An Experiment for the Virtual Traffic Laboratory: Calibrating Speed Dependency on Heavy Traffic

A Demonstration of a Study in a Data Driven Traffic Analysis

Arnoud Visser, Joost Zoetebier, Hakan Yakali, and Bob Hertzberger

Informatics Institute, University of Amsterdam

Abstract. In this paper we introduce an application for the Virtual Traffic Laboratory. We have seamlessly integrated the analyses of aggregated information from simulation and measurements in a Matlab environment, in which one can concentrate on finding the dependencies of the different parameters, select subsets in the measurements, and extrapolate the measurements via simulation. Available aggregated information is directly displayed and new aggregate information, produced in the background, is displayed as soon as it is available.

1 Introduction

Our ability to regulate and manage the traffic on our road-infrastructure, essential for the economic welfare of a country, relies on an accurate understanding of the dynamics of such system. Recent studies have shown very complex structures in the traffic flow [1], [2]. This state is called the synchronized state, which has to be distinguished from a free flowing and congested state. The difficulty to understand the dynamics originates from the difficulty to relate the observed dynamics in speed and density to the underlying dynamics of the drivers behaviors, and the changes therein as function of the circumstances and driver motivation [3].

Simulations play an essential role in evaluating different aspects of the dynamics of traffic systems. As in most application areas, the available computing power is the determining factor with respect to the level of detail that can be simulated [4] and, consequently, lack of it leads to more abstract models [5]. To be able to afford more detailed situations, we looked how we could use the resources provided by for instance Condor[6], or the Grid [7].

Simulation and real world experimentation both generate huge amount of data. Much of the effort in the computer sciences groups is directed into giving scientists smooth access to storage and visualization resources; the so called middle-ware on top of the grid-technology. Yet, for a scientist seamless integration of the information from simulated data and measurements is the most important issue, the so called data-driven approach (see for instance [8]).

M. Bubak et al. (Eds.): ICCS 2004, LNCS 3038, pp. 771–778, 2004.

Our department participated in the Grid-based Virtual Laboratory Amsterdam (VLAM-G) [9]. VLAM-G had as goal to hide resource access details from scientific users, and to allow scientific programmers to build scientific portals. These portals give access to the user interfaces for scientific studies: combinations of information gathering, processing, visualization, interpretation and documentation. Typical applications can be found in Astronomy, Earth Observation and High Energy Physics, Medical Diagnosis and Imaging, Food- and Bio-Informatics, as bundled in the 'Virtual Laboratory for e-science' [10].

In this article we show our experience with building our Virtual Traffic Laboratory as a data driven experimentation environment. This experience can be used as input for the future development of the Virtual Laboratory on other application domains.

2 VLAM-G Architecture

The Scientist is the person that actually is performing the studies. In a study often the same steps are repeated, as for instance testing a hypothesis on a certain dataset. Some steps can be quite time-consuming, so the Scientist can log-out from this study, prepare another study, and come back to inspect the intermediate results and perform another step of the study.

So, when the Scientist starts working with VLAM-G, there is support in the form of the information management system VIMCO and the run time system RTS [11]. VIMCO archives study, module and experiment descriptions [12], together with the application specific databases. The RTS takes care of scheduling, instantiating and monitoring the computational modules of an experiment. It makes thereby extensive use of Globus services, the actual standard in Grid computing.

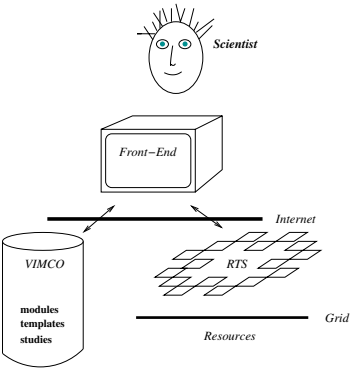


Fig. 1. The different systems for a study

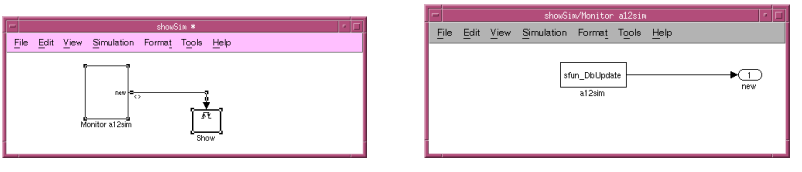
3 The Front-End

The Scientist can use the front-end that is optimal for its domain. For complex system engineering, as traffic systems, we favor the Matlab environment. So, we have coupled a prototype of the RTS [13] with the Matlab environment. Here the RTS is used for the heavy computational tasks, while the Matlab environment is used for analysis and visualization of the results.

To be able to demonstrate the possibilities of Matlab as front-end, we have implemented a gateway routine, which allows the user to load VLAM-G modules, couple them, configure them and start them. We have hidden our gateway routine inside a user-defined block of Simulink. Simulink is an interactive, graphical, tool for modeling, simulating, and analyzing dynamic systems.

Further we used the Condor system to start up a cluster of jobs on several resources, including the Globus-resources at our site. The Simulink system was used to monitor the progress of the jobs, by monitoring the database where the results of the analysis and simulation are stored.

In the following figure one can see an example of such Simulink model. The first block generates a trigger when new data is available in the database, the second block then queries the database and updates a figure with the new data.



(a) top level (b) bottom level

Fig. 2. The ShowSim Monitor in Simulink

Simulink is a part of the Matlab suite. It has an extensive library of predefined blocks, which perform operations on their input and generate output. In that sense they are comparable with the modules of VLAM-G. The difference is that these operations are performed as threads of the current Matlab process, on the current Machine, while at VLAM-G the modules are processes on remote resources. The VLAM-G concept of re-usable modules is perfect for the initial processing, analyzing and cleaning of the data, while the library of functions that Simulink has to offer is perfect to perform some final filtering of the results before they are visualized to the Scientist.

4 The Application

Traffic flow on the Dutch highway A12 is investigated for a wide variety of circumstances in the years 1999-2001. The location was especially selected for the absence of disturbing effects like nearby curvature, entrees or exits. This

location has the unique characteristic that, although the flow of traffic is high, traffic jams are very sporadically seen. In this sense it is a unique measurement point to gather experimental facts to understand the microscopic structures in synchronized traffic states ([2]), which was not reported outside of Germany yet.

Previous research ([1]) has shown that three types of synchronized traffic can be distinguished:

i stationary and homogeneous states, where the average speed and flow are nearly constant for several minutes
ii stationary, non-homogeneous states, where the average speed is constant for a lane, but the flow noticeably changes.
iii non-stationary and non-homogeneous states, where both average speed and flow change abruptly.

In addition, it is found that transitions from these states to free flowing traffic are rare, but between synchronized states are frequent.

This means that for understanding the microscopic structures in synchronized traffic states the relations between several aggregates of single vehicle measurements have to be made. Important aggregate measurements are for instance average speed, average flow, average density, headway distribution and speed difference distribution. The dynamics of these one-minute aggregates over 5-10 minutes periods are important for a correct identification of the state.

To facilitate the analysis of aggregate measurements over time we designed the following architecture:

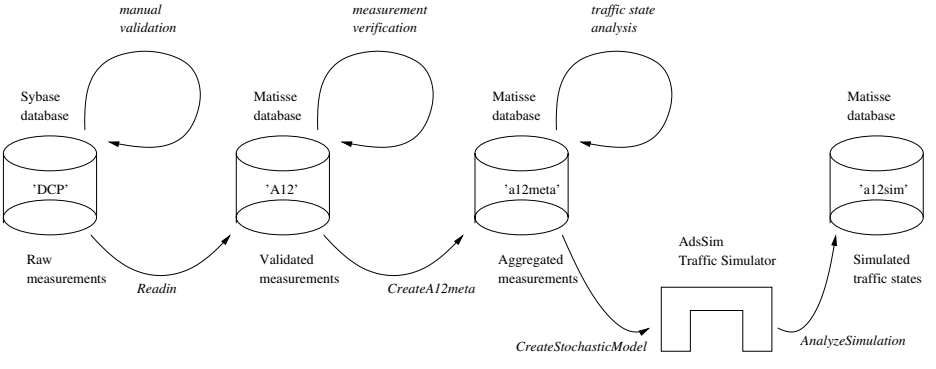


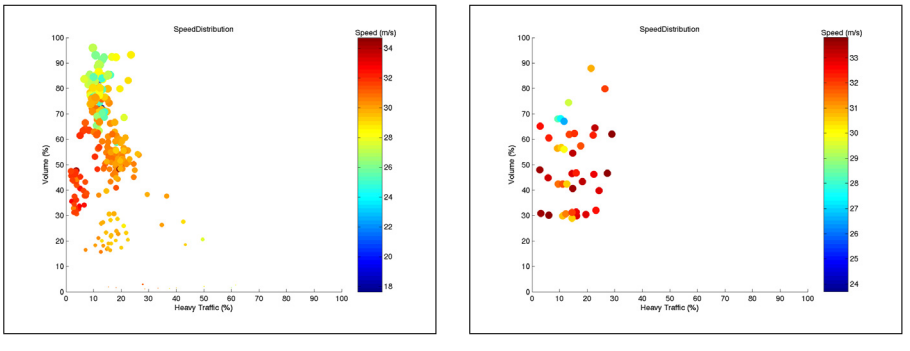
Fig. 3. The measurement analysis architecture

Along the A12 there was a relational database from Sybase that collected the measurements from two independent measurement systems. One system was based on inductive loops in the road, the other on an optical system on a gantry above the road. Although both were quality systems, some discrepancies occur between measurements due to different physical principles. Video recordings were used to manually decide the ground truth when the measurements were not clear.

After this validation process, the measurements were converted to an object oriented database from Matisse. This database was used to verify the quality of the measurement systems themselves. While the manual validation process was used to get the overall statistics of errors in the measurements, the object oriented database was used to analyze the circumstances of the measurement errors. Several hypothesis of underlying failure processes were raised, as for instance the characteristics of trailers that had higher changes to be characterized as an independent passage.

The validated measurements were used to generate the statistics that characterize the traffic flow. Different measurements-periods could be combined based on different criteria, for instance 'Flowing'/'Congestion', 'High-Density'/'Low-Density', weather conditions, fraction of heavy traffic, etc, etc. The right combination of criteria results in candidate traffic flow states. The statistics that are important to characterize the microscopic structure of the traffic flow are not only averages and standard deviations, but require for non-Gaussian distributions fits of complex probability density functions. Generating such statistics typically takes 20 minutes on a UltraSPARC-II workstation, which makes it worthwhile to store the results a separate database.

An example of such analysis is given in figure 4, where the average speed is given as a function of the flow (as percentage of the maximum flow) and the fraction of lorries (as percentage of the number of passages).



(a) measured (b) simulated

Fig. 4. The average speed as function of the flow and the fraction heavy traffic

The average speed is indicated with a colorcode, red (top of the bar) indicates high speeds, blue (bottom of the bar) indicates low speeds. Each point indicates an aggregate over longer period (30-60 minutes), which are typically equivalent with a few thousand passages.

Combinations of measurement-periods that showed the same patterns on their aggregated traffic flow measurements over time were candidate traffic flow states. These aggregated measurements could be translated into the parameters

of a microscopic traffic simulator, AdsSim [14], which is based on the microscopic Mixic model [15].

The characteristics of the simulated data were aggregated in the same way as the real data, and the resulting dynamics were compared to the original dynamics, to see if the model was complete (see figure 4). As one can see, the simulated points (each representing 5000 passages) are more homogeneous spread over the spectrum because one can ask for certain combination of parameters. Yet, the results are less to be trusted when one has to extrapolate far from actual measured parameter-combinations space. For instance, the average speed is unexpectedly high for Heavy Traffic percentages above 30%. This is due to the fact that this type traffic was only seen for low Volumes, when the fast lane is only sporadically used (with high speeds). When the flow increases, the fast lane is used more often, which gives this lane more influence on the average speed, and the speed-characteristics of this flow are extrapolated from the sporadic passages at low Volumes.

The *CreateA12meta* and *CreateStochasticModel* were originally Matlab-functions with more than 1000 lines of code. We converted those functions to standalone programs, which made it possible to run those functions in the background with the aid of the Condor software[6]. The latest versions of this software even make it possible to add Grid-resources to the pool with the *glide-in* technique [16].

The major advantage of our approach was to store all these meta-data in separate databases. This made it possible to start from the Matlab commandline a number of daemons, implemented in Simulink, which constantly monitor those databases and update the diagrams when new analysis results are ready.

5 Discussion

We have chosen this application, because of the complexity of both the measurement analysis and the traffic flow model. For instance, the Mixic model has 68 parameters in its traffic flow model [15], and most parameters are described as functions of single vehicle data such as lane, speed and headway. For AdsSim this resulted in 585 variables that can be adjusted to a specific traffic condition. Compare this with the 150 keywords in the standard application in molecular dynamics [17] in the UniCore environment [18].

To be able to calibrate such a model for a certain traffic state, the Scientist needs to be able to select characteristic subsets in the bulk of measurements, and visualize the dynamics of the aggregates in different ways. It is no problem that it takes some time to generate aggregates, as long as the Scientist is able to switch fast between diagrams of parameters and their dependencies as soon as the aggregates are ready. Storing the analysis results in a database solves this problem.

Typically, the Scientist can concentrate on a few diagrams (say three) at a time. The Scientist sees a certain pattern in the dependency, and can select

measurements and simulation to add extra points to the diagram, to look if the pattern holds.

While processes at the background fill in the missing data-points, the Scientist starts the visualization of other dependencies, till an unexpected pattern appears. At that moment other subsets in the measurements become important. This means that new analysis has started up, and the decision has to be made to stop the current analysis.

In most cases this decision is negative, because the analysis of the previous characteristic is often quite far, and the Scientist wants to be sure how the unexpected pattern looks for that complete subset, even as there is a subset identified that should show the effect more clearly.

The key of our approach is that we don't see the Scientist as a user that repeats the same analysis repeatedly on the different datasets, but is an expert analyzing the problem from different viewpoints. These viewpoints are not known beforehand, and slowly shifting. The Matlab environment allows full control of a dataset, and facilitates different ways to search, fit and display dependencies. At the moment an interesting viewpoint is found, additional datapoints can be generated in the background, with an interface a high throughput system like Condor. The results are automatically displayed by monitoring the databases with meta-data via Simulink.

This approach differs from the approach of for instance [8], where Matlab is seen as inappropriate due to license policies and speed issues. By using high throughput systems in the background speed is no longer an issue. With its license the Matlab environment provides the Scientist directly a rich set of graphics and solvers, without the need to construct this functionality from home-made modules. Yet, both approaches do not exclude each other. In our view the programming effort can concentrate on often used algorithms, and optimize these algorithms into modules that can be executed in the background, while Matlab is used for direct analysis and visualization.

6 Conclusions

In this article we have introduced an experiment for the Virtual Traffic Laboratory. To aid the scientist, analysis results are stored in databases with aggregated data. This allows to repeatedly display the results from different viewpoints, where the scientist does not have to worry that too rigorous filtering will force him to do the aggregation again.

New aggregate data can be generated by exploring a dependency by performing new analysis on sets selected on different parameter-combinations in the background. This analysis can be performed seamlessly on both real data and simulated data. New data can be automatically displayed by adding monitors to the databases.

References

1. B.S. Kerner and H. Rehborn, "Experimental properties of complexity in traffic flow", Physical Review E, Vol. 53, No. 5, May 1996.
2. L. Neubert, et al., "Single-vehicle data of highway traffic: A statistical analysis", Physical Review E, Vol. 60, No. 6, December 1999.
3. Th. Jörgensohn, M. Irmscher, H.-P. Willumeit, "Modelling Human Behaviour, a Must for Human Centred Automation in Transport Systems?", Proc. BASYS 2000, Berlin, Germany, September 27-29, 2000.
4. K. Nagel, M.Rickert, "Dynamic traffic assignment on parallel computers in TRANSIMS", in: Future Generation Computer Systems, vol. 17, 2001, pp.637-648.
5. A. Visser et al. "An hierarchical view on modelling the reliability of a DSRC-link for ETC applications", IEEE Transactions on Intelligent Transportation Systems, Vol. 3: No. 2, June 2002.
6. Douglas Thain, Todd Tannenbaum, and Miron Livny, "Condor and the Grid", in Grid Computing: Making The Global Infrastructure a Reality, John Wiley, 2003. ISBN: 0-470-85319-0
7. I. Foster, C. Kesselman, "The Grid: Blueprint for a New Computing Infrastructure", Morgan Kaufmann, 1999.
8. Craig C. Douglas, "Virtual Telemetry for Dynamic Data-Driven Application Simulations"; Proc. ICCS 2003, Melbourne, Australia, Lecture Notes in Computer Science 2657. p. 279-288, Springer-Verlag 2003.
9. H. Afsarmanesh, et al. "VLAM: A Grid-Based virtual laboratory", Scientific Programming (IOS Press), Special Issue on Grid Computing, Vol. 10, No. 2, p. 173-181, 2002.
10. http://www.vl-e.nl
11. A. Belloum et al. "The VL Abstract Machine: A Data and Process Handling System on the Grid". Proc. HPCN Europe 2001, 2001
12. E. C. Kaletas, H. Afsarmanesh, and L. O. Hertzberger. "Modelling Multi-Disciplinary Scientific Experiments and Information". In Proc. ISCIS'03, 2003.
13. B. van Halderen, "Virtual Laboratory Abstract Machine Model for Module Writers", Internal Design Document, July 2000.
    See $http://www.dutchgrid.nl/VLAM-G/colla/proto/berry\_running/$
14. A. Visser et al.
15. Tampére, C. and Vlist, M. van der, "A Random Traffic Generator for Microscopic Simulation", Proceedings 78th TRB Annual Meeting, Januari 1999, Washington DC, USA.
16. James Frey et al. "Condor-G: A Computation Management Agent for Multi-Institutional Grids", Proceedings of the Tenth IEEE Symposium on High Performance Distributed Computing (HPDC10) San Francisco, California, August 7-9, 2001.
17. W. Andreoni and A. Curioni "New Advances in Chemistry and Material Science with CPMD and Parallel Computing", Parallel Computing 26, p. 819, 2000 .
18. Dietmar W. Erwin and David F. Snelling, "UNICORE: A Grid Computing Environment", in Lecture Notes in Computer Science 2150, p. 825-839, Springer-Verlag Berlin Heidelberg, 2001.