Reports

Machine Learning and Inference Laboratory

A Natural Induction Approach to Traffic Prediction for Autonomous Agent-based Vehicle Route Planning

Jan D. Gehrke and Janusz Wojtusiak

MLI 08-1

February 17, 2008

George Mason University

NATURAL INDUCTION APPROACH TO TRAFFIC PREDICTION FOR AUTONOMOUS AGENT-BASED VEHICLE ROUTE PLANNING

Jan D. Gehrke [1] and Janusz Wojtusiak [2]

[1] Collaborative Research Center 637 and
Center for Computing Technologies − TZI
University of Bremen
28359 Bremen, Germany

[2] Machine Learning and Inference Laboratory
Department of Health Administration and Policy
George Mason University
Fairfax, VA 22030, USA

Abstract

This paper describes a methodology and initial results of predicting traffic by autonomous agents within a vehicle route planning system. The predictions of the traffic are made using AQ21, a natural induction system that learns and applies attributional rules. The presented methodology is implemented and experimentally evaluated within the multiagent-based simulation system PlaSMA. Results obtained within the simulation system indicate advantage of agents using AQ21 predictions when compared to naïve agents that make no predictions, and agents that use only weather-related information.

Keywords: Traffic Prediction, Multiagent-based Simulation, Natural Induction, Computational Learning, Transportation

Acknowledgments

The authors thank Abdur Chowdhury for his contributions to experiment design, and Jarek Pietrzykowski for his comments that helped to improve this paper. This research was partially funded by the German Research Foundation (DFG) within the Collaborative Research Centre 637 "Autonomous Cooperating Logistic Processes" (SFB 637). Development of the AQ21 system was partially funded by the National Science Foundation Grants IIS 9906858 and IIS 0097476.

1 INTRODUCTION

The importance of information technology (IT) in logistics has increased remarkably. IT systems support or take responsibility for logistics planning of a forwarder's whole motor pool or single vehicles. The just-in-time paradigm demands for high robustness to situation changes and real-time coordination abilities of all participants in the logistics network. These requirements are facilitated by pervasive mobile communication networks and devices as well as intelligent systems processing incoming information (Scholz-Reiter et al., 2004). Due to the fierce competition in the logistics service market, companies are searching for new technologies that provide advantages in quality of service (e. g., promptitude and robustness to disturbances) and cost of carriage. Thus, in this area even seemingly small differences may bring about a considerable competitive advantage and high economic impact.

The agent-based approach to intelligent logistics systems delegates planning and decision making from central (corporate) planning systems to single logistic entities, such as trucks, parcels, containers, that decide autonomously and locally. The decisions take place where they should be executed. This decentralized approach provides easier access to local information needed for making the best decision and, due to reduced complexity for (re-)planning, it is less vulnerable to environmental changes such as new transport orders or breakdown of vehicles. Decentralized local decision-making cannot guarantee optimal solutions in global or company-wide scale (Wörner and Wörn, 2006). But in practice, the goal of having a steadily optimized plan for a large-scale set of resources that face unpredictable events in a dynamic environment becomes an illusion anyway. A seemingly optimal solution may prove bad in the end when situation changes. In general, global or beyond-local plans may provide better solutions than local plans even on average. Local planning aims at avoiding the most harmful cases that really make the difference with respect to costs. One might also think about hybrid solutions that try to combine the best of both worlds by providing global plans in a limited temporal scope and allowing local adaptations if required by upcoming events.

The ability of autonomous systems to react to situation changes calls for situation awareness provided by local sensor, other agents, or external information sources such as databases or web pages. In order to make use of this information the agent needs to know its relevance, i.e., how specific information or a kind of information influences its cost function or expected utility during planning of actions. Because information relevance in logistics is hard to assess in advance, an experience-driven approach is needed. Furthermore, the agent has to consider the spatio-temporal scope of information, i.e., where and how long the information is valid and valuable. Because the scope of agent planning is in the future and possibly remote locations, the information needed may not be accessible. That is why agents require prediction abilities.

In this report we examine the influence of environmental knowledge in vehicle route planning in simulation experiments. In particular, we analyze how road traffic predictions used in route planning affect vehicles' performance measured as the time needed to reach their destination. In order to do this, we compare performance of ignorant agents with agents that are provided with weather information and agents that use rules induced from historical traffic data for predictions.

The rules used in the latter approach are learned using *natural induction* system AQ21. The system puts an equal importance to rules' accuracy and understandability, thus it makes learned knowledge verifiable and modifiable by humans.

Section 2 introduces the basic planning algorithm applied in this study. Section 3 describes the simulation model including the participating software agents. Section 4 discusses the learning approach based on natural induction; Section 5 is concerned with its application for traffic and route assessment. Section 6 provides experiment descriptions and simulation results. The report is concluded with a brief summary and a discussion of future research directions.

2 AGENT-BASED VEHICLE ROUTE PLANNING

The performance of autonomous intelligent planning systems heavily depends on a proper analysis of the current situation and prediction of future events. Depending on the specific domain, the planning scope may cover from half a day (for regional transports) to a few days for international or overseas routes. Thus, an important requirement for intelligent planning systems is situation awareness (Endsley, 2000) as well as situation prediction.

In order to initially investigate the impact of such abilities in logistics and to understand the spatial and temporal constraints we reduced the problem to a simplified single vehicle transportation scenario. A software agent represents a truck that aims at finding the fastest route in a graph-based highway network model. While there are several equally short routes, they differ in driving time. This is a complex problem because there are highly dynamic environmental conditions that may enforce or reduce the speed at which the vehicle may travel. The examined conditions include traffic density and weather which both vary in space and time.

The agent-based vehicle route planning applies an A* search algorithm (Hart et al., 1968) with cost function (2-1) for reaching destination d when using (partial) route r at departure time t_{dep}:

$$\mathrm{f}(r,\ d,\ t_{\mathrm{dep}}) = \mathrm{g}(r,\ t_{\mathrm{dep}}) + \mathrm{h}(end_{\mathrm{r}},\ d) \tag{2-1}$$

with g as the estimated driving time for *r* and *h* as the estimated driving time from *r*'s endpoint end_r to *d*. Heuristics *h* is calculated as driving time at straight line distance from end_r to *d* at maximum vehicle speed. The route *r* consists of *n* consecutive edges (i.e., roads) $e_i \in r$ with $0 \le i < n$. The route segment of first *k* edges is denoted by $r_{\mathrm{k\text{-}1}}$. The driving time *g* on each potential route *r* is calculated by:

$$g(r, t_{dep}) = \sum_{i=0}^{n-1} \frac{length(e_i)}{v_{est}(e_i, t_{dep,e_i})} \tag{2-2}$$

with $v_{est}(e_i, t_{dep,e_i})$ as the estimated vehicle speed on edge e_i which depends on the vehicle agent implementation (Sect. 3). While departure time for first edge e_0 equals t_{dep} time for edges with $\mathrm{i} > 0$ is defined by:

$$t_{dep,e_i} = g(r_{i-1}, t_{dep}) \tag{2-3}$$

Because this setting ensures the criteria for the A* algorithm (non-negative costs and optimistic heuristics) it guarantees the optimal solution. However, the found route is only optimal provided

that agent's knowledge about the environment used in the cost function is complete and correct. When considering the cost function for roads that are ahead in time these assumptions are possibly wrong because the environment continuously changes in a way that cannot be predicted exactly.

3 SIMULATION MODEL

In order to evaluate the impact of environmental knowledge and driving time prediction abilities we set up experiments with the simulation system *PlaSMA* (*Platform for Simulation with Multiple Agents*[1]). PlaSMA (Gehrke, 2007) is a multiagent-based simulation (MABS) system, i. e., discrete event, distributed simulation using software agents as logical simulation processes. MABS provides a natural way to evaluate software agent behavior and interaction.

For the purpose of vehicle route planning, the simulation model includes two *world agents* and multiple *vehicle agents*. The world agents simulate weather and the traffic within the simulation environment. The vehicle agents can be categorized in three classes: the *ignorant* agent, the *weather-aware* agent, and the *predictive* agent. While all vehicle agents use A* search with time cost function for routing (Sect. 2), they use different knowledge in planning and thereby provide a comparative measure to evaluate the impact of their knowledge.

3.1 Weather and Traffic: The World Model

For the logistics scenario we set up a road network as a 6 × 6 grid graph with a set $\mathcal{E}$ of directed edges (i. e., unidirectional roads) of 100 km length each (see Figure 1). Although this idealized grid structure may appear not very complex and not well-fitting for possible real world applications, it does ensure that results are not owing to a special network structure that may have no implications for the general problem. Nevertheless, experiments can be conducted with other networks as well. The two world agents generate simulation events that affect this graph with respect to traffic density $\mathrm{dens}(e,t)$ and maximum safe speed $v_{\mathrm{safe}}(e,t)$ for each edge $e \in \mathcal{E}$ at simulation time $t \in \mathbb{N}$. The simulation time value corresponds to milliseconds since 1970 and thus enables the assignment to time of day and day of week.

The notion of traffic density in this model differs from the usual definition of vehicles per kilometer (FGSV, 2005). The usual definition's value has an open range increasing with number of vehicles. Additionally, the value does not say anything about the quality of traffic because it depends on the number of lanes present. For example, according to guideline values of German state agency FGSV (FGSV, 2005), a German road with two lanes is considered overloaded if density is greater or equal 45. But unfortunately there is no exact mapping of traffic density to its effect on vehicle speed as needed for the simulation model. Hence, the concept of traffic density is used differently in our experiments: Function $\mathrm{dens}(e,t)$ provides a linear traffic *quality* measure that is normalized to 1 and indicates the ability of a vehicle to drive at a reference speed

[1] Available from `http://plasma.informatik.uni-bremen.de/`

v_{ref}, i. e., maximum possible speed on every edge e is:

$$v_{\max}(e,t) = v_{\text{ref}}(e) \cdot \text{dens}(e,t) \tag{3-1}$$

For our experiments we use $v_{\text{ref}}(e)$=130 km/h for all roads. This value corresponds to the speed on German Autobahn roads for passenger cars that is recommended by German federal by-law "Autobahn-Richtgeschwindigkeits-Verordnung" (ARV, 1978/1997) in case there are no traffic signs posting a lower general speed restriction or another recommended speed. Consequently, if traffic density is 0.5 a vehicle is not able to drive faster than 65 km/h on average.

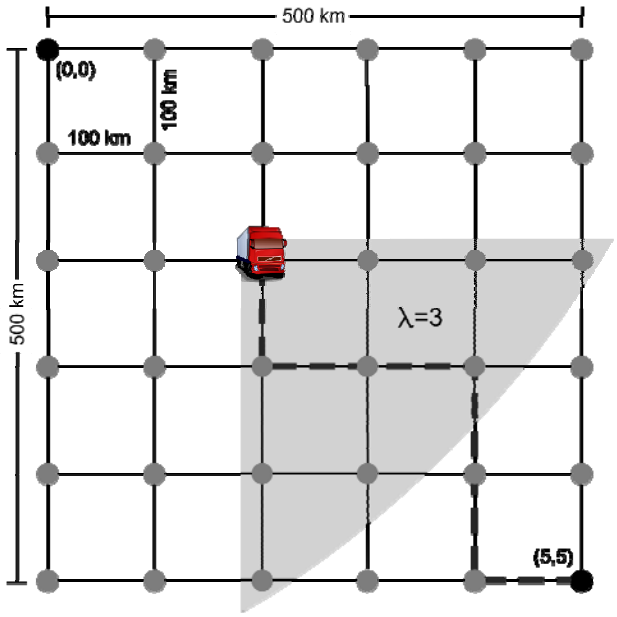

Figure 1: Scenario road grid showing a weather-aware truck with 300 km look-ahead distance.

Speed v_{safe} depends on the current weather on each road. Different roads may have different weather but the weather within the area of each road is homogeneous. The simulation model is designed for a finite set $\mathcal{W}$ of qualitative weather types

$$\mathcal{W} = \{VeryBad,\ Bad,\ Moderate,\ Good\}$$

with the antisymmetric linear orders

$$\preccurlyeq \subset \mathcal{W} \times \mathcal{W} \quad \text{and} \quad \succcurlyeq \subset \mathcal{W} \times \mathcal{W}$$

denoting the relative severity of weather. Each weather type $w \in \mathcal{W}$ corresponds to an edge-independent and truck-oriented maximum safe speed $v_{\text{safe}}(w)$ when facing that weather. More severe weather always causes lower safe speed, i. e.,

$$w_{\text{i}} \succcurlyeq w_{\text{k}} \rightarrow v_{\text{safe}}(w_{\text{i}}) < v_{\text{safe}}(w_{\text{k}}) \tag{3-2}$$

For the experimental setting in this report $v_{\text{safe}}(w)$ is defined as:

$$\mathrm{v}_{\mathrm{safe}}(\mathrm{VeryBad}) = 35\ \mathrm{km/h}$$
$$\mathrm{v}_{\mathrm{safe}}(\mathrm{Bad}) = 65\ \mathrm{km/h}$$
$$\mathrm{v}_{\mathrm{safe}}(\mathrm{Moderate}) = 80\ \mathrm{km/h}$$
$$\mathrm{v}_{\mathrm{safe}}(\mathrm{Good}) = 100\ \mathrm{km/h}$$

The function $\mathrm{weather}(e,t)$ describes the weather at an edge $e \in \mathcal{E}$ and at time *t*. Then, the maximum safe speed for each edge e is determined by

$$v_{\mathrm{safe}}(e,t) = v_{\mathrm{safe}}(\mathrm{weather}(e,t)) \tag{3-3}$$

Together, v_{max} and v_{safe} determine the simulated average speed $v_{\mathrm{avg}}(e,i)$ of a vehicle on edge *e* during a time interval $i = [i_{\mathrm{start}}, i_{\mathrm{end}})$ with constant weather and traffic conditions:

$$v_{\mathrm{avg}}(e,i) = \min(v_{\mathrm{safe}}(e,i_{\mathrm{start}}),\ v_{\mathrm{max}}(e,i_{\mathrm{start}})) \tag{3-4}$$

Weather generation

The weather agent updates weather in an interval $\Delta t_{\mathcal{W}}$ separately for each edge in $\mathcal{E}$. The new weather type $w \in \mathcal{W}$ on each edge e depends on the previous weather at e and the basic probability distribution $\mathbf{P}(\mathcal{W})$ for weather which is assumed to be location-independent. Let $t_{\mathcal{W},\mathrm{i}}$ be the last weather update then the next weather update is

$$t_{\mathcal{W},\mathrm{i+1}} = t_{\mathcal{W},\mathrm{i}} + \Delta t_{\mathcal{W}} \tag{3-5}$$

Weather does not change during this interval, i. e.,

$$\forall\, e \in \mathcal{E}\ \forall\ t:\ t \in [t_{\mathcal{W},\mathrm{i}},\ t_{\mathcal{W},\mathrm{i+1}}) \rightarrow \mathrm{weather}(e,t) = \mathrm{weather}(e,t_{\mathcal{W},\mathrm{i}}) \tag{3-6}$$

The next $\mathrm{weather}(e,t_{\mathcal{W},\mathrm{i+1}})$ is determined by the weather w randomly drawn according to $\mathbf{P}(\mathcal{W})$ but changes may be constrained depending on the current weather to avoid sudden changes as determined by transition probability model $\mathbf{P}(\mathcal{W}_{t+\Delta t\mathcal{W}} \mid \mathcal{W}_t)$. The actual distributions for weather and length of $\Delta t_{\mathcal{W}}$ are subject of the experimental setup (Sect. 6).

The weather agent also implements a service that enables vehicle agent to retrieve information on current weather of locations that are considered interesting for their route planning (Sect. 3.3).

Traffic generation

Similarly to the weather agent, the traffic simulation agent updates traffic density on each edge in interval $\Delta t_{\mathcal{T}}$. Though the generated traffic density is a real number, the traffic generator is based on a qualitative traffic model with a set of traffic classes $\mathcal{T}$ covering disjoint intervals of traffic density. Similarly to traffic qualities (*level of service*) A to F in US Highway Capacity Manual (HCM, 2000) and its German pendant (FGSV, 2005), the model distinguishes six traffic classes with

$$\mathcal{T} = \{VeryLow,\ Low,\ Medium,\ High,\ VeryHigh,\ Jam\}$$

and the following traffic density intervals:

$$VeryLow = [0.0,\ 0.1)$$
$$Low = [0.1,\ 0.25)$$
$$Medium = [0.25,\ 0.4)$$
$$High = [0.4,\ 0.6)$$
$$VeryHigh = [0.6,\ 0.85)$$
$$Jam = [0.85,\ 1.0].$$

The traffic density on an edge is determined by an edge-specific density matrix for time of day and day of week. For this purpose, we analyzed traffic volume data from German (BASt) and Austrian (ASFINAG) agencies that count traffic on national highways. According to the combined days and hours in aggregated agency models we set up the basic model matrix depicted in Table 1 that determines the basic mean traffic density value $\mathcal{T}_{\mu}(t)$ at time t. The basic model is used for all edges but each edge has an additional and time-independent traffic bias function $\Delta\mathrm{dens}(e)$ that may shift the basic model to higher or lower traffic densities. One limitation of this model is that it does not allow for time-specific changes, e. g., modeling commuter roads that, in one direction, are congested in the morning but free in the afternoon and the opposite for the other direction.

Table 1: Basic traffic density matrix

		Mo	Tu-Th	Fr	Sa	Su
Morning	6am-10am	High	Medium	Medium	VeryLow	VeryLow
Noon	10am-3pm	Low	Low	High	Low	Low
Afternoon	3pm-7pm	Medium	Medium	High	Medium	Medium
Evening	7pm-10pm	Low	Low	Medium	Low	Low
Night	10pm-6am	VeryLow	VeryLow	Low	VeryLow	Low

The actual values for the density bias function $\Delta\mathrm{dens}$ depend on the experimental setup (see Sect. 6). In general, the bias value is in interval [-0.2, +0.2]. The traffic generation world agent calculates the actual density $\mathrm{dens}(e,t)$ with Gaussian distribution $\mathrm{N}(\mu, \sigma)$:

$$\mathrm{dens}(e,t) \sim \mathrm{N}(\mathcal{T}_{\mu}(t) + \Delta\mathrm{dens}(e), \sigma) \tag{3-7}$$

with the distribution variance σ^2 depending on the experimental setup.

3.2 Ignorant Agent

The *ignorant agent* is not aware of any environmental information and has no predictive abilities. Like all other vehicle agents it uses the routing algorithm described in Sect. 2. However, the ignorant agent does not make any assumptions on changing speed on roads because it has no information that could indicate such changes. Because this agent assumes that all roads allow equal speed its decisive planning criteria becomes distance. Thus, within the road grid the ignorant agent chooses a route that most closely matches the straight line, disregarding possible severe weather conditions or traffic congestions. Although this agent might appear simpleminded,

its behavior must still be considered rational provided its lack of knowledge.

3.3 Weather-aware Agent

The *weather-aware agent* is a vehicle agent that acquires environmental status information (i. e., on weather) for interesting locations. The agent uses this information to determine speed $v_{\text{est}}(e,t_{\text{dep,e}})$ for route planning (Sect. 2). It does not attempt to predict any traffic or changes of weather but naïvely assumes that there is no (relevant) traffic or change of weather with respect to current time t_{cur}, i. e.,:

$$v_{\text{est}}(e,t_{\text{dep,e}}) = v_{\text{safe}}(e,t_{\text{cur}}) = v_{\text{safe}}(\text{weather}(e,t_{\text{cur}})) \tag{3-8}$$

The status information is provided by the weather agent. Which locations are considered interesting depends on the current vehicle location, its destination, and a look-ahead parameter. Look-ahead λ specifies the spatial range and the area for which an agent acquires environmental status information. The actual look-ahead distance used in this study is defined as $\lambda \cdot 100$ km, that is, λ corresponds to the number of (100 km long) edges ahead that are considered interesting in each potential route. Thus, an ignorant agent is equivalent to a weather-aware agent with $\lambda = 0$. In simulation we tested different λ values to evaluate the spatio-temporal constraints making information useful. Because weather-aware agents assume the weather to be static when planning the fastest route, it may turn out to be suboptimal for edges that are far ahead. The probability to be wrong depends on the basic probability of the expected weather, the time distance to the location, and the weather change interval $\Delta t_{\mathcal{W}}$. The impact of being wrong will be determined by simulation. On the other hand, the vehicle agents may reconsider their route at each junction, i. e., wrong decisions can be corrected if there are new, better alternatives. Nevertheless, the more the chosen route is off the theoretical straight line route the fewer good alternatives are available. This is due to the grid road network structure.

3.4 Predictive Agent

The *predictive agent* is an extension of the weather-aware agent. It also assumes that weather at edge e will not change until arrival at e, but it tries to determine traffic at e for time of arrival and potential departure $t_{\text{dep,e}}$ to determine $v_{\text{est}}(e,t_{\text{dep,e}})$ based on an prediction function p_e:

$$v_{\text{est}}(e,t_{\text{dep,e}}) = \text{p}_e(\text{weather}(e,t_{\text{cur}}),\ t_{\text{dep,e}}) \tag{3-9}$$

Each prediction p relies on previous experiences that are used as data to learn situation- and edge-specific rules for expected speed (Sect. 5). The prediction does not take into account $\text{dens}(e,\ t_{\text{dep,e}})$ or $\text{dens}(e,\ t_{\text{cur,}})$ because they are unknown. Besides time of day and day of week the prediction depends on available weather information and thus λ. Edges that are not within λ distance are assumed to have the most likely weather, e. g., $Moderate$.

4 NATURAL INDUCTION

Induction of traffic models that can be used for accurate predictions may be approached using many different methods. In contrast to most methods known in the literature, including different forms of statistical learning (Hastie et al., 2003), the approach used in this study puts an equal importance to the accuracy of learned models, and to their interpretability by human experts. While the importance of the former does not require any justification, the latter may not seem so clear, especially in the transportation area where the learned knowledge is acquired and used by autonomous agents. Models learned by many methods can be regarded as a "black box" which may give very good predictions, but it is hard to understand, and therefore to validate and refine by human experts. While for "normal" operation of autonomous agents, such "black box" models are sufficient, there are special cases when an agent is unable to make decision, thus the decision is to be made by a human operator and understandability of knowledge is crucial. Learned models described in the language that is easy to understand, for example, natural language or easy to interpret rules, can also be modified by experts to reflect special cases and the experts' background knowledge.

Natural induction proposed by Michalski (2004) is an approach to inductive learning whose goal is to achieve the above stated high understandability of learned knowledge. It uses a highly expressive language *attributional calculus* (AC) that combines predicate, propositional and multi-valued logics. Because one of the main goals of attributional calculus is to be able to express knowledge in the forms that are easy to understand by people, all forms of knowledge used in AC correspond to different constructs of natural language. One way of looking at attributional calculus is as the formal representation of simple forms of natural language.

The following subsections describe representation of knowledge in attributional calculus, and present a symbolic machine learning system AQ21. The system realizes a simple form of natural induction, by inductively learning hypotheses in attributional calculus.

4.1 Knowledge Representation

Attributional rules are the main form of knowledge in attributional calculus. They follow the general *if … then …* schema, but are more general than those learned by most learning programs. This is because attributional rules use more expressive language which allows describing simply more complicated statements. Formally, attributional rules take the forms (4-1) – (4-3).

$$CONSEQUENT \Leftarrow PREMISE \tag{4-1}$$

$$CONSEQUENT \Leftarrow PREMISE \llcorner EXCEPTION \tag{4-2}$$

$$CONSEQUENT \Leftarrow PREMISE \ulcorner PRECONDITION \tag{4-3}$$

Here, *CONSEQUENT*, *PREMISE* and *PRECONDITION* are conjunctions of attributional conditions. *EXCEPTION* can be either a conjunction of attributional conditions or an explicit list of examples constituting exceptions to the rule. Attributional conditions are often called selectors, and their conjunctions are often called complexes.

There are many forms of attributional conditions, all of which resemble simple natural language statements. The form of attributional conditions used in this study can be expressed by (4-4).

$$[L \ \ rel \ \ R] \tag{4-4}$$

In this study L is an attribute; R is a value, a disjunction of values, or a conjunction of values if L is a compound attribute; and *rel* is a relation that applies to L and R. Examples of attributional conditions in the above form are presented in Table 2.

Table 2: Examples of attributional conditions.

Condition	**Explanation**
[Day = Friday]	The day is Friday.
[Time=morning..afternoon]	The time is between morning and afternoon.
[Weather=moderate v good]	The weather is moderate or good.
[Speed > 50 km/h]	The speed is greater than fifty kilometers per hour.

Other forms of attributional conditions may involve count attributes, simple arithmetical expressions, conjunctions and disjunctions of attributes, comparison or attributes, etc. Attributional rules can be automatically translated to natural language sentences as described by Michalski and Wojtusiak (2007).

4.2 The AQ21 Machine Learning System

A simple form of natural induction is implemented in the currently developed AQ21 system (Wojtusiak et al., 2006). The system is the newest member of the AQ family of symbolic machine learning systems. It integrates many features that are either unique to AQ21, or are present only separately in other machine learning programs. This section gives a brief overview of AQ21, lists its most important features, and describes its use in learning and predicting traffic patterns.

The AQ21 system learns descriptions of classes (concepts) given examples belonging to these classes, and examples not belonging to them. For instance, a class may represent situations in which a vehicle speed on a particular road is fast, in contrast to all situations where the speed is not fast. The examples are given in the form of attribute-value tables as exemplified in Table 3. In this table, Day, Time, and Weather are input (independent) attributes and Speed is an output (dependent) attribute.

Table 3: Example input data table for AQ21.

Day	Time	Weather	Speed
TuTh	afternoon	moderate	fast
TuTh	evening	moderate	fast
TuTh	night	moderate	fast
Sa	Evening	moderate	fast
Sa	noon	moderate	fast
Mo	night	moderate	fast
Mo	noon	good	very_fast
TuTh	afternoon	bad	medium
TuTh	noon	good	very_fast
Mo	afternoon	very_bad	very_slow
Mo	morning	moderate	medium

Given input data, problem definition, and optional background knowledge, the program induces rules in the forms (4-1) – (4-3) briefly discussed in the previous section and also optionally outputs their natural language equivalences. A set of rules constituting a description of a given class is called a *ruleset* or a *model* of the given class. By repeating learning for all classes defined by values of an output attribute, AQ21 generates a *classifier*. For instance, given the examples presented in Table 3, AQ21 induces the following rule describing for fast speed:

```
[Speed=fast]
   <= [Time=noon..night: 6,4,60%,6,4,60%]
      [Weather=moderate: 6,1,85%,6,0,100%]
       : p=6,n=0,cx=7 #(Rule 1)
```

The learned ruleset consists of only one rule, which is sufficient to describe all examples of speed fast. The program also automatically generates the following sentence equivalent to the learned rule: *Speed is fast if time is between noon and night and weather is moderate.* The six numbers inside conditions are: the number of matching positive examples p_c, the number of matching negative examples n_c, confidence $p_c/(p_c+n_c)$. Next three numbers correspond to cumulative coverage and confidence, that is, coverage based on matching a given condition and all previous conditions. Three numbers in the rule annotation indicate its positive coverage p, negative coverage n, and complexity cx. Depending on AQ21's parameter settings only some or none of the numbers may be displayed.

In order to learn rules for a given class, AQ21 concentrates on one example, called a *seed*, of the class. It generates a star, which is a set of maximally general rules that cover the seed and do not cover any examples from other classes. This is done by repeating an *extension-against* operation that generalizes the seed against a given example not belonging to the concept being learned. Results of applying extension-against are intersected and the best rules are selected according to user-defined criteria. If selected rules do not cover all examples of the class, another seed is selected (from the not covered examples) and additional rules are learned. The process is repeated until all examples of the class are covered by learned rules (e.g. Wojtusiak et al., 2006).

AQ21 implements several modifications to the above basic algorithm. These include, selecting

multiple seeds at the same time, allowing partial inconsistency of learned rules, ordering examples before applying extension-against, learning exceptions, improving representation spaces, and others. The system operates in three basic modes. In the first one, called *theory formation*, the learned rules are complete and consistent with the data. This mode is often applied to relatively small datasets with no noise. In the second mode, called *approximate theory formation*, the program learns rules that may be partially inconsistent or incomplete, but optimize the Q(w) quality measure (Kaufman and Michalski, 2000). In the third mode, called *pattern discovery*, the program finds a collection of strong regularities in data. The regularities, called patterns, may not cover all examples of the class, and may also be inconsistent. This mode is intended for rather exploratory analysis of data and aims at finding regularities in data, rather than for learning models that can be directly used for classification.

Application of learned rules to classify new examples is done using ATEST module implemented in AQ21. Matching of learned rules and examples is done on three levels, namely conditions, rules, and rulesets. On each level it can be done strictly, with match or no match, or flexibly, with a degree of match ranging from zero to one. Details of different matching schemas are described by Michalski (2004).

In the experiments presented in this paper the agent uses qualitative speed predictions from AQ21 and maps them to v_{est} (see Section 2). The ATEST module in AQ21 can give imprecise answer by classifying an example to more than one possible speed class. Two approaches of resolving imprecise classifications are investigated here. The first approach is pessimistic, namely it assumes the worst of the given answers (the lowest predicted speed). The second approach is based on frequency of classes. For instance, if an example matches two classes "slow" and "very slow," and the former is more frequent in the training data, it is reasonable to assume that the prediction should be "slow."

5 LEARNING TRAFFIC MODELS

In the presented study traffic models were learned using simulated data collected over 15 years of simulation time. The training data consisted of 131,488 examples for each type of edge.

To learn traffic models we used the AQ21 system described in the previous section. The program was executed with different settings of parameters, from which we selected the best according to predictive accuracy on a testing data and simulation results. An example header of the AQ21 input files used in this study is presented in Figure 2. It starts with definition of attributes, their types and domains used in the data. The four attributes defined in the data are Day, Time, Weather, and Speed. Each attribute corresponds to one column in the input data. Additionally, the fourth column in the data is ignored. The next part of the AQ21 input file defines learning problem and parameters. Here, one learning problem, called "Speeds," is specified for learning a classifier for the output attribute Speed as defined by Consequent = [Speed=*]. In the example input file AQ21 is executed in the approximate theory formation mode, with weight of completeness against consistency gain equal 0.1, as indicated by Mode = ATF and W=0.1 parameters, respectively. The parameter MaxStar indicates depth of search for the best rules.

```
Attributes
{
Day     cyclic {Mo, TuTh, Fr, Sa, Su}
Time    cyclic {evening, night, morning, noon, afternoon}
Weather linear {very_bad, bad, moderate, good}
Ignore
Speed   nominal {SPEED_30, SPEED_40, SPEED_50, SPEED_60, SPEED_70, SPEED_80,
                 SPEED_90, SPEED_100}
}

Runs
{
  Speeds
    {
      Mode = ATF
      W = 0.1
      MaxStar = 30
      Consequent = [Speed=*]
      Ambiguity = IncludeInMajority
    }
}
```

Figure 2: Header of AQ21 input file used for learning traffic models.

Due to random factor in the used simulation, large portion of training examples is ambiguous, meaning that for identical values of Day, Time, and Weather, different speeds are assigned. We investigated two methods of handling ambiguous examples. The first one assigns the most frequent class, and the second method always treats ambiguous examples as positive examples. Details of the methods and AQ21 parameters are described in its user's guide (Wojtusiak, 2004).

Application of models to new examples is done by executing AQ21's testing module. The program is provided with input files as exemplified in Figure 3, and one or more testing examples.

In addition to definition of attributes and problem specification, in which mode is set to "test" because no learning is invoked, a new input file section is defined to specify testing parameters. The section defines one test, named "test1," that uses the ATEST method to apply provided rules to classify testing examples. Parameters Evaluation_of_conjunction and Evaluation_of_disjunction specify methods for evaluating conjunctions between selectors in rules, and disjunctions of rules in rulesets, respectively. In the performed experiments, the evaluation of conjunction is either "min" which is equivalent to strict rule matching or "selectors_ratio" which is equivalent to flexible rule matching. In the latter case the degree of match equals the number of matching selectors in a rule to the total number of selectors in the rule. In all reported experiments, evaluation of disjunction is set to "max," meaning that the strongest degree of match to a rule is used. Detailed descriptions of the parameters are in (Michalski, 2004; Wojtusiak, 2004).

```
Attributes
{
Day     cyclic {Mo, TuTh, Fr, Sa, Su}
Time    cyclic {evening, night, morning, noon, afternoon}
Weather linear {very_bad, bad, moderate, good}
Ignore
Speed   nominal {SPEED_10, SPEED_20, SPEED_30, SPEED_40, SPEED_50, SPEED_60,
               SPEED_70, SPEED_80, SPEED_90, SPEED_100}
}

Runs
{
  SPEED_10
    {
      Mode = test
      Consequent = [Speed=SPEED_10]
    }
…
  SPEED_100
    {
      Mode = test
      Consequent = [Speed=SPEED_100]
    }
}

Tests
{
  test1
    {
      Method = ATEST
      Evaluation_of_conjunction = min
      Evaluation_of_disjunction = max
      Full_report = true
      Print_classification_file = "pred_out"
    }
}
```

Figure 3: Header of the AQ21 input file used for applying traffic models.

Selected models (hypotheses) learned by AQ21 for predicting traffic on the edge 1 are presented in Figure 4. The program was executed in theory formation mode and with the majority method for handling ambiguous examples. Note that the figure does not show complete learned classifier, but only rulesets for selected classes. The key word "Input_hypotheses" is used because the presented rulesets are used as an input to AQ21 testing module.

```
Input_hypotheses SPEED_40
{
[Speed=SPEED_40]
   <= [Weather=very_bad]
       : p=1317,n=0,q=1,cx=7 #(Rule 1)
}


Input_hypotheses SPEED_60
{
[Speed=SPEED_60]
   <= [Day=Fr]
      [Time=noon..afternoon]
      [Weather=bad..good]
       : p=4603,n=0,q=0.964,cx=14 #(Rule 1)

   <= [Day=Mo]
      [Time=morning]
      [Weather=moderate..goof]
       : p=1607,n=0,q=0.868,cx=14 #(Rule 2)
}


Input_hypotheses SPEED_100
{
[Speed=SPEED_100]
   <= [Time=evening..night]
      [Weather=good]
       : p=8441,n=370,q=0.921,cx=7 #(Rule 1)

   <= [Time=night,noon]
      [Weather=good]
       : p=9766,n=625,q=0.917,cx=7 #(Rule 2)
}
```

Figure 4: Selected learned models for predicting traffic on the edge 1 in theory formation mode with majority method for handling ambiguity of training examples.

Average predictive accuracies (5-1) and precisions (5-2) of models learned and applied with different AQ21 parameters on testing data with variances 0.001, 0.005 and 0.01, ranged from 91% to 100% and 71% to 100%, respectively. The best results were obtained by AQ21 in the theory formation mode, with ambiguities treated as majority and flexible rule interpretation. There was 95% predictive accuracy and 100% precision. All presented values are averaged for all types of roads in the simulation model.

$$predictive\ accuracy = |cc| \ / \ |T| \tag{5-1}$$

$$precision = (|T| \cdot |C| - |tc|) \ / \ (|tc| \cdot (|C| - 1)) \tag{5-2}$$

Here, $|cc|$ is the number of correct classifications, $|T|$ is the size of testing dataset, $|C|$ is the number of classes, and $|tc|$ is the total number of classifications.

Note, that due to ambiguity in the data, it is not possible to achieve 100% accuracy and 100% precision at the same time.

Table 4: Predictive accuracies and precision of models learned and applied with different AQ21 parameters to testing data with variance 0.001.

Experiment		Edge 0	Edge 1	Edge -1	Edge 2	Edge -1	Average	St. dev.
atf 01 majority flexible	acc	93.07%	89.48%	99.47%	78.82%	99.52%	92.07%	0.076
	prec	97.25%	93.09%	97.41%	86.04%	100%	94.76%	0.048
atf 01 majority strict	acc	93.07%	89.48%	99.47%	76.97%	99.52%	91.70%	0.083
	prec	97.25%	93.09%	97.41%	89.74%	100%	95.50%	0.036
atf 01 pos flexible	acc	95.09%	96.06%	99.84%	94.15%	100%	97.03%	0.024
	prec	87.78%	58.10%	92.73%	36.18%	83.74%	71.71%	0.214
atf 01 pos strict	acc	95.09%	96.06%	99.84%	94.15%	100%	97.03%	0.024
	prec	87.78%	58.10%	92.73%	36.18%	83.74%	71.71%	0.214
tf majority flexible	acc	97.20%	93.65%	99.47%	85.25%	99.52%	95.02%	0.053
	prec	100%	100%	100%	100%	100%	100.00%	0
tf majority strict	acc	97.20%	93.65%	99.47%	85.25%	99.52%	95.02%	0.053
	prec	100%	100%	100%	100%	100%	100.00%	0
tf pos flexible	acc	100.00%	100.00%	100.00%	99.99%	100.00%	100.00%	3.56E-05
	prec	79.72%	56.48%	88.22%	34.36%	97.75%	71.31%	0.229
tf pos strict	acc	100.00%	100.00%	100.00%	99.99%	100.00%	100.00%	3.56E-05
	prec	79.72%	56.48%	88.22%	34.36%	97.75%	71.31%	0.229

Table 5: Predictive accuracies and precisions of models learned and applied with different AQ21 parameters to testing data with variance 0.05.

Experiment		Edge 0	Edge 1	Edge -1	Edge 2	Edge -1	Average	St. dev.
atf 01 majority flexible	acc	92.19%	83.90%	97.93%	71.61%	99.39%	89.00%	0.102
	prec	97.15%	93.14%	97.31%	85.99%	100%	94.72%	0.048
atf 01 majority strict	acc	92.17%	83.72%	97.93%	69.96%	99.39%	88.63%	0.108
	prec	97.15%	93.14%	97.31%	89.72%	100%	95.47%	0.036
atf 01 pos flexible	acc	95.35%	92.00%	98.69%	93.38%	100%	95.87%	0.030
	prec	87.74%	58.18%	92.60%	36.18%	83.74%	71.69%	0.213
atf 01 pos strict	acc	95.35%	92.00%	98.69%	93.38%	100%	95.87%	0.030
	prec	87.74%	58.18%	92.60%	36.18%	83.74%	71.69%	0.213
tf majority flexible	acc	93.00%	85.88%	97.80%	73.57%	99.39%	89.93%	0.094
	prec	100%	100%	100%	100%	100%	100.00%	0
tf majority strict	acc	93.00%	85.88%	97.80%	73.57%	99.39%	89.93%	0.094
	prec	100%	100%	100%	100%	100%	100.00%	0
tf pos flexible	acc	98.82%	98.12%	99.66%	98.47%	99.87%	98.99%	0.006
	prec	79.67%	56.53%	88.12%	34.40%	97.75%	71.29%	0.229
tf pos strict	acc	98.82%	98.12%	99.66%	98.47%	99.87%	98.99%	0.006
	prec	79.67%	56.53%	88.12%	34.40%	97.75%	71.29%	0.229

Table 6: Predictive accuracies and precision of models learned and applied with different AQ21 parameters to testing data with variance 0.1.

Experiment		Edge 0	Edge 1	Edge -1	Edge 2	Edge -1	Average	St. dev.
atf 01 majority flexible	acc	90.62%	81.26%	96.93%	67.46%	99.00%	87.05%	0.115
	prec	97.13%	93.14%	97.30%	85.98%	100%	94.71%	0.048
atf 01 majority strict	acc	90.58%	80.89%	96.93%	65.88%	99.00%	86.66%	0.121
	prec	97.13%	93.14%	97.30%	89.72%	100%	95.46%	0.036
atf 01 pos flexible	acc	93.71%	89.15%	97.75%	89.89%	100%	94.00%	0.041
	prec	87.70%	58.18%	92.58%	36.19%	83.72%	71.67%	0.213
atf 01 pos strict	acc	93.71%	89.15%	97.75%	89.89%	100%	94.00%	0.041
	prec	87.70%	58.18%	92.58%	36.19%	83.72%	71.67%	0.213
tf majority flexible	acc	90.27%	82.10%	96.68%	67.21%	99.00%	87.05%	0.115
	prec	100%	100%	100%	100%	100%	100.00%	0
tf majority strict	acc	90.27%	82.10%	96.68%	67.21%	99.00%	87.05%	0.115
	prec	100%	100%	100%	100%	100%	100.00%	0
tf pos flexible	acc	96.49%	94.41%	98.82%	94.28%	99.44%	96.69%	0.021
	prec	79.63%	56.53%	88.11%	34.40%	97.71%	71.27%	0.229
tf pos strict	acc	96.49%	94.41%	98.82%	94.28%	99.44%	96.69%	0.021
	prec	79.63%	56.53%	88.11%	34.40%	97.71%	71.27%	0.229

6 EXPERIMENTAL EVALUATION

In order to evaluate the presented methodology we conducted experiments using the simulation system PlaSMA. Each experiment is specified by a simulation model parameter setting and 9 participating vehicle agents. Although the experiments include multiple agents running concurrently these agents do not influence each other. There is one ignorant (named "IA"), one weather-aware ("WA") and seven traffic-predicting agents ("AQ"). The latter use different models for predicting traffic conditions (see Sect. 5), as indicated by their index. The agent named AQ_M replicates a traffic-predicting agent whose rules have been created manually *knowing* the actual traffic simulation model. Thus, this agent should provide results close to the achievable optimum for the applied algorithm. Other AQ agents use models automatically learned by the AQ21 system form historic simulation data.

All agents try to optimize the driving time of a 1000 km trip in the road network starting at location $\langle \mathrm{A},\mathrm{A} \rangle$ and heading to location $\langle \mathrm{F},\mathrm{F} \rangle$. Simulation results are provided as average driving time and its standard deviation for each vehicle agent and parameter setting. For statistical significance experiments were repeated 4,200 to 5,000 times with each run corresponding to one vehicle trip. With significance level $\alpha = 0.05$ stated average values do not differ more than 0.02 hours from actual value.

The simulation model parameters used in our experiments are the weather change interval $\Delta t_{\mathcal{W}}$, the weather probability distribution $\mathbf{P}(\mathcal{W})$, the look-ahead λ for the acquired location-specific weather information, the traffic generation interval $\Delta t_{\mathcal{T}}$, the edge-specific traffic density biases $\Delta\mathrm{dens}(e)$, and the variance $\sigma_{\mathcal{T}}^2$ for traffic generation based on normal distributions. For all presented experiments we set world agent parameters to $\Delta t_{\mathcal{T}} = 1$ h and $\Delta t_{\mathcal{W}} = 3$ h. Weather probabilities are set to

$$\mathbf{P}(\mathcal{W}) = \langle \mathrm{P}(\mathcal{W}{=}\mathrm{VeryBad}){=}0.05, \mathrm{P}(\mathcal{W}{=}\mathrm{Bad}){=}0.2,$$
$$\mathrm{P}(\mathcal{W}{=}\mathrm{Moderate}){=}0.55, \mathrm{P}(\mathcal{W}{=}\mathrm{Good}){=}0.2\rangle$$

The edge specific density bias $\Delta\mathrm{dens}(e)$ was examined in two different settings with $\Delta\mathrm{dens}(e) \in [\text{-}0.2,{+}0.2]$. The first setting ($\Delta\mathrm{dens}_{\mathrm{S1}}$) is characterized by a fairly well-shuffled distribution of slower and faster edges (Figure 5). Thus, each route will incorporate nearly as many slow as fast edges and all routes will not differ much in expected driving time when ignoring weather. This setting will not exploit the potential of traffic prediction because even perfect predictions will not get very valuable in comparison to solely weather-focused predictions. So we created another setting ($\Delta\mathrm{dens}_{\mathrm{S2}}$) that equals $\Delta\mathrm{dens}_{\mathrm{S1}}$ but reduces $\Delta\mathrm{dens}(e)$ on three consecutive edges with

$$\Delta\mathrm{dens}_{\mathrm{S2}}((\mathrm{D,B})\rightarrow(\mathrm{D,C})) = \text{-}0.2$$
$$\Delta\mathrm{dens}_{\mathrm{S2}}((\mathrm{D,C})\rightarrow(\mathrm{E,C})) = \text{-}0.2$$
$$\Delta\mathrm{dens}_{\mathrm{S2}}((\mathrm{E,C})\rightarrow(\mathrm{E,D})) = \text{-}0.2$$

These edges are not part of routes that most closely match the theoretical straight line route. Hence, ignorant vehicle agents that are dominated by the straight line heuristics and always choose such centered routes will not be affected by those changes. Weather-aware agents may pass the changed edges and are thus affected. Nevertheless, weather-aware agents should not benefit as much as traffic-predicting agents because only the latter will actually realize the traffic properties and consider them in planning.

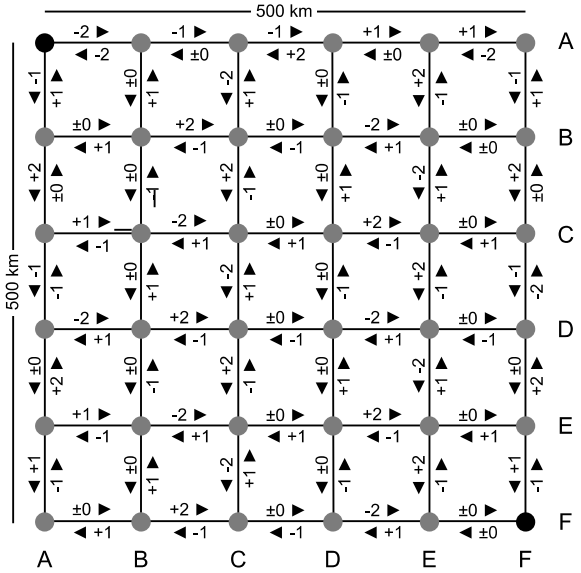

Figure 5: Traffic density bias $10{\cdot}\Delta\mathrm{dens}_{\mathrm{S1}}(\mathrm{e})$ within the road grid as difference to default densities.

6.1 Agent Comparison

The vehicle represented by the ignorant agent ($\lambda=0$) needs an average driving time 13.44±0.92 hours while traffic-predicting agents using AQ's normal theory formation mode (TF) are up to 0.74 hours or 5.5 % faster (AQ_3: 12.70±0.67 h). The standard deviation is considerably smaller, too. The agent that uses solely weather information for route assessment needs 12.87±0.89 hours on average. Thus, the AQ agents are only 1.3 % faster but there standard deviation is significantly smaller again (-24.7 %).

Table 7: Vehicle driving time in hours for experiments with $\Delta\mathrm{dens}_{\mathrm{S1}}$, $\sigma_T^2=0.01$ with look-ahead $\lambda=1$

IA	WA	AQ_M	AQ_1 TF pos, str	AQ_2 TF pos, flx	AQ_3 TF maj, str	AQ_4 TF maj, flx	AQ_5 ATF pos, str	AQ_6 ATF pos, flx	AQ_7 ATF maj, str	AQ_8 ATF maj, flx
13.44 ±0.92	12.87 ±0.89	12.72 ±0.68	12.73 ±0.70	12.74 ±0.70	12.70 ±0.67	12.72 ±0.67	12.82 ±0.67	12.83 ±0.68	12.76 ±0.69	12.77 ±0.71

The difference between weather-aware agents and AQ agents very much depends on the influence of weather on traffic. Possibly the chosen simulation model overweighs weather so the additional knowledge on traffic is not very valuable. As stated above, the mixed distribution of edges with high or low traffic in $\Delta\mathrm{dens}_{\mathrm{S1}}$ also decreases the influence of traffic-related knowledge. However, the smaller standard deviation could be explained by the AQ agents' knowledge on how to avoid roads with high traffic jam probability while weather seems to be decisive on average.

Traffic-predicting agents with TF learning mode also outperform or match the driving time of agent AQ_M with a model-aware ruleset. Thus the learned rules' quality is very high as already indicated by their precision and accuracy evaluated in Sect. 5. Agents with approximate theory formation (ATF) perform slightly worse. Especially the positive mode (*pos*) yields slower driving time. But there is no notable advantage of positive or majority (*maj*) mode when using normal theory formation (TF).

6.2 Basic Traffic Density Model with λ Variation

Weather-aware agents as well as agents that anticipate traffic and expected speed need environmental information to determine expected speed v_{est} on any road e. The look-ahead parameter λ and the vehicle location at current time t_{cur} specify the area for which these agents acquire weather information $\mathrm{weather}(e,t_{\mathrm{cur}})$. This information only reflects the current status at a location and thus it might change until the time of arrival. Anyhow, the agents are not capable of weather forecasts and just expect weather status to persist. Hence a look-ahead $\lambda>0$ seems important for planning but large values might cause wrong decisions (see Sect. 3.3).

To find the best value of λ we conducted experiments with its different settings, as outlined in Table 8. Given the weather change interval Δt_W = 3 h driving time of all vehicles decreases slightly from $\lambda=0$ to $\lambda=2$ (e.g., AQ_2: 0.08 h shorter). But for $\lambda>2$ driving time does not change

significantly for most agents. Thus, the weather information of remote locations does not influence driving time significantly in good or worse. The latter case is notable because weather information of remote locations might be obsolete and thus harmful with respect to its usage in the agents speed estimation. Probably this effect does not occur due to the agent's possibility to change the vehicle route at every junction and thereby avoiding routes that seem to be slow given *new* weather information. Other network topologies might yield other results where greater look-ahead gets more important.

Table 8: Vehicle driving time in hours for experiments with $\Delta\mathrm{dens}_{S1}$, σ_T^2=0.01 and variations of look-ahead λ

	WA	AQ_M	AQ_1 TF pos, str	AQ_2 TF pos, flx	AQ_3 TF maj, str	AQ_4 TF maj, flx	AQ_5 ATF pos, str	AQ_6 ATF pos, flx	AQ_7 ATF maj, str	AQ_8 ATF maj, flx
λ=1	12.87 ±0.89	12.72 ±0.68	12.73 ±0.70	12.74 ±0.70	12.70 ±0.67	12.72 ±0.67	12.82 ±0.67	12.83 ±0.68	12.76 ±0.69	12.77 ±0.71
λ=2	12.79 ±0.89	12.69 ±0.70	12.66 ±0.69	12.66 ±0.70	12.66 ±0.68	12.67 ±0.68	12.75 ±0.67	12.77 ±0.68	12.72 ±0.68	12.72 ±0.71
λ=3	12.79 ±0.88	12.69 ±0.70	12.65 ±0.68	12.67 ±0.70	12.68 ±0.68	12.68 ±0.68	12.77 ±0.68	12.76 ±0.68	12.72 ±0.70	12.73 ±0.71
λ=4	12.79 ±0.87	12.69 ±0.70	12.67 ±0.69	12.65 ±0.67	12.68 ±0.67	12.68 ±0.69	12.76 ±0.67	12.77 ±0.67	12.72 ±0.69	12.73 ±0.71

6.3 Influence of Traffic Density Model

In order to reduce and analyze the possible effects of the traffic density bias model $\Delta\mathrm{dens}_{S1}$ we compare driving time with second model $\Delta\mathrm{dens}_{S2}$. Since λ=2 has turned out to be best for Δt_W=3 h simulation results for other λ values have been omitted here. Table 9 shows the simulation results for both traffic density settings with density variance σ_T^2 = 0.01. The second density setting should provide better driving time for all weather-aware and/or traffic-predicting vehicle agents because density bias $\Delta\mathrm{dens}$ is reduced for three edges that are connected to a fast potential partial route from ⟨A,A⟩ to ⟨F,F⟩. In this setting, the ignorant agent still needs 13.44±0.92 hours driving time because the changed edges do not intersect with its routes that are optimized to straight line distance (see Sect. 3.2). The weather-aware agent needs 12.74±0.86 h, i.e., it is 0.05 h faster than with model $\Delta\mathrm{dens}_{S1}$. As in the previous density setting, the prediction ruleset AQ_3 performs best with 12.63±0.66 h (0.03 h better). In comparison to the first traffic density setting results are indeed slightly better. However, the differences are very small and AQ agents do not benefit as much as expected. Again, this is owing to the strong influence of weather. Furthermore, changes concern only 15 % of the accelerated route (with return route).

Table 9: Experiments with λ=2, σ_T^2 =0.01 and $\Delta\mathrm{dens}$ variation.

$\Delta\mathrm{dens}$	IA	WA	AQ_M	AQ_1 TF pos, str	AQ_2 TF pos, flx	AQ_3 TF maj, str	AQ_4 TF maj, flx	AQ_5 ATF pos, str	AQ_6 ATF pos, flx	AQ_7 ATF maj, str	AQ_8 ATF maj, flx
S1	13.44 ±0.92	12.79 ±0.89	12.69 ±0.70	12.66 ±0.69	12.66 ±0.70	12.66 ±0.68	12.67 ±0.68	12.75 ±0.67	12.77 ±0.68	12.72 ±0.68	12.72 ±0.71
S2	13.44 ±0.92	12.74 ±0.86	12.64 ±0.68	12.63 ±0.68	12.63 ±0.68	12.63 ±0.66	12.63 ±0.66	12.75 ±0.67	12.74 ±0.64	12.70 ±0.66	12.69 ±0.71

6.4 Impact of Traffic Density Variance

Increasing variance σ_T^2 for stochastic generation of traffic density should result in longer driving time and greater standard deviations for all agents. More frequently, traffic-predicting agents will misjudge traffic and thus choose routes that have high traffic. Although the estimations may also be worse than actual driving time in some cases the negative underestimated cases outweigh the positive cases. This is due to the preference of roads that are seemingly good without much possible potential to be better, i. e., the maximum vehicle speeds restricts the influence of higher variance in positive direction but not in negative direction.

Table 10: Experiments with traffic model $\Delta\mathrm{dens}_{S2}$, λ=2 and σ_T^2 variation.

σ_T^2	IA	WA	AQ_M	AQ_1 TF pos, str	AQ_2 TF pos, flx	AQ_3 TF maj, str	AQ_4 TF maj, flx	AQ_5 ATF pos, str	AQ_6 ATF pos, flx	AQ_7 ATF maj, str	AQ_8 ATF maj, flx
0.001	13.31 ±0.86	12.61 ±0.78	12.51 ±0.60	12.51 ±0.64	12.56 ±0.63	12.47 ±0.60	12.50 ±0.61	12.62 ±0.60	12.63 ±0.64	12.54 ±0.64	12.57 ±0.65
0.01	13.44 ±0.92	12.74 ±0.86	12.64 ±0.68	12.63 ±0.68	12.63 ±0.68	12.63 ±0.66	12.63 ±0.66	12.75 ±0.67	12.74 ±0.64	12.64 ±0.66	12.69 ±0.71

Table1 10 shows simulation results for different settings of σ_T^2 backing this expectation for traffic density setting is $\Delta\mathrm{dens}_{S2}$. Due to a higher probability of wrong predictions agent AQ_3 performs 0.16 hours worse with highest variance in comparison to lowest variance. This change is rather small and both situation-aware agents still clearly outperform the ignorant agent. Thus, the agents can be considered sufficiently robust to more dynamic and less predictable environments.

7 CONCLUSIONS

This paper presented an approach to the problem of predicting traffic for vehicle routing. Autonomous agents make predictions based on rules induced by the AQ21 system from historic data. By inducing attributional rules AQ21 realizes natural induction that is an inductive learning process whose results are both accurate and easy to understand by people. Experiments performed within the PlaSMA multiagent-based simulation system indicated advantage of agents that use AQ21 predictions over naïve agents that consider only distance to the destination and agents that use only weather-related information.

Future research includes comparison with different learning methods, investigation of the effect of the amount of historic data on the learning results and predictions, and investigation of the effects of changes of the environment on the predictions. Other important research directions include learning individualized models for each agent, based on the agent's experience and preferences, and application of the system in real environment.

REFERENCES

ARV, “Verordnung über eine allgemeine Richtgeschwindigkeit auf Autobahnen und ähnlichen Straßen (ARV)”. BGBl. I, p. 1824., 1978, updated 1997 (German federal by-law.)

Endsley, M. R., “Theoretical Underpinnings of Situation Awareness: A Critical Review”. In Endsley, M. R., Garland, D. J. (eds.), *Situation Awareness Analysis and Measurement*. Lawrence Erlbaum Assoc., Mahwah, NJ, 2000.

FGSV (ed.), “Handbuch für die Bemessung von Straßenverkehrsanlagen”. FGSV-Verlag, Köln, 2005. (German pendant of US “Highway Capacity Manual”, HCM.)

Gehrke; J. D. and Ober-Blöbaum, C., “Multiagent-based Logistics Simulation with PlaSMA”. In *Informatik 2007. 37. Jahrestagung der Gesellschaft für Informatik*, Sept. 24–27, 2007.

Hart, P. E., Nilsson, N. J., and Raphael, B., “A formal basis for the heuristic determination of minimum cost paths”. *IEEE Transactions on Systems Science and Cybernetics*, SSC-4(2), pp. 100–107, 1968.

Hastie, T., Tibshirani, R., and Friedman, J. H., *The Elements of Statistical Learning*, 3rd Edition, Springer, 2003.

HCM, “Highway Capacity Manual (HCM)”. Transportation Research Board, 2000

Kaufman, K. and Michalski, R. S., “An Adjustable Rule Learner for Pattern Discovery Using the AQ Methodology”, *Journal of Intelligent Information Systems*, 14, pp. 199–216, 2000.

Michalski, R. S., "ATTRIBUTIONAL CALCULUS: A Logic and Representation Language for Natural Induction," *Reports of the Machine Learning and Inference Laboratory*, MLI 04-2, George Mason University, Fairfax, VA, April, 2004.

Michalski, R. S. and Wojtusiak, J., "Generalizing Data in Natural Language," *Proceedings of the International Conference Rough Sets and Emerging Intelligent Systems Paradigms*, RSEISP'07, Lecture Notes in Computer Science, Springer, 2007.

Scholz-Reiter, B., Windt, K. and Freitag, M, “Autonomous logistic processes: new demands and first approaches”. In Monostri, L. (ed.), *Proceedings of the 37th CIRP International Seminar on Manufacturing Systems* (Budapest, Hungary), pp. 357–362, 2004.

Wojtusiak, J., "AQ21 User's Guide," *Reports of the Machine Learning and Inference Laboratory*, MLI 04-3, George Mason University, Fairfax, VA, September, 2004 (updated in September, 2005).

Wojtusiak, J., Michalski, R. S., Kaufman, K. and Pietrzykowski, J., "The AQ21 Natural Induction Program for Pattern Discovery: Initial Version and its Novel Features," *Proceedings of The 18th IEEE International Conference on Tools with Artificial Intelligence*, Washington D.C., November 13–15, 2006.

Wörner, J.; Wörn, H., “Benchmarking of Multiagent Systems in a Production Planning and Control Environment”. In Kirn, S., Herzog, O., Lockemann, P., Spaniol, O. (eds.), *Multiagent Engi-neering: Theory and Applications in Enterprises*, pp. 115–133. Springer-Verlag, 2006.

A publication of the *Machine Learning and Inference Laboratory*
George Mason University
Fairfax, VA 22030-4444 U.S.A.
http://www.mli.gmu.edu

Editor: Janusz Wojtusiak

The *Machine Learning and Inference (MLI) Laboratory Reports* are an official publication of the Machine Learning and Inference Laboratory, which has been published continuously since 1971 by R.S. Michalski's research group (until 1987, while the group was at the University of Illinois, they were called ISG (Intelligent Systems Group) Reports, or were part of the Department of Computer Science Reports).