Organic traffic light control for urban road networks

Holger Prothmann*, Jürgen Branke and Hartmut Schmeck

Institute AIFB,
Karlsruhe Institute of Technology,
Universität Karlsruhe (TH),
76128 Karlsruhe, Germany
E-mail: prothmann@aifb.uni-karlsruhe.de
E-mail: branke@aifb.uni-karlsruhe.de
E-mail: schmeck@aifb.uni-karlsruhe.de
*Corresponding author

Sven Tomforde, Fabian Rochner,
Jörg Hähner and Christian Müller-Schloer

Institute of Systems Engineering,
Leibniz Universität Hannover,
Appelstr. 4,
30167 Hannover, Germany
E-mail: tomforde@sra.uni-hannover.de
E-mail: rochner@sra.uni-hannover.de
E-mail: haehner@sra.uni-hannover.de
E-mail: cms@sra.uni-hannover.de

Abstract: In recent years, autonomic and organic computing have become areas of active research in the informatics community. Both initiatives aim at handling the growing complexity in technical systems by focusing on adaptation and self-optimisation capabilities. A promising application for organic concepts is the control of road traffic signals in urban areas. This article presents an organic approach to traffic light control in urban areas that exhibits adaptation and learning capabilities, allowing traffic lights to autonomously react on changing traffic conditions. A coordination mechanism for neighbouring traffic lights is presented that relies solely on locally available traffic data and communication among neighbouring intersections, resulting in a distributed and self-organising traffic system for urban areas. The organic system’s efficiency is demonstrated in a simulation-based evaluation.

Keywords: adaptive traffic control; learning traffic light controller; observer/controller architecture; organic computing.


Copyright © 2009 Inderscience Enterprises Ltd.
Biographical notes: Holger Prothmann is a PhD candidate at the University of Karlsruhe, Germany. He received his Diploma in Computer Science from the University of Dortmund, Germany in 2004. His research focuses on learning and optimisation mechanisms for self-organising traffic systems.

Jürgen Branke currently holds the position of a Research Associate at the University of Karlsruhe, Germany. He has been active in the area of nature-inspired optimisation since 1994, and is a leading expert on optimisation in the presence of uncertainties, including noisy or dynamically changing environments. Further research interests include organic computing, complex system optimisation, multi-objective optimisation, robustness of solutions and parallelisation.

Hartmut Schmeck is a Full Professor of Applied Informatics at the University of Karlsruhe within the Karlsruhe Institute of Technology – KIT (Germany). His current major research interest is on self-organisation and adaptivity in complex technical systems. He is a Key Member of the Organic Computing Initiative and Coordinator of the German Priority Programme on Organic Computing. At the KIT, he is the Scientific Spokesperson of the newly formed KIT-Focus Area COMMputation addressing the inherent combination of communication and computation which is a characteristic feature of smart application systems.

Sven Tomforde is a PhD candidate at the Leibniz University Hannover, Germany, where he also received his M.Sc. in Computer Science in 2007. His current work focuses on distributed and self-organised collaboration mechanisms in traffic control systems.

Fabian Rochner received his Diploma in Civil Engineering at the Leibniz University Hannover, Germany in 2001. Afterwards, he worked as a Research Associate covering the subject of the application of self-organised learning to problems of traffic control. Recently, he left the university for a job as a civil engineer but keeps in touch with the research programme.

Jörg Hähner received his Diploma in Computer Science from the Darmstadt University of Technology, Germany in 2001 and the ‘Dr. rer. nat.’ degree in Computer Science from the University of Stuttgart, Germany in 2006. He worked in the area of data management in mobile ad-hoc networks and was appointed as an Assistant Professor in the System and Computer Architecture Group at Leibniz University Hannover, Germany in 2006. His research focuses on architectures and algorithms in the field of organic computing (e.g. distributed smart camera systems, mobile ad-hoc and sensor networks and global scale peer-to-peer systems).

Christian Müller-Schloer studied Electrical Engineering at the Technical University of Munich. From 1977 to 1990, he was a Member of Siemens Corporate Technology in Munich, Germany and Princeton, NJ. In 1991, he was appointed as a Full Professor of Computer Architecture and Operating Systems at the Leibniz University Hannover. He is one of the Founders of the German Organic Computing Initiative and Co-initiator of the Priority Programme on Organic Computing of the German Research Foundation. His present projects – predominantly in the area of organic computing – deal with quantitative emergence and self-organisation, organic traffic control, self-organising smart camera systems and ontology-based self-organising embedded systems.
1 Introduction

Autonomic (Kephart and Chess, 2003) and organic computing (Schmeck, 2005) aim at handling the growing complexity in today’s technical systems. Their focus is on principles that enable the creation of systems with ‘life-like’ properties that are capable of adapting to changing environments and handling unforeseen situations. Autonomic and organic systems exhibit self-x properties including self-configuration, self-optimisation, self-protection or self-healing capabilities. While autonomic computing has a strong focus on server architectures, organic computing investigates self-organising technical systems in general.

Urban traffic networks are one promising application domain for organic computing since traffic congestion in urban areas is a problem with a huge environmental and economical impact. For the USA, the Urban Mobility Report (Schrank and Lomax, 2007) calculates that in 2005 urban Americans had to spend an extra of 4.2 billion hours in traffic due to congestion, purchasing additional fuel for a total of $78 billion, which is an increase of 220 million hours or $5 billion compared to the previous year. In many cities, these rising demands cannot be counteracted by further extending the existing road infrastructure giving a special importance to the efficient use of the existing network. In this respect, traffic lights are a vital factor since good control strategies are often capable of improving the network-wide traffic flows. The environmental and economic importance of traffic control systems combined with the distributed nature of traffic nodes and their constantly changing traffic demands make traffic light control an ideal test case for organic computing approaches.

In the remainder of this article, a decentralised approach to organic traffic control (OTC) is presented. Section 2 briefly reviews existing concepts for the control of traffic lights and their coordination and investigates (evolutionary) optimisation approaches for the control problem. Section 3 introduces the adaptive and learning OTC approach for intersection control. Results comparing a ‘conventional’ intersection controller and the OTC version are presented in Section 4 and show the benefits of the OTC approach. Section 5 extends the OTC approach with a distributed mechanism that establishes a progressive signal system (PSS) dynamically. The mechanism relies solely on locally available traffic data and communication among neighbouring nodes, thereby eliminating the need for any central component in the architecture. The resulting adaptive, distributed control system is tested in a simulated traffic network using a microscopic traffic simulator from the field of traffic engineering. Results of these tests are presented in Section 6. Section 7 concludes with a summary of the presented concepts and results and gives an outlook.

2 State of the art

The control of traffic systems is a complex task due to the dynamic nature of traffic and the large number of possible objectives that may interact and conflict. This complexity has early made traffic control a research focus in different fields of science (including civil engineering, physics and informatics) with a still growing interest in innovative traffic solutions today. Section 2.1 introduces basic concepts for traffic light control and presents some important centralised traffic control systems commonly used in the field today. As this article deals with a self-organising organic approach that makes use of
evolutionary optimisation, Section 2.2 discusses the use of evolutionary algorithms for traffic light optimisation and Section 2.3 presents recently developed self-organising traffic control approaches.

2.1 Traffic control

A signalised intersection can be controlled by a simple fixed-time controller (FTC) or by a more complex traffic-responsive variant. In both cases, the intersection's turning movements are grouped into signal phases that obtain the right way in a reoccurring sequence. The duration of this phase sequence (that might include additional interphases to allow intersection clearing in-between phases) defines the cycle time of the controller. For FTCs, the duration of phases and their sequence are fixed, resulting in a constant cycle time. Traffic-responsive controllers can vary the phase durations and their sequence based on the number of waiting vehicles or on gaps in the approaching traffic.

Since in urban traffic networks, neighbouring intersections are often located closely, the coordination of traffic lights can have a beneficial effect on the traffic flows. A prerequisite for coordinated traffic lights (that form a PSS) is a common cycle time of the participating nodes. Furthermore, each participating node needs to determine a synchronised phase that always starts at a fixed point of the cycle. By defining appropriate offsets for neighbouring nodes that take into account the travel time in between the nodes, a PSS can be established that enables vehicles to pass several intersections without having to stop due to a red light. While FTCs can be coordinated easily due to their fixed phase durations and cycle times, the task is more complex for traffic-responsive controller variants where additional prerequisites have to be met to allow a coordinated operation.

Today, traffic lights in urban areas are usually operated by a traffic control center. Several centralised systems have been developed for this task, including Split, Cycle and Offset Optimisation Technique (SCOOT), Sydney Coordinated Adaptive Traffic System (SCATS) and Traffic-responsive Urban Control (TUC): SCOOT (Robertson and Bretherton, 1991) is used in more than 200 installations worldwide. The system computes a single cycle time for all intersections, splits this cycle time into green times for each intersection and then adjusts offset times in order to minimise waiting times. For this adjustment, the network is split into subnetworks, and a dynamic traffic model is used.

In the SCATS architecture (Sims and Dobinson, 1980), regional computers are used for strategic control of local traffic controllers. The local controllers are responsible for data collection and tactical control. SCATS relies on a library of controls which are selected according to traffic conditions. The optimisation criterion can depend on the current traffic state, i.e. the system might minimise the number of stops during the night and maximise throughput at day time.

A more recent development is TUC (Diakaki et al., 2003), focusing on traffic-responsive coordinated signal control of large-scale urban networks. TUC tries to establish PSSs by adapting splits, cycle times and offsets and can take into account public transport priorities. A cycle control mechanism is used to adjust the cycle time to the maximum saturation level while a decentralised offset control algorithm coordinates the main phases of successive nodes. The control decisions of these two components serve as input for split control that aims at minimising the vehicle queues at the intersection. The resulting signal plans are finally adapted by a public transport priority module that provides priority to public transport vehicles by applying a rule-based algorithm.
2.2 Evolutionary optimisation of traffic lights

Evolutionary algorithms are randomised optimisation heuristics that mimic biological evolution to tackle optimisation problems. Their general scheme is simple: starting with a set (called population) of randomly generated initial solutions, an evolutionary algorithm selects solutions with a relatively high quality from its population as parents, which are then combined and locally modified by crossover and mutation operators to form new offspring solutions. Based on their quality, some of the parents and offspring are selected to form the next generation of solutions that replaces the old population. This process is repeated until a stopping criterion (usually a maximum number of generations, a time limit or some quality level) is reached. Selection, crossover and mutation are randomised operations, but good solutions have a higher probability to survive and generate offspring. Therefore, the overall quality of solutions is likely to improve over time while the random influence of mutation helps to prevent premature convergence on some local optimum.

Due to their simple working principle and the fact that evolutionary algorithms are black box algorithms that can be applied to any problem where a quality (or fitness) can be assigned to a solution, evolutionary algorithms are widely used in many real world optimisation problems. They also have been applied in the optimisation of traffic light controllers (TLCs), some recent results are presented in the remainder of this section.

Stevanovic, Martin and Stevanovic (2007) used evolutionary algorithms to optimise the traffic lights of an arterial road consisting of 12 intersections in Park City, USA. They optimised cycle length, offsets, phase sequences and green splits of the networks’ intersections, trying to minimise their performance index that combines delay and the resulting number of stops into a single objective. The controller considered in their work was a traffic-responsive NEMA controller (National Electrical Manufacturers Association, 2003) that is common in the USA. Solutions discovered by this approach outperformed timing plans found by SYNCHRO – a traditional optimisation tool – by at least 8%.

Sun, Benekohal and Waller (2003) and Branke, Goldate and Prothmann (2007) investigated the use of multi-objective evolutionary algorithms that treat delays and number of stops as separate criteria. Sun, Benekohal and Waller used NSGA-II – a multi-objective evolutionary algorithm – to minimise delay times and the resulting number of stops for a two-phase isolated intersection controlled by a FTC. Approximation formulas by Webster and Akçelik served as objective functions in their experiments. Branke, Goldate and Prothmann (2007) implemented NSGA-II for the optimisation of an isolated intersection at Karlsruhe, Germany, that was equipped with a traffic-responsive VS-Plus controller (Swiss Verkehrs-Systeme AG, 2008). Again, delay time and number of stops served as objectives, but controller settings were evaluated with the help of a microscopic traffic simulation software. Solutions found by NSGA-II outperformed a reference solution provided by a traffic engineer with respect to the considered objectives.

In all mentioned references, evolutionary algorithms have been used for the off-line optimisation of TLC settings, i.e. the controller parameters are optimised before they are applied, but no further online optimisations take place when the parameters are used in the traffic system. Therefore, the parameters’ quality runs the risk of being decreased over time due to changing traffic demands. To avoid this problem, parameters can be adapted online, but the online usage of evolutionary algorithms is challenging due to their...
run-time requirements. Braun et al. (2008) investigated the use of evolutionary algorithms for network-wide online optimisations. An evolutionary algorithm is used for the optimisation of a frame signal plan that specifies the network-wide cycle time as well as intersection specific offsets, phase sequences, and time frames bounding possible phase endings. Based on the frame signal plan, local traffic-responsive controllers can adapt the green times at each intersection within the specified time frames. Optimisations aim at minimising a single-objective problem that aggregates the delay at all intersections. Evolved frame signal plans are evaluated using the traffic flow model of the traffic control system BALANCE which represents the network’s traffic demands online.

The approach has been evaluated in a field test at Ingolstadt, Germany, which includes 46 intersections within the city’s main road network. The intersections are grouped into three sub-networks for which frame signal plans are separately optimised. Frame signal plans evolved by the evolutionary algorithm were compared to a basic scenario having only local actuated control and to frame signal plans optimised by a hill climber that is part of BALANCE. Using the evolutionary algorithm, delays could be reduced by 21% compared to the basic scenario and by 10% compared to the hill climber. The number of stops were reduced by 17% and 8%, respectively.

While the approach presented in Braun et al. (2008) relies on a centralised optimisation component, the organic approach presented here combines evolutionary algorithms and Learning Classifier Systems – which are rule-based learning systems – to create a decentralised online system with optimisation capabilities.

2.3 Self-organising approaches

Helbing, Lämmer and Lebacque (2005) developed a fluid-dynamic model for the simulation of traffic networks. Based on this model, a self-organising control principle for traffic lights is proposed. Cars waiting to be served generate a ‘pressure’ on the traffic light that depends on the number of waiting cars. Simultaneously, cars blocking subsequent road sections create a ‘counterpressure’ when green times cannot be used effectively in the current situation. The active traffic lights for the next time period are selected based on the current pressures and counterpressures resulting in a dynamic composition of turning movements for each time period. In contrast to the approach presented here, Helbing, Lämmer and Lebacque do not establish an explicit synchronisation of neighbouring intersections and therefore do not rely on a communication mechanism. Intersections are only loosely linked by monitoring their connecting road segments. According to Helbing, Lämmer and Lebacque, this loose coupling is sufficient to dynamically create progressive signals.

Despite being decentralised and adaptive, the approach of Helbing, Lämmer and Lebacque has drawbacks. The dynamic composition of turning movements for each time period might lead to acceptance problems for road users since the system behaviour cannot be easily understood from their point of view. Extensions are necessary to incorporate legal restrictions that are imposed on traffic lights (like preventing conflicts during phase changes). Furthermore, in its current form, the model cannot handle commonly used ‘qualified conflicting traffic streams’ (e.g. traffic going straight ahead cannot be combined with left-turning traffic from the opposite direction in the same period).

Another approach towards self-organising traffic lights (SOTL) is presented by Gershenson (2007). Similar to Helbing, Lämmer and Lebacque, Gershenson’s SOTL
approach does not explicitly establish a PSS, but relies on traffic-responsive local controllers that take into account the number of waiting cars or the gaps between arriving vehicles, thereby being similar to uncoordinated NEMA controllers (National Electrical Manufacturers Association, 2003). Traffic lights keep a count $\kappa$ of the number of cars waiting in front of them, each car being weighted by its waiting time. As soon as $\kappa$ reaches a threshold, the traffic light changes. Several restrictions to avoid fast switching of traffic lights, the interruption of moving platoons and deadlocks caused by long platoons are implemented in the SOTL control method. Although there is no explicit coordination, Gershenson describes the observation of coordination effects similar to those being achieved by PSSs.

In Bazzan’s (2005) approach to distributed traffic signal coordination, intersections are modelled as individually-motivated agents. Each agent possesses a set of predefined control strategies to choose from. The selection process is based on local events occurring at the intersection as well as on the results of ‘coordination games’ that are played among neighbouring agents. The principal applicability of the approach is demonstrated in a simple scenario of an arterial road consisting of ten intersections. Each intersection agent has to choose between two strategies, each of which favours one of the two arterial directions over the other. The distributed approach is compared to a central controller that creates synchronised traffic lights in one of the arterial directions based on detector readings from the network. The agent-based approach proves to be better in situations where the flow of traffic in the different directions is nearly equal. An important difference between Bazzan’s approach and the organic system presented here is that Bazzan relies solely on the selection of pre-determined strategies while in the approach presented here, new strategies are generated dynamically.

3 An organic architecture for intersection control

This section presents the OTC architecture for the control of signalised intersections. An industry-standard TLC – the system under observation and control (SuOC) in terms of organic computing – is extended by an observer/controller component that reconfigures the TLC depending on current traffic conditions. The resulting self-optimising architecture – which is an implementation of the generic observer/controller architecture presented in Branke et al. (2006) – is depicted in Figure 1 and explained in the remainder of this section.

System under observation and control. The SuOC consists of a parametrisable TLC responsible for physically setting the intersection’s traffic lights. Different industry-standard TLCS may be implemented in the SuOC, the only precondition being that the controller is parametrisable, i.e. that its behaviour can be specified by a set of parameters which can be varied by the observer/controller. Possible controllers include simple FTC or more complex traffic-responsive variants like VS-Plus (Swiss Verkehrs-Systeme AG, 2008) or NEMA controllers (National Electrical Manufacturers Association, 2003). A good setup of the TLC’s parameters that matches the current traffic conditions has an important influence on the resulting delay times and number of stops for these systems. In the OTC architecture, the parameter setup is optimised online by the observer/controller component.
Figure 1  The organic traffic control architecture for traffic light control (see online version for colours)

Observer/controller component. The observer/controller component can be split into two separate layers according to the different tasks it performs. Layer 1 is responsible for the online selection of TLC parameters depending on local traffic conditions. An observer component monitors the traffic flows crossing the intersection and combines the determined flow values into a vector representing the local traffic situation. For an intersection with $n$ turnings, the observer produces an $n$-dimensional real-valued vector containing the traffic flows measured in vehicles per hour (veh/h) for each of the intersection’s turnings. This vector is then provided as input for a LCS that is responsible for selecting appropriate traffic light parameters for the observed situation.

A LCS (Butz, 2005) is a rule-based reinforcement learning system that aims at learning the best-rewarded response to any input it gets. The core component of a LCS is its rule base, where each rule (or classifier) consists of a condition, an action and an evaluation part. The condition specifies for which inputs the classifier is applicable and the evaluation part provides a reward the LCS predicts to receive when applying the classifier’s action under the specified conditions. The selection of an appropriate action for an input based on the classifiers stored in the rule base is a two-step process: From all classifiers, a subset called ‘match set’ is built containing all applicable (or ‘matching’) classifiers for the current input. For all distinct actions in the match set, the average evaluation of all classifiers advocating that action is computed. The action with the best
evaluation is selected for execution and all classifiers in the match set advocating that action form the ‘action set’. The reward subsequently received from the environment is used to update the evaluation of all classifiers in the action set.

In the OTC architecture, a modified real-valued variant of Wilson’s XCS (Wilson, 1995) is used as classifier system. The current traffic situation in form of an \( n \)-dimensional vector containing the intersection’s current vehicle flows is used as input for the LCS. The condition part of a classifier accordingly consists of \( n \) interval predicates forming an \( n \)-dimensional hyper-rectangle containing all traffic situations matched by the classifier. The action of a classifier consists of a parameter set for the TLC in the SuOC. For a given traffic situation, the LCS determines all matching classifiers, selects the TLC parameters that are predicted to be most appropriate for the current situation and applies these parameters in the SuOC. Based on the resulting performance (e.g. based on the resulting delays or queue lengths) of the TLC running the selected parameters, the LCS receives a reward for its selection and updates the evaluation of all classifiers in the action set accordingly. For details on the evaluation update, the reader is referred to Wilson (1995).

The selection process described so far works when the LCS rule base contains at least one classifier matching the input, but the creation of classifiers has not been discussed. In Wilson’s XCS, new classifiers are generated in two different ways: whenever the match set is empty, at least one classifier consisting of a condition matching the current input, a random action and a default evaluation part is inserted into the rule base in a process called ‘covering’. Furthermore, occasionally, some classifiers are selected to be the ‘parent individuals’ for a reproduction cycle. Genetic operators like crossover and mutation are applied to copies of the parents to form offspring which are inserted into the rule base.

Unfortunately, this standard way of creating new classifiers is infeasible for traffic control. Using a stochastic process to create classifiers and evaluating their quality by applying their actions at the intersection without prior tests would in most cases result in an extremely poor system performance. Most of the randomly created classifiers would result in large average delays and long queues, and the LCS would have to learn this from repeated negative experiences gained at the real intersection.

In the OTC architecture, new classifiers are therefore created only on Layer 2 by off-line optimisation (see Figure 1). New classifiers – or more precisely their action parts containing the TLC parameters – are evolved by an evolutionary algorithm that uses a traffic simulation software to evaluate the parameters’ quality with respect to a specific traffic situation. Using this off-line simulation-based approach, optimised classifiers are found and an approximate quality of a classifier is known even if it has not been previously applied at the intersection. Small imprecisions induced by the simulation-based evaluation are corrected online by the LCS when the classifier’s action containing the TLC parameters is applied in the SuOC and its impact is evaluated later on by determining its reward value.

Unfortunately, evolving good parameters based on simulations takes some time while an LCS is expected to react on new traffic situations immediately. If the rule base of the LCS does not contain classifiers matching an observed traffic situation, a classifier located most ‘closely’ to the unmatched situation is selected and its condition is widened as far as necessary to match the situation. This enables an immediate response of the LCS
while on the other hand the situation-dependent quality of TLC parameters remains (somewhat) predictable. For further details on the OTC architecture, the interested reader is referred to Prothmann et al. (2008).

4 Experimental results for intersections

The OTC architecture presented in Section 3 has been evaluated for different three- and four-armed traffic nodes. This section provides details of the experimental setup and presents the obtained results.

4.1 Experimental setup

To perform the experiments, simulation models of existing traffic nodes have been built using the microscopic traffic simulator AIMSUN 5.1 (Barceló et al., 2005). The models (called K3 and K7) are based on maps of intersections located at Hamburg, Germany. They are depicted in Figure 2. While K7 is a three-armed intersection allowing six turning manoeuvres, K3 is four-armed and consists of eleven turnings.

For both nodes, a traffic engineer provided a fixed-time signal programme that is applied as a reference controller in the evaluation. Traffic demands are modelled according to data taken from a traffic census that was conducted by the local authorities. In the census, cars and trucks passing the intersection were counted and documented for each turning with a time resolution of 15 minutes.

Experiments were conducted for a simulated period of six hours starting at 6 am. This period was chosen because it starts with a phase of low traffic density that is quickly replaced by the morning peak hour (lasting approximately from 7.30 to 8.30 am) with high traffic demands. Till noon, traffic settles down to a medium level. The total number of vehicles passing K3 and K7 is depicted in Figure 3.

Figure 2  Simulation models of the intersections K3 and K7 (see online version for colours)
To compare the performance of different TLCs, the intersection’s average delay is used which is the basis for the established level of service classification (Transportation Research Board, 2000) and should be minimised by a TLC. Delays have been measured using the microscopic traffic simulator AIMSUN 5.1 (Barceló et al., 2005), which was used to simulate the SuOCC and to provide a fitness evaluator for the evolutionary algorithm on Layer 2 of the OTC architecture.
The OTC approach was evaluated in three consecutive experiments (labelled Day 1, Day 2 and Day 3). At the beginning of Day 1, the rule base of the LCS was empty. For Days 2 and 3, the rule base that evolved on the previous day(s) was used. Simulations of each day have been repeated at least three times using different random seeds. The evolutionary algorithm optimised cycle length and phase splits for the intersections while using the phase sequence from the reference TLC.

4.2 Simulation results

This section presents results of the simulation study, comparing the average vehicle delay resulting from the reference solution and the OTC approach.

Results for K7. Results of the experiments for K7 are depicted in Figure 4. For Day 1, the OTC approach can quickly improve the average vehicle delay compared to the reference solution for the low traffic period preceding the morning peak. In this period, TLCs found by the evolutionary algorithm can easily outperform the reference solution that was designed to suit higher traffic volumes. During the morning peak, the OTC approach performs slightly better than the reference solution. Due to the quickly rising traffic demand at the intersection, Layer 2 is heavily used during this period and existing classifiers need to be widened frequently since the initially empty rule base does not contain appropriate classifiers. After the morning peak, the OTC approach leads to smaller delays than the reference solution. Overall, the average improvement for Day 1 with respect to the reference solution is about 10%.

For Days 2 and 3, the OTC approach can outperform the reference solution for the whole simulation period. The system has learned appropriate TLC parameters for most traffic situations recognised by the observer, therefore appropriate TLC parameters are often available instantly or existing classifiers need to be widened only to a small extent. The average improvement with respect to the reference solution is about 12%.

Results for K3. Results obtained for intersection K3 are depicted in Figure 5. The results resemble the simulations for K7 presented above. For Day 1, an improvement of about 6% was obtained in comparison to the reference controller despite the initially empty LCS rule base. For Days 2 and 3, the system profits from its populated rule base, handling especially the morning peak better than on Day 1. This results in an average delay reduced by 8% compared to the reference solution for both days.

The simulations indicate that the OTC approach is capable of autonomously improving the performance of signalised intersections by executing a continuous online adaptation of TLC parameters to changing traffic demands. In Section 5, the organic nodes are extended with an additional collaboration mechanism that can further improve the performance of intersections located in urban areas.
Figure 4  Comparison of organic traffic control approach and reference solution for K7

Figure 5  Comparison of organic traffic control approach and reference solution for K3

5 Distributed progressive signal systems

Vehicles travelling in urban areas often have to pass several neighbouring intersections on their journey. Whenever these intersections are not or badly coordinated, this result in a large number of stops and increased travel times. In this section, the organic nodes are extended with a decentralised collaboration mechanism. The mechanism was originally published in Tomforde et al. (Accepted for Publication) and allows the traffic-responsive creation of PSS that can further improve the traffic flows.
5.1 A distributed progressive signal system algorithm

The distributed calculation of PSSs proposed here is a three step process. In a first step, the network nodes determine partners that collaborate to form a PSS. Once the partnerships are established, the collaborating nodes agree on a common cycle time which is a prerequisite for synchronisation. In a third step, the partners select TLC parameters that respect the common cycle time, calculate offsets and finally establish the PSS. In the process, it is assumed that all nodes have synchronised clocks. The three steps are described in detail in the remainder of this section.

First step: determine collaborating nodes. To determine a sequence of traffic nodes that can establish a PSS improving the network’s traffic flows, each node determines which of its local turning movements exhibits the strongest vehicle flow. Based on the common clock available to the nodes, this check can be performed periodically at all nodes at the same point in time. In the following, let node $j$ determine the turning from upstream node $i$ to downstream node $k$ as its strongest turning movement. For node $j$, it should be beneficial to synchronise the (longest) signal phase serving the selected turning from $i$ to $k$ with the respective upstream intersection $i$, thereby creating a synchronised phase. To initiate the partnership, node $j$ informs its desired predecessor $i$ that it would like to be $i$’s successor in a PSS. After all nodes informed their desired predecessor, a local matching takes place. Each node $j$ checks whether it was chosen by its downstream node $k$ as $k$’s desired predecessor. If this is the case, $j$ acknowledges the partnership with $k$. Other nodes that registered with $j$ receive a reject message and no partnership is established with these nodes initially.

Based on the acknowledged partnerships, each node can determine whether it is part of a PSS and which of its neighbours is its predecessor or successor in the system. The first (last) node of a PSS knows its special position since it has no predecessor (no successor) but a successor (a predecessor) and nodes that were not integrated in a PSS did not send or receive any acknowledgements. Nodes that were not integrated in a PSS can repeat the above process with their second most heavily used turning movement and other nodes not participating in a PSS. For all established PSSs, the collaborating nodes know their partners when the first step is completed and can start to negotiate a common cycle time in the second step.

Second step: determine a common cycle time. Due to its influence on the capacity of the nodes, the common cycle time needs to be selected carefully. For a longer cycle, the constant interphase durations that allow the traffic of an ending phase to leave the intersection before the traffic of the following phase enters, make up a smaller percentage of the cycle time which results in an increased capacity of the node. On the other hand, longer cycles increase vehicle delays in undersaturated conditions due to the increased waiting times resulting from longer red periods. Therefore, the common cycle time for the PSS should be long enough to provide all participating nodes with a sufficient capacity while it should be as short as possible to reduce the induced waiting times.

To determine a common cycle time for the PSS that fits these requirements, each node $i$ keeps track of its own desired cycle time ($DCT_i$) and an agreed cycle time ($ACT$) for the PSS.

The desired cycle time $DCT_i$ is the cycle time node $i$ would prefer for the current traffic situation if it was not part of a PSS. It is determined by activating the node’s LCS
for the current traffic situation (see Section 3) and storing the cycle time of the returned TLC as $DCT_i$. Since the LCS’ selection process performs a local optimisation and focuses on short average delays at the intersection, the LCS tends to return TLCs with relatively short (but not too short) cycles for the current traffic situation.

The agreed cycle time $ACT$ is the cycle time the nodes taking part in the PSS agreed on. Since the desired cycle time $DCT_i$ of each node tends to be as short as possible, $ACT$ is selected as the maximum of all DCTs of nodes $i$ in the PSS (i.e. $ACT = \max \{DCT_i\}$). A shorter $ACT$ might reduce the capacity of the most heavily used node more than is acceptable, leading to rising queues in its approaches.

To decentralize the agreed cycle time $ACT$, each node $i$ stores its knowledge on the agreed time locally as $ACT_i$ and takes part in the following ‘echo algorithm’ (cf. Chang (1982)). The first node in the PSS updates its desired cycle time $DCT_1$, sets $ACT_1 = DCT_1$, and sends $ACT_1$ to its successor in the PSS. The succeeding nodes $i, i = 2, \ldots, n$, where $n$ is the last node in the PSS, successively update their desired cycle time $DCT_i$ by activating their LCS, setting

$$ACT_i := \max \{DCT_i, ACT_{i-1}\} = \max_{j \in \{1, \ldots, i\}} \{DCT_j\},$$

and sending $ACT_i$ to the next node in the PSS. This process continues until the last node $n$ of the PSS is reached. By then, $ACT_n$ equals the maximum DCT in the PSS. $ACT_n$ is now propagated back to the beginning of the PSS, such that each node $i$ in the PSS can replace its knowledge on the agreed cycle time by $ACT_n$ (i.e. $ACT_i = ACT_n$ for $i = 1, \ldots, n - 1$). At the end of this process, all nodes in the PSS have agreed on the same $ACT$ that does not reduce the capacity of the most heavily used node more than is acceptable while being as short as possible.

Third step: determine offsets and establish synchronisation. After the nodes that participate in a PSS have been determined and all participants agreed on a common cycle time, appropriate TLCs respecting the ACT can be selected and offsets for the nodes can be calculated.

For the first node in the PSS, no offset restriction exists. An extended LCS selection procedure that can handle cycle time restrictions (see Section 5.2) is used to select a TLC that on the one hand is appropriate for the local traffic situation and that on the other hand respects the cycle time constraint induced by the ACT. For each succeeding node $i, i = 2, \ldots, n$, the offset $o_i$ depends

- on the predecessor’s offset $o_{i-1}$
- on the start $p_{i-1}$ of the synchronised phase within the predecessor’s TLC
- on the time $d_{i-1}$, vehicles need to arrive from the predecessor
- on the start $p_i$ of the synchronised phase within the node’s own TLC
- on the time $q_i$, needed to serve queued vehicles for the synchronised phase.

Furthermore, the absolute time $s$ when the first node activates its selected TLC – the start time for the PSS – must be known to all successors. Again, all necessary information is successively propagated through the PSS from node to node. After the first node communicates the start time $s$, its offset (without loss of generality $o_1 = 0$, i.e. the first node starts the PSS at time $s$) and the start $p_1$ of the synchronised phase in its TLC to its successor, the nodes $i, i = 2, \ldots, n$, successively select a TLC that respects the cycle
constraint induced by the ACT (thereby determining \( p_i \)) and calculate their own offset relative to the first node in the PSS using the formula

\[
o_i = (o_{i-1} + p_{i-1} + d_{i-1,i} - p_i - q_i) \mod \text{ACT}.
\]

Here, it is assumed that the time \( d_{i-1,i} \) is stored locally at each node for all its neighbours \( j \) (one of which is node \( i-1 \)). This assumption is reasonable since \( d_{i-1,i} \) depends on the fixed distance and speed limits between neighbouring nodes, which are usually constant for a network. The value of \( q_i \) is based on the average queue length observed for the synchronised phase. Once the offset calculation is finished, the values for \( s \), \( o_i \) and \( p_i \) are forwarded to the succeeding node in the PSS until the last node is reached and the offset calculation is finished.

To establish the calculated offset at an intersection without inappropriately interfering with the active signalling, a temporary TLC is activated for exactly one cycle after the currently active TLC’s cycle has ended. The temporary TLC is obtained by proportionally adapting the non-interphase durations of the currently active TLC. Its cycle time \( t \) is given by the equation

\[
t = s + o_i - r - c
\]

where \( r \) denotes the remaining duration of the active TLC’s cycle and \( c \) is the current time at the node. If the calculated cycle time \( t \) cannot be realised because it is shorter than the required minimum duration for each phase plus all interphase durations, \( t \) can be redefined as \( t = t + \text{ACT} \). By using \( t \) as cycle time for the temporary TLC, the node’s calculated offset is reached by simply activating the TLC that was selected for operation in the PSS when the temporary TLC’s cycle is finished. When the temporary TLC’s have been replaced at each node, the PSS has been established.

5.2 Integration of distributed progressive signal systems into the organic traffic control architecture

To integrate the distributed progressive signal systems (DPSS) algorithm in the OTC approach presented in Section 3, the LCS needs to be extended to handle an additional cycle time restriction that is necessary when selecting TLCs for use in a PSS. Therefore, the selection procedure of the LCS has to be adapted to handle the additional restriction. Initially, the new selection process tries to consider only those classifiers whose condition matches the current situation and whose cycle time equals the cycle time restriction. If none of the matching rules satisfies the cycle time restriction, rules (closely) matching the traffic situation are modified to fit the cycle time restriction by proportionally adapting the non-interphase durations of their actions. Modified rules are included in the rule set. By this LCS extension, the DPSS algorithm can be integrated into the existing OTC approach resulting in a collaborative traffic control system that is evaluated in the following section.

6 Experimental results for networks

To examine the behaviour of the proposed DPSS mechanism, the performance of uncoordinated OTC nodes (presented in Section 3) was compared to OTC-DPSS nodes.
additionally running the DPSS algorithm (presented in Section 5). This section describes the traffic scenarios used in the conducted simulation study and presents the obtained results.

6.1 Experimental setup

The OTC-DPSS approach was evaluated on two different test networks – an arterial road and a Manhattan network.

Scenario I: arterial road. The arterial road network consists of five intersections that are located in 250 m distance along an arterial road (see Figure 6). The intersections have an identical topology that allows all possible turning manoeuvres. The intersection approaches are one-laned, but the arterial road segments provide an additional side-lane for left-turns. Each intersection is controlled by an OTC system (as presented in Section 3) that dynamically adapts cycle time and phase durations (splits) of a three-phased FTC that is responsible for setting the physical traffic lights at the intersection. Phase I serves traffic leaving the arterial by a left-turn, Phase II handles the arterial traffic and all vehicles turning right to leave the arterial and Phase III serves traffic arriving from the side roads (see Figure 6).

In the experiments, a sequence of different traffic demands with a total duration of four hours was simulated to provide a dynamic environment for the evaluation. While the most heavily used origin/destination (O/D) pair was B-A (see Figure 6 for O/D labels) during the first half of the simulation, the predominant traffic direction was reversed with the beginning of the simulation’s second half. After 25 and 75% of the simulation duration, the traffic demand in the network was increased, but the predominant traffic direction was kept. The detailed traffic demands are listed in Table 1.

Table 1  Traffic demands for Scenario I

<table>
<thead>
<tr>
<th>Traffic demands for O/D pairs (in veh/h)</th>
<th>1st Hour</th>
<th>2nd Hour</th>
<th>3rd Hour</th>
<th>4th Hour</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin A to destination B</td>
<td>250</td>
<td>300</td>
<td>500</td>
<td>650</td>
</tr>
<tr>
<td>Origin B to destination A</td>
<td>500</td>
<td>650</td>
<td>250</td>
<td>300</td>
</tr>
<tr>
<td>Other O/D pairs</td>
<td>10</td>
<td>10</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Total</td>
<td>2,050</td>
<td>2,250</td>
<td>2,050</td>
<td>2,250</td>
</tr>
</tbody>
</table>
Scenario II: Manhattan network. The Manhattan network consists of six intersections that are located in two rows of three intersections each (see Figure 7). Like in the arterial road scenario, the intersections have an identical topology that allows for all possible turning movements. The connecting road segments are one-laned, have a length of 250 m, and provide an additional side-lane for left-turns when approaching a node. Similar to Scenario I, each intersection is controlled by an OTC system controlling a four-phased FTC. Phases I and III serve left-turning vehicles while the Phases II and IV serve vehicles going straight ahead or turning right (see Figure 7).

Scenario II was simulated for a period of three hours. During the first half of the simulation, the most heavily used routes in the network start at origin A and D and end at destination B and C, respectively. These changes for the second half of the simulation, when most of the traffic uses routes from origins F, H and J to destinations E, G and I, respectively. The detailed demands are listed in Table 2.

Comparison of OTC and OTC-DPSS. For both scenarios, the uncoordinated OTC system and the extended OTC-DPSS system including the DPSS algorithm for distributed coordination were compared based on

- the average local delay times at the intersections
- the average travel times for the complete network
- the average number of stops for the complete network.

The setup of the OTC system is identical to the one used in Prothmann et al. (2008). The rule set of the LCS was empty at the beginning of the simulations for Scenario I and contained some learned rules approximately covering the traffic demands for Scenario II. PSSs were established (based on the distributed algorithm presented in Section 5.1) every ten minutes depending on the current traffic situation. All simulations were conducted using the microscopic traffic simulation software AIMSUN 5.1 (Barceló et al., 2005) running on a 3 GHz quad-core machine.
Table 2  Traffic demands for Scenario II

<table>
<thead>
<tr>
<th>Traffic demands for O/D pairs (in veh/h)</th>
<th>1st Half</th>
<th>2nd Half</th>
</tr>
</thead>
<tbody>
<tr>
<td>Origin A to destination B</td>
<td>400</td>
<td>150</td>
</tr>
<tr>
<td>Origin B to destination A</td>
<td>200</td>
<td>150</td>
</tr>
<tr>
<td>Origin C to destination D</td>
<td>200</td>
<td>150</td>
</tr>
<tr>
<td>Origin D to destination C</td>
<td>400</td>
<td>150</td>
</tr>
<tr>
<td>Origin E to destination F</td>
<td>150</td>
<td>200</td>
</tr>
<tr>
<td>Origin F to destination E</td>
<td>150</td>
<td>400</td>
</tr>
<tr>
<td>Origin G to destination H</td>
<td>150</td>
<td>200</td>
</tr>
<tr>
<td>Origin H to destination G</td>
<td>150</td>
<td>400</td>
</tr>
<tr>
<td>Origin I to destination J</td>
<td>150</td>
<td>200</td>
</tr>
<tr>
<td>Origin J to destination I</td>
<td>150</td>
<td>400</td>
</tr>
<tr>
<td>Other O/D pairs</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>Total</td>
<td>2,900</td>
<td>3,200</td>
</tr>
</tbody>
</table>

6.2 Simulation results

The remainder of this section presents the obtained results for Scenarios I and II that are derived from five independent simulation runs.

Scenario I: arterial road. In the arterial road scenario, the PSSs established by OTC-DPSS included all five arterial nodes. For the first half of the simulation, the PSS started at the eastern node (K429) and extended over the inner nodes (K350, K179 and K158) to the western node of the arterial (K143). In the second half, the direction of the PSS was reverted as soon as the changing traffic demand was recognised.

Figure 8 shows the resulting average delays and the average number of stops for the arterial network compared to an uncoordinated OTC system. The vertical lines mark the change of traffic demands in the simulation. The dashed lines indicate an increased traffic volume; the solid line marks a change in the main traffic direction (compare Section 6.1). Compared to the uncoordinated OTC system, the average number of stops was reduced for nearly the complete simulation period. Only after the change of the main traffic direction, the number of stops is periodically slightly increased due to the time OTC-DPSS needs to detect the changed traffic demand and establish a new PSS. For the complete simulation, the total reduction in the average number of stops compared with the uncoordinated OTC nodes is about 15% while the network-wide travel time was only slightly improved.

While Figure 8 presents a network-wide performance graph, this information is not available to the OTC-DPSS nodes. Each node decides on its desired traffic-light settings solely based on local traffic and performance data and adapts the desired settings to the negotiated result of the DPSS algorithm. Figure 9 presents the local average delays for two arterial nodes.

K350 is an inner node of the arterial that benefits from synchronisation through the entire simulation period (see Figure 9a). Through synchronisation, the vehicle platoons arriving from the previous intersection can pass the node without having to stop, thereby reducing the average local delays at the node. The delay charts for K158 and K179 – the other inner nodes – are omitted here, since they are similar to the one presented for K350.
Figure 8  Average travel time and number of stops for arterial network

Figure 9  Average delay for arterial nodes

K350

K429
K429 is the eastern outer node of the network that benefits from synchronisation only in the second half of the simulation (see Figure 9b), when it can synchronise on a predecessor in the established PSS. In the first half of the simulation period, K429 is the start node of the PSS. In this situation, the OTC-DPSS traffic-light settings that need to fit the negotiated cycle time PSS algorithm are worse than the settings selected by an uncoordinated OTC node resulting in increased average delays at the node for this period. Results for K143 – the other outer node of the network – show the same effects and are omitted here.

Scenario II: Manhattan network. In the Manhattan scenario, several parallel PSSs were established by the DPSS algorithm. For the first half of the simulations, the two most heavily used routes from origin A to destination B (passing the nodes K143, K158 and K179) and from origin D to destination C (passing K403, K384 and K357) were served by PSSs. After the traffic demands changed at the beginning of the simulation’s second half, three parallel PSSs were established by the dynamic DPSS algorithm. The PSSs serve the routes from origin F to destination E (including nodes K357 and K143), from origin H to destination G (including nodes K384 and K158) and from origin J to destination I (including nodes K403 and K179), respectively. These routes are the most heavily used in this simulation period.

The average travel time and the number of stops for the Manhattan network are depicted in Figure 10. Compared to the uncoordinated OTC nodes, the OTC-DPSS approach could reduce the number of stops by 7% while the average travel time in the network was kept for most of the simulation period. Increased travel times and stop counts occur after the sudden major change in traffic at simulation second 5,400 that would never occur this abrupt in a real traffic network. The temporarily decreased system performance is due to the time needed to detect the changing traffic demands. An improved observation mechanism (that uses shorter rolling averages for situation detection) should be able to significantly speed up the system’s reaction time.

For both investigated scenarios, the networks benefited from the established collaboration in the OTC-DPSS approach, especially with respect to the resulting number of stops, which indicates that the distributed traffic-dependent creation of PSSs is a promising approach that will be further refined in the future.

Figure 10 Average travel time and number of stops for Manhattan network
7 Conclusions

This article presented traffic control as an interesting application for organic computing. After introducing existing centralised and self-organising traffic control systems and discussing optimisation approaches, a novel concept for organic traffic lights that are adaptive and learning has been presented. The organic approach is based on industry-standard TLCs that are extended with an observer/controller architecture responsible for the selection and optimisation of appropriate configurations for the operated traffic light. A comparison of an organic traffic light and a system running without the observer/controller showed that the average delay time was reduced for the test cases located at Hamburg, Germany, thereby proving the feasibility of the presented approach. To further improve the performance of the organic nodes in urban areas, a decentralised collaboration mechanism for neighbouring nodes has been presented. The mechanism allows the traffic-responsive creation of PSSs. A simulation study investigated the benefits of collaborative traffic lights for several test networks. Compared to uncoordinated traffic lights, especially the average number of stops (which is an important factor with respect to vehicle emissions) was reduced while the average delays in the network were improved slightly.

Acknowledgement

The financial support by the German Research Foundation (DFG) within the priority programme 1183 ‘Organic Computing’ is gratefully acknowledged.

References


