Adaptive Decentralized and Collaborative Control of Traffic Lights

F. Rochner
rochner@sra.uni-hannover.de

C. Müller-Schloer
cms@sra.uni-hannover.de

Abstract: We present an approach to design an improved control system for arbitrary urban traffic networks based on Organic Computing concepts. For control of these networks a decentralized structure consisting of small computers at every node of the net is proposed. Each node is viewed as an autonomous agent with limited sensory horizon. It tries to adapt itself such that the local traffic throughput is maximized. Additionally communication takes place between adjacent nodes which leads to an iterative global optimisation through collaboration of the agents. The paper discusses the approach in detail and introduces the controller architecture.

1 Introduction

Many urban areas suffer from congestion of their traffic networks. In most cases, the roads themselves have sufficient performance but intersections and similar "nodes" are bottlenecks and the reason for delays. This is where we think further optimisation is necessary and possible.

Many traffic control systems used today have no or very limited means to react on varying traffic situations. Especially in dense urban areas considerable effort has been made to achieve a more or less global optimisation, often by means of a *centralized* control structure. Because of the great complexity of this task (it is known to be np-complete [LA97]), partly caused by the high dynamics of traffic systems, the problem has to be simplified to be solvable in real time. But despite the simplification considerable computing power has to be used if the results are to be sufficiently accurate. Furthermore, these systems usually have to be customized for each application by transferring knowledge of human experts into the system.

2 A New Approach

Our goal is a traffic control system which does not need any central components but still is able to do a global optimisation, adapts itself to different environments, reacts quickly to changing traffic situations, is stable and remains operative even if part of the system fails. This leads to the system having some kind of organic – and hence lifelike – structure.

Quite a few techniques do exist that are eligible to be used as a part of a traffic control system: Simulated Annealing [AvL92] is a stochastic optimisation method, which in analogy to nature tries to imitate a solidifying molten metal mass – easily shaped in the beginning but allowing only small changes in the end. Genetic algorithms [Go89] make use of large populations of genetically coded individuals to search for a solution of a given problem. The search is done by recombination of individuals, mutation and selection mechanisms. Classifier systems [BGH87] use simple if-then rules, which react on input events and generate corresponding actions. The adaptive component is realized by reward and punishment mechanisms (reinforcement learning). Learning Classifier Systems (LCS) [HB+00] usually incorporate a genetic algorithm to create new rules and thereby explore the search space.

In the last few years we have explored the practical usability of Classifier Systems and how these can be extended to be usable in technical systems [BEMS95, KMS02].

2.1 Decentralization

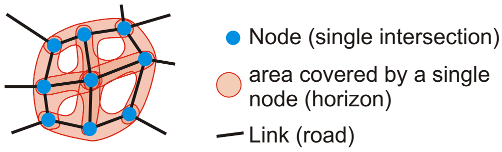

Figure 1: Schematic view of a traffic network.

The central idea of our approach is to spread the computing power needed to control a road traffic network all over this network, in fact placing a small computer at every node (junction with traffic lights) of it. This computer controls only this single junction and collects traffic data by means of the junction's sensors. The collected data and the decisions made by the algorithm running at each node on the basis of these data are communicated to adjacent nodes, and this additional information is also used for the decisions made. This way the area each node controls overlaps with those of the adjacent nodes (fig. 1), and collaboration between nodes is possible.

However, optimisation through collaborating agents might lead to instability through oscillations. Stable control in normal conditions has to be balanced against rapid response to changes in the environment.

2.2 Adaptive Control

The central part of the new controller is an adaptive algorithm. It has to be designed to be able to deal with many different (traffic) situations quickly and autonomously (without requiring human interference), interacting with its environment through sensors and effectors. Figure 2 shows a more general illustration of an autonomous agent resembling a kind of artificial life form. In our case this agent will have to adapt to a highly dynamic environment, using the adaptive algorithm as its competence module. Previous work [KMS02] has shown that a Fuzzy Classifier System (FCS) is well suited for this task.

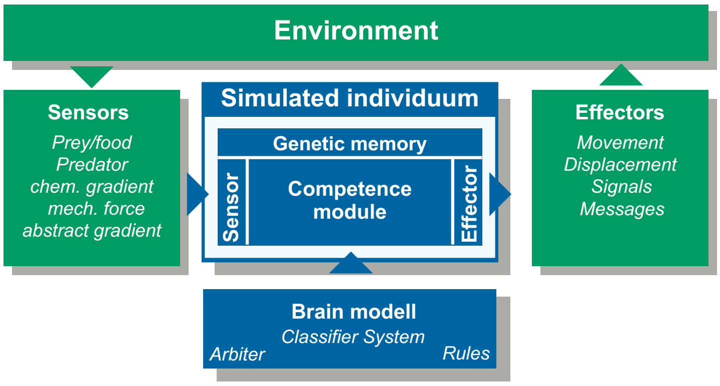

Figure 2: Controller as an artificial life form.

A Classifier System (CS) consists of rules (classifiers) linking certain situations, represented by a set of input variables (condition), to actions to be taken if the condition is matched by actual input data. If the system is to act adaptively, the set of rules used is not fixed but evolves over time. This is done by associating each rule with an additional parameter representing the value of this rule. Every time a rule's action is executed, the effect of this action on the environment is evaluated and the outcome of this evaluation (reward or punishment) is used to update the rule's value. The higher the value of a rule, the higher is the probability of this rule being selected for execution in case more than one rule matches the current input data.

So far, only existing rules are used and evaluated. To scan the search space, a genetic algorithm (GA) is used which generates new rules by combining existing ones and changing parts of them with a certain probability (mutation). Rules used for recombination are chosen according to their value. To make sure this value is based on a reasonable number of tests, another parameter is added to the rules representing the age of the rule (increasing every time a rule's value is recalculated), introducing effects similar to Simulated Annealing to the system. This information is used to reduce the probability of choosing very young rules for recombination as well as to reduce the probability of changing rules through mutation that have proven useful in many tests. This way, the GA scans the entire search space but intensifies search around particular rules which already have been successfully applied for a long time. So there can be rigid, essential rules, and new ones, which yet have to prove successful and thus easily might be replaced or changed.

The age of a rule is furthermore used to build a kind of long-term memory of rules well qualified for certain traffic situations. If these have been used often enough, they won't be altered or deleted from the rule set, even if the corresponding traffic situation is not encountered for a longer period of time. This way, once acquired knowledge is not lost and is instantly available if this situation reoccurs. This is one component of a strategy followed to reduce reaction times of the control system when confronted with a changed environment, as it is typical for traffic situations to occur repeatedly once a day (morning peak, evening peak, . . .) or more irregularly (events like soccer matches).

To be usable in real applications, the control system has to act in a sensible way right from the start of operation. This means that the system cannot start from scratch but must be provided with a-priori knowledge in the form of a generic rule set derived from expert knowledge. If a situation not covered by this generic set is encountered, a rule with matching condition and random action is generated (covering).

Another problem to be solved is the fact that practical applications usually provide the control system with real-valued input data, whereas classifier systems have extensively been used in discrete-valued environments. As the classifier conditions cannot handle real-valued data directly a suitable mapping has to be found. For this, we use fuzzy sets. The range of possible input values is divided into a set of trapezoid membership functions, which can be associated with linguistic terms like e. g. {small; medium; large}. A crisp input value is fuzzified by determining its degree of membership to each of these functions. The condition of classifiers is composed of these linguistic terms, thus classifiers are activated proportional to the degree of membership of a fuzzified input value to the functions corresponding to these linguistic terms, and the action to be carried out is composed of this "action set" according to the degree of activation. Figure 3 shows a Classifier System with fuzzification of input values and defuzzification of output values.

This procedure has certain advantages over simply dividing the real values in fixed intervals. As the transition between two intervals is smooth (membership functions should always overlap), there is no discontinuity for slight changes of values. This is especially important with regard to noisy data, as it should limit oscillations and thus enhance robustness.

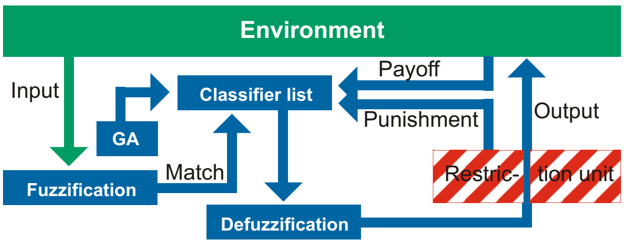

Figure 3: Structural view of a Learning Fuzzy Classifier System incorporating restriction unit and Genetic Algorithm.

Traffic control is a security critical application. Certain constraints may under no circumstances be violated, therefore a *restriction unit* is required to ensure this. However, "creativity" in inventing new rules should be restricted as little as possible. Thus, the restriction unit is to be placed between the rule invention/refinement section and the execution unit. If the action proposed by the system leads to a violation of a constraint (*"all traffic lights green"*) the corresponding rule(s) will be deleted or at least severely punished and the system is to choose another action.

A Fuzzy Classifier System including most of the mentioned features has been implemented at our institute to solve the task of playing Air Hockey (see [KMS02]). However, it lacks communication features as these are not required for this kind of task.

2.3 Extensive Tests through Precise Simulated Environments

Before a new traffic control system can be tested in the field, extensive tests are necessary to ensure functionality and secure operation. Previous research projects in this field often used very simple simulated traffic networks. Our system will be tested in different networks, varying in size and complexity. As the first part of this project we developed a simple traffic simulation tool capable of microscopic simulation of small to medium networks with reduced complexity (e. g. only one lane per direction). Future tests will be done using state-of-the-art traffic simulation tools. These are suitable to model traffic networks found in reality and to be calibrated to deliver data very much alike to what would be received if the control system was used in the real network.

3 Status and Outlook / Conclusion

Currently we are working on a simplified controller to be tested using a traffic model of reduced complexity as a proof of concept. Work on the mentioned Air Hockey controller continues and is expected to give further valuable stimuli.

References

[AvL92] Aarts, E. H. L. and van Laarhoven, P. J. M.: *Simulated annealing: theory and applications*. Kluwer. Dordrecht. 1992.

[BEMS95] Bartscht, E., Engel, J., and Müller-Schloer, C.: The investigation of lamarckian inheritance with classifier systems in a massively parallel simulation environment. In: *Proc. ECAL 1995, Lecture Notes in Artificial Intelligence: Advances in Artificial Life, Granada*. page 920. Springer. 1995.

[BGH87] Booker, L. B., Goldberg, D. E., and Holland, J. H.: *Classifier systems and genetic algorithms*. Ann arbor, Mich. 1987.

[Go89] Goldberg, D. E.: *Genetic algorithms in search, optimisation and machine learning*. Addison-Wesley. 1989.

[HB+00] Holland, J. H., Booker, L. B., et al.: What is a Learning Classifier System? In: Lanzi, P. L., Stolzmann, W., and Wilson, S. W. (editors), *Learning Classifier Systems. From Foundations to Applications*. volume 1813 of *LNAI*. pages 3–32. Berlin. 2000. Springer-Verlag.

[KMS02] Koblitz, D. and Müller-Schloer, C.: Extension of fuzzy classifier-mechanisms for adaptive embedded systems by a-priori knowledge and constraints. In: Polani, D., Kim, J., and Martinetz, T. (editors), *Fifth German Workshop on Artificial Life : GWAL-5*. pages 1–8. Akademische Verlagsgesellschaft Aka. Berlin. 2002.

[LA97] Lenstra, J. K. and Aarts, E. L.: *Local Search in Combinatorical Optimization*. Chichester. 1997.