmental Observations and Information

eCarUs - a tool for optimal placement of stations for electric vehicle batteries

Jérôme Agater, Helge Arjangui, Malin Gandor, Michael Mengelkamp, Henning Schröder, Janina Siemer, Marcel Strudthoff, Bernhard Wagner, Ute Vogel, Christian Hinrichs[1], Stefan Scherfke[2]

Abstract

The eCarUs tool implements heuristic optimizations to find the optimal placement for battery stations for electric cars by using a simulation of traffic using real world data. A second optimization phase can calculate the required number of battery slots for each station.

1. Introduction

According to predictions between 1 and 2.5 million electric cars will drive on Germany's streets by the year 2020[3]. That would be 1-2% of all cars. The greenhouse effect might be reduced by saving non-renewable resources. The reliance on oil is a economical as well as a political problem while electricity could be generated locally e.g. from sustainable energy sources like wind farms. But besides these benefits - with this relatively new technology also comes new challenges.
A full battery lasts for about 200 kilometres and charging takes a few hours. Additionally there is no fully developed infrastructure for electric vehicles (EVs). This leads to acceptance issues for the introduction of EVs in Germany.

Current developments suggest that instead of charging a battery, the better alternative would be to exchange the battery of the EV with a new one which would take only a few minutes. This is where the eCarUs tool comes in. It helps to plan the needed and currently missing infrastructure. It is able to find the optimal placement for exchange stations. Additionally it calculates the required number of batteries which each exchange station needs to keep in stock.

The eCarUs tool was developed by a group of eight students from the Carl von Ossietzky University of Oldenburg. We worked for one year on this project, each student about 16-18 hours a week.

2. Requirements

The main project requirements were developed around the question how an EV could travel a long-distance route like Hamburg–Munich without having to do longer stops for recharging its battery. This resulted in the following functional requirements:

---

1 Carl von Ossitzky University Oldenburg, Department for Computer Science, Environmental Informatics, D-26111 Oldenburg, Germany

2 OFFIS e.V., Escherweg 2, D-26121 Oldenburg

3 http://bundesumweltministerium.de/english/mobility/doc/47503.php (Cabinet Adopts Government Programme for Electric Mobility, last visited June 2011)

- The goal is to place stations with minimum costs and maximum satisfaction as well as determining the stock of batteries held at each station. This means a car should always find a station before the battery is drained. Each station should only keep as much batteries in stock as really needed.
- The quality of found solutions should be rated by customer satisfaction, because this is relevant for the acceptance of the technology.

Non-functional Requirements

- The traffic simulation should run on a real road map of Germany. Adding other countries later on should be no problem.
- The traffic flow should be based on realistic data, meaning different traffic flow for holidays, work-days, commuting traffic etc.
- The user should be able to configure all settings easily. So the application should have a graphical user interface (GUI).
- The optimization progress should be shown to the user. The placement of stations could be shown on a map in the GUI.
- Exporting the results should be possible for further processing (CSV/XLS preferred).

3. Design

3.1 Data sets

3.1.1 Traffic Data

From the Federal Ministry for the Environment, Nature, Conservation and Nuclear Safety we got statistic data about traffic based on actual counting and extrapolation. In the bachelor thesis of one team member a program was created which transforms these statistics into traffic flow data which is usable for the simulator ([Strudthoff2010]). The output data is basically a very long comma-separated file which contains a time, a starting- and an end-point.

3.1.2 Geographical Data

The OpenStreetMap[4] project provides free map data. The quality and depth of detail meets the requirements to simulate cars driving on major streets and highways. The map data is available in XML format and can be converted and imported into a PostGIS database. The PgRouting extension for PostGIS implements Dijkstra's algorithm for the shortest path problem ([Dijkstra1959]) which can be used by our simulation to let the vehicles drive on realistic routes for given start- and end-points.

3.2 Technologies

The SimPy framework[5] fulfilled the requirements for the simulation engine. SimPy is an object-oriented, process-based discrete-event simulation framework. The usage is easy and it provides good documentation.

The simulation has to keep track of all driving EVs and their batteries. The batteries discharge while driving and will be recharged at the battery station. A full battery can then be exchanged again. The distance an EV can drive with a full battery, the time it takes to charge etc. will be configurable with the GUI. By choosing SimPy the programming language[6] was automatically set Python. Python is a modern very-

[4] http://www.openstreetmap.org/

[5] http://simpy.sourceforge.net/

[6] http://www.python.org

high-level language which is often used in scientific computing. It is a interpreted object-oriented language which is known for rapid development but cannot offer the same speed as compiled languages. Nevertheless letting the simulation engine handle thousands of EVs was no real problem. The PyQt library[7] was chosen as the best option to create complex user interfaces. PyQt is a binding for the platform neutral GUI toolkit Qt[8].

4. Optimization

Several optimization algorithms were evaluated. Here are the most promising candidates. Algorithms like ant colony optimization (ACO) were also evaluated but will be omitted in this paper. To determine customer satisfaction for a temporal solution, the simulation runs with the current set of battery exchange station positions. This takes a fixed amount of time. Some optimization algorithms need many of such simulation runs to find a viable solution which prevented their use due to timing constraints.

4.1 Tabu search

Tabu search requires a neighbourhood search function. Our database contains about 680,000 edges which are all connected to form the major streets and highways. If every square kilometre in Germany contains only 1.9 edges every station would have over 300 neighbours in a range of 10 kilometres. For 20 stations the number of possible neighbours raises to the power of 300^{20}. Therefore tabu search was not applicable for finding the optimal placement.

4.2 Evolutionary algorithms

For these algorithms a start population would be created using random positions. Depending on the start population the solution space might not be fully searched. Another disadvantage is that a population could contain many solutions. Calculating the fitness for each solution can take a very long time because the fitness is calculated by the simulation results (basically the ratio between total number of EVs and the number of EVs which did reach their destination).

4.3 Flow Refuelling Location Model

Kuby et al. published research ([Kuby2009]) about optimal placement of stations for vehicles with alternative fuel, especially hydrogen-powered cars in California. The objectives of FRLM seem very similar to our problem but it was not clear if the model would equally work well with the dense German road network. Exchanging batteries was not intended in FRLM. The integration of our simulation seemed impossible, too.

4.4 Heuristic algorithms

In conclusion it was decided to create our own heuristic algorithms which are very specific to our problem. A simple one basically works like this: Every EV leaves a trace. Where traces overlap most a station will be placed if possible (depending on extra factors like e.g. investment sum). The simulation is used to evaluate the placement algorithms. The placement will be rated by fitness functions, which get the number of EVs which reached their destination, as input.

---

7 http://www.riverbankcomputing.co.uk/software/pyqt/intro

8 http://qt.nokia.com/products/

5. Implementation

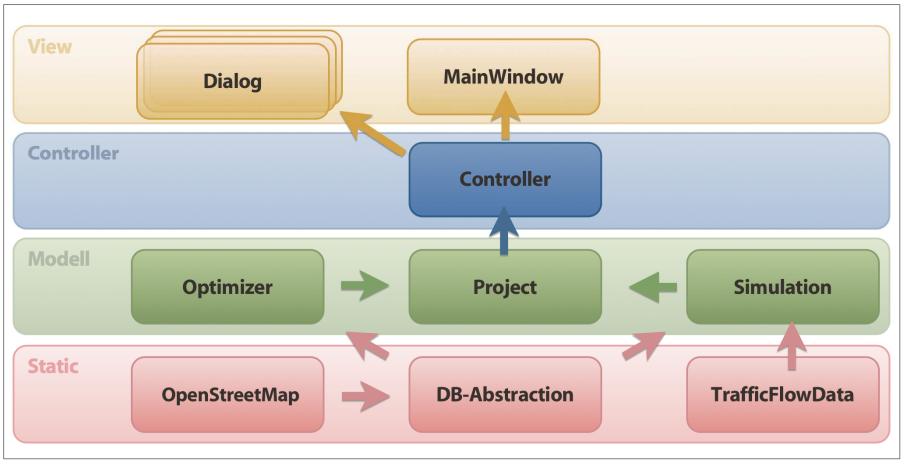

Figure 1: Architecture

The application has a classic model-view-controller architecture and makes heavy use of common patterns like the observer and the delegate pattern. Besides the main GUI application a simple command line application was created for debugging purposes which could be used as a base for a distributed application for parallel processing very large data sets. Using a geographic information system greatly helped to concentrate on the main problem without having to worry about efficient spatial data storage and retrieval.

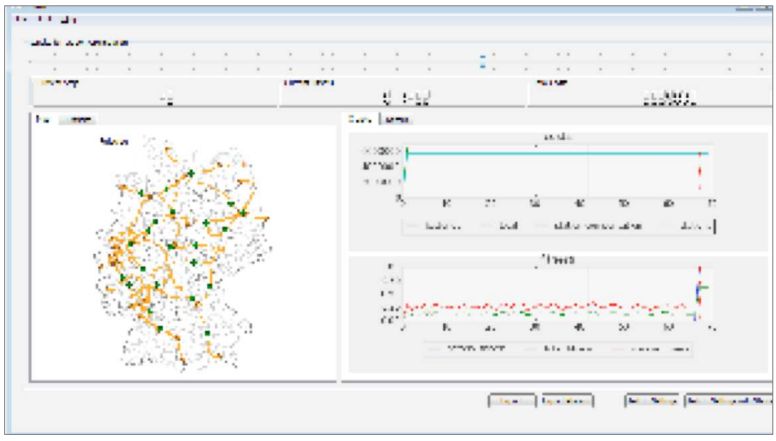

Figure 2: GUI

The interfaces between the different components are highly abstracted. Exchanging e.g. the routing layer or the optimization algorithms can be done without touching the code of the GUI or the simulation. To prevent inter-component-dependencies the simulation, the optimization and the fitters are called by the controller. At the end the possibility to export the complete simulation-, optimization- and fitness-results into delimiter separated text files and Excel sheets was implemented.

6. Conclusions

The presented application is a tool to plan the infrastructure for the emerging e-mobility market. It complies with all specified requirements and can find good placements for battery exchange stations. Additionally it can predict how many batteries each station has to keep in stock based on simulated traffic flow. The flexible architecture leaves room for experiments with new algorithms and extensions.
The current model assumes that batteries are charged at the station and can be exchanged again after they are fully charged. It would be interesting to see how deferred charging (e.g. based on varying electricity prices) might affect the number of required batteries. Large scale controlling unused batteries at all stations could also help to stabilize the operating reserve.

Another idea is to equip each EV with a (virtual) intelligent navigation system which can communicate with the stations in the near range to gather information where batteries are available. If the driver accepts a small detour the number of required stations/batteries might be reduced significantly.

Bibliography

[Strudthoff2010] Strudthoff, Marcel (2010): Entwicklung eines Modells zur Generierung von Verkehrsstromdaten im Hinblick auf einen steigenden Bedarf von Wechselstationen für Elektroautos. Universität Oldenburg, Oldenburg (Oldb.)

[Dijkstra1959] Dijkstra, E. W.(1959): A note on two problems in connexion with graphs

[Kuby2009] Kuby, Michael (et al.) (2009): Optimization of hydrogen stations in floriada using the flow-refueling location model. International Journal of Hydrogen Energy