Enabling the Use of a Planning Agent for Urban Traffic Management via Enriched and Integrated Urban Data

Grigoris Antoniou (G.Antoniou@hud.ac.uk)
University of Huddersfield, Queensgate, Huddersfield, HD1 3DH, United Kingdom

Sotiris Batsakis (S.Batsakis@hud.ac.uk)
University of Huddersfield, Queensgate, Huddersfield, HD13DH, United Kingdom
Technical University of Crete, Chania 731 00, Greece

John Davies (john.nj.davies@bt.com)
British Telecommunications, Adstral Park, Ipswich, IP5 3RE, United Kingdom

Alistair Duke (alistair.duke@bt.com)
British Telecommunications, Adstral Park, Ipswich, IP5 3RE, United Kingdom

Thomas L. McCluskey (t.l.mccluskey@hud.ac.uk)
University of Huddersfield, Queensgate, Huddersfield, HD1 3DH, United Kingdom

Evtim Peytchev (evtim.peytchev@gmail.com)
Nottingham Trent University, 50 Shakespeare St, Nottingham NG1 4FQ, United Kingdom

Ilias Tachmazidis (I.Tachmazidis@hud.ac.uk)
University of Huddersfield, Queensgate, Huddersfield, HD1 3DH, United Kingdom

Mauro Vallati (m.vallati@hud.ac.uk) corresponding author
University of Huddersfield, Queensgate, Huddersfield, HD1 3DH, United Kingdom

Abstract
Improving a city's infrastructure is seen as a crucial part of its sustainability, leading to efficiencies and opportunities driven by technology integration. One significant step is to support the integration and enrichment of a broad variety of data, often using state of the art linked data approaches. Among the many advantages of such enrichment is that this may enable the use of intelligent processes to autonomously manage urban facilities such as traffic signal controls.
In this paper we document an attempt to integrate sets of sensor and historical data using a data hub and a set of ontologies for the data. We argue that access to such high level integrated data sources leads to the enhancement of the capabilities of an urban transport operator. We demonstrate this by documenting the development of a planning agent which uses such data as inputs in the form of logic statements, and when given traffic goals to achieve, outputs complex traffic signal strategies which help transport operators deal with exceptional events such as road closures or road traffic saturation. The aim is to create an autonomous agent which reacts to commands from transport operators in the face of exceptional events involving saturated roads, and creates, executes and monitors plans to deal with the effects of such events. We evaluate the intelligent agent in a region of a large urban area, under the direction of urban transport operators.

1 Introduction

The central idea of smart city initiatives is to use technology to improve the efficiency, effectiveness and capability of various city services, thus improving the quality of the inhabitants' lives (Townsend 2013). Recent smart city initiatives are emerging all over the world. A fundamental difference between smart cities and similar uses of technology in other areas, like business, government or

education, is the vast variety of the technologies used, the types and volumes of data, and the services and applications targeted (d'Aquin et al. 2015). To develop successful smart city solutions, there is a need to (i) collect and maintain relevant data, both static and dynamic in the form of IoT data, and (ii) use a variety of technologies for improving decision making and support and optimizing solutions and services. The complexity and diversity of relevant data in smart cities is one of the main reasons why they have emerged as key use cases for linked data (Bizer et al. 2009) and Semantic Web technologies (Berners-Lee et al. 2001). Linked data is well suited for data integration because it allows integration and unified access through common, high-level vocabularies, while leaving the original data distributed and managed under the control of their original contributors (d'Aquin et al. 2015). In addition, the use of linked data and Semantic Web technologies allows the semantic enrichment of data by linking them with additional information, thus providing context and also aiding data cleaning.

The work reported here was one of the deliverables of a funded project called "SimplyfAI" which was carried out by a consortium consisting of a university (University of Huddersfield), a major transport authority (Transport for Greater Manchester – TfGM), a large technology supplier (British Telecommunications - BT), one SME performing the project management (KamFutures Ltd), and another SME

providing independent validation of the results (InfoHub). Thus the paper reports on research results and their evaluation in a real setting: the problem area, the data, the validation and the management were all performed to a large part independently of the research group at the centre of the project. The overall aims were in the context of developing smart city technology, taking advantage of the wide range of data available in a modern urban area. In particular the project focussed on exploiting the real-time and historical data sources to pursue better congestion control during times when the road network was saturated.

In general, urban traffic management and control (UTMC) is a primary concern of any city, and urban traffic transport operators often have at their disposal a disparate variety of real time and historical data, traffic controls (the most common of which are traffic signals) and controlling software. They use these assets to perform the crucial role of tackling road congestion, minimising delay to the road-based traveller, and minimising traffic related environmental effects. Like in many organisations, however, the software systems used in UTMC tend to be based around a syntactic, product specific integration of data, which at best shares data externally at a relational database level. They adopt a vertical systems design, and though eminently configurable within the range of their function, they are not integrated at a horizontal level with the overall function of the UTMC centre where they operate. Within UTMC operations this perpetuates the status quo of recurrent system replacement, rather than system evolution and data and software reuse.

To counter this status quo, the project was engaged in developing semantic technology in order to better capture and exploit real-time and historical urban data sources, while pursuing the higher level of data integration. We aim to make UTMC systems less brittle and more adaptable by raising the level of traffic control software integration via semantic component inter-operability. In doing this we have the longer-time aim of utilising an autonomic approach to UTMC in particular, and road transport support in general, as developed in the EU's transport network ARTS[1] (McCluskey et al. 2016). Results of the Network supported the idea of the construction of a semantic systems level for UTMC, consistent with work on integrating decision support within semantic technologies (Blomqvist 2014, Antunes et al. 2016). Among the benefits of a higher level of information integration is a more joined up UTMC capability, where the flexibility of a knowledge level representation gives the opportunity to use intelligent agents to provide a more autonomous approach to tackle UTMC issues. Indeed, one of the enablers of deploying agents that can reason with the knowledge of an application area is that the data available embodies its own semantics in the consistent use of meta-data.

1 EU's COST Action 1102 "Towards Autonomic Road Transport Support Systems" (ARTS)

Within this context, we present a novel planning agent which addresses a well known functional drawback of established UTMC tools referred to above: they do not work adequately in the face of exceptional or unexpected conditions affecting urban regions (containing many hundreds or thousands of road vehicles). Hence the contributions of the paper are in the demonstrations of how:

- the diversity of traffic-related data can be represented using semantic technologies, to enable the integration into a unified form through a common, high-level vocabulary;
- a planning agent can take advantage of the semantically-enriched data to assist in better control and decision making in traffic management;
- the planning agent can generate traffic control strategies (actions which change signals at a specified time) in real time in response to detection of traffic congestion caused by exceptional circumstances in an area of Manchester, UK.

The quality of the strategies output from the planning agent was evaluated firstly by hand inspecting the strategies to check that they were "sensible". In this case the strategies were inspected by transport operators, to check they embodied common sense uses of the traffic signal within them. Secondly, their effect was compared against optimised strategies derived from historical data by simulating their execution using both the AIMSUN micro-modelling software[2], and the off-the-shelf SUMO (Krajzewicz et al. 2012) micro-modelling software. In each, transport engineers compared the results of simulations using both automated planning generated strategies, and optimised strategies derived from historical data. In both these simulators, run by different members of the consortium, the agent-generated strategies produced noticeable savings - approximately a 20% efficiency gain. Given we know of no other operational system that generates region wide strategies for exceptional events in busy urban areas, the implications of taking this approach are, we believe, of a step changing nature for UTMC. Additionally, while this application demonstrates what can be done currently, we perceive a range of benefits from the existence of the integration of a wide range of data, expressed at a semantic level. Currently we are engaged in the adaptation of the system to deal with additional effector actions (rather than only controlling traffic signal change) and a more sophisticated, flexible goal language with which the operators can communicate with the agent.

2 Creation of a Semantic Level for Traffic-related Data

This section is focused on the semantic enrichment of traffic-related data.

2.1 Overview

The initial focus of the SimplyfAI project concentrated on the semantic enrichment of traffic data in a collaboration involving university academics and the technology provider, BT. In broad outline, the raw data was taken from transport and environment sources and integrated into a data hub3, using semantic technologies such as the universal RDF triple format and a data ontology. The method was to take real time feeds and process them until they produced logical facts about a traffic scenario, which serves as the dynamic data input to the planning agent. These facts contain the real-time occupancy of road links (a link is a uni-directional part of a road between two intersections) and the signal phase of the traffic lights in intersections connecting the road links within an operator-defined region.

Introducing machine readable semantics for data sharing and integration calls for a formal language for conceptualization of application domains, the related concepts, their properties and their relationships. Based upon existing work on knowledge representation, logic and ontologies, RDF (Hayes 2004) forms the basis for Semantic Web standards for data representation, while RDFS and OWL are used for defining concepts, their properties and their relationships. OWL 2 (Hitzler et al. 27 October 2009) is the current standard for ontology definitions. These definitions can reference and

2 http://aimsun.com
3 http://portal.bt-hypercat.com/

reuse existing definitions and data forming distributed interconnected datasets (Linked Data), which are typically open (Linked Open Data-LOD)[4].

2.2 Data Hub

In SimplyfAI, the data hub provides a focal point for the sharing and consumption of related datasets, such as traffic data. The role of the data hub is to enable information from various sources to be brought onto a common platform. Its portal provides a direct interface through which data consumers, such as app developers, can browse a data catalogue and select and subscribe to data feeds that they want to use. In addition, a JSON-based Hypercat (Beart 2016) machine-readable catalogue, described further below, is also provided (as well as a recently proposed RDF-based Hypercat (Tachmazidis et al. 2016) catalogue). An API enables access to data feeds, secured by API keys, from browsers or within computer programs, while a relational, GIS capable, database enables complex queries that data can be filtered according to a wide range of criteria.

A set of adapters enables information coming onto the hub to be converted to a standard format for use inside the platform's core. It also provides a consistent API to end users and developers. The hub provides a consistent approach to integration between data exposed by sensors, systems and individuals via communication networks and the applications that can use derived information to improve decision making, e.g., in control situations that we elucidate in Section 3 below. It includes a set of adapters for ingress (input) and egress (output). These are potentially specific to each data source or application feed and may be implemented on a case by case basis. There is therefore a need to translate data between arbitrary external formats and the data formats used internally.

[Figure 1 about here] Top Level Concepts.

In addition, as mentioned above, a Hypercat catalogue is implemented which is included via the Hypercat API. Hypercat is in essence a standard for representing and exposing Internet of Things data hub catalogues over web technologies. The idea is to enable distributed data repositories (data hubs) to be used jointly by applications through making it possible to query their catalogues in a machine readable format. This enables the creation of "knowledge graphs" of available datasets across multiple hubs that applications can exploit and query to identify and access the data they need, whatever the data hub in which they are held.

2.3 Data Enrichment

While Hypercat offers a syntactic first step, providing semantically enriched data, it also allows the unique identification of existing resources, interoperability across various domains and further enrichment by combining internally stored data with the Linked Open Data cloud. Developing an ontology that provides definitions to required concepts at a level of complexity that allows fast reasoning over big data is the main challenge during this work at the semantic level. Thus, data enrichment in the data hub is achieved by representing data in RDF using concepts and properties defined in an OWL ontology. Figure 1 shows the top level concepts of the developed ontology. A major design decision in this work is to propose a minimalistic representation that can be the basis of an efficient reasoning mechanism over Big Data (e.g., IoT applications) thus reusability of existing definitions on other ontologies (e.g., definition of lat/long coordinates) and axioms were restricted to the minimum required. Additional axioms and links to existing definitions will be added in future applications if necessary, but reasoning and querying performance is a factor that must be taken into account.

In the developed ontology, *Feed* is the top level class for any data feed that is asserted in the knowledge base. It contains the semantic properties of feeds. These include the feed id, creator, update date, title, url, status, description, location name, domain and disposition. There are also subclasses

4 http://lod-cloud.net/

of class *Feed*, namely: *SensorFeed*, *EventFeed* and *LocationFeed* representing feeds for sensors, events and locations respectively.
Data that has been modelled for the SimplyfAI project has been incorporated in the data hub as one of the following feed types: (a) *SensorFeed*, (b) *EventFeed*, and (c) *LocationFeed*. Practically, each data source can advertise available information through the hub by providing a feed. A feed should be understood as a source of sensor readings, events or locations. Within each feed, data is available through datastreams (a class *Datastream* is defined, which has two subclasses namely: *SensorStream* and *EventStream* representing datastreams for sensors and events respectively). Thus, a given feed may provide a range of datastreams that are closely related e.g., for a weather data feed, different datastreams may provide sensor readings for temperature, humidity and visibility. Considering information about locations, a feed (of type *LocationFeed*) provides information directly by returning locations (note that class *Location* models any given location), namely locations are attached to and provided by a given feed. Note that time-stamping feeds in conjunction with their location provides the required dynamic information for the planning agent.
A Hypercat online catalogue[5] contains details of feeds and information sources along with additional metadata such as tags, which allow improved search and discovery. The developed semantic model enables a semantic annotation and linkage of available feeds and datastreams. An OWL ontology has been developed and made available with the uri:

http://portal.bt-hypercat.com/ontologies/bt-hypercat

By defining an ontology, semantically enriched data can be provided in RDF format, while prior to the SimplyfAI project only XML and JSON formats were available. RDF data is represented in N-Triples[6] format since such a format facilitates both storage and processing of data. Thus, following W3C standards ensures interoperability and enables the utilization of existing tools and applications. By providing semantically enriched data through the data hub, the developed approach enables reusability of existing information, which in turn facilitates future extensions (e.g., modelling an entire traffic network). In this way, data imports to services (such as the planning agent, described in Section 3) are scalable in terms of the size of the network, namely when moving from a specific traffic area to a large traffic network. Note that handling large traffic networks would impose significant challenges in case of an ad hoc data management.
One of the key advantages of semantic representation over traditional DBs (and the corresponding SPARQL queries over SQL queries) is that the semantic representation (and SPARQL queries) incorporate semantic reasoning within the returned results. For example, classes *SensorFeed* and *EventFeed* are subclasses of class *Feed*. Thus, the reasoner classifies all objects that belong to either *SensorFeed* or *EventFeed* as *Feed*.
The data hub includes additional adapters for egress (output) in order to provide data in RDF format. Here is an example of how subject, predicate and object are generated for a *SensorFeed*. Initially, the URI of each *SensorFeed* is generated, namely

<http://api.bt-hypercat.com/sensors/feeds/feedID>

Note that “http://api.bt-hypercat.com/” is the prefix URI for any data provided by the data hub. In addition, “/sensors” provides information about the type of the feed (here *SensorFeed*), followed by “/feeds”, which indicates that this URI belongs to a resource describing a feed, and finally “/feedID” is an id that uniquely identifies the given feed. For each *SensorFeed*, the data hub provides its type, namely:

Subject: <http://api.bt-hypercat.com/sensors/feeds/feedID>
Predicate: <http://www.w3.org/1999/02/22-rdf-syntax-ns#type>
Object: <http://portal.bt-hypercat.com/ontologies/bt-hypercat#SensorFeed>

5 http://portal.bt-hypercat.com/cat -- http://portal.bt-hypercat.com/cat-rdf
6 https://www.w3.org/TR/n-triples/

In this application, *SensorFeeds* (as well as other types) are tagged when added into the knowledge base, while the generation of other triples follows a similar rational. Note that due to performance reasons the ontology contains definitions that are necessary in this application. However, the proposed semantic representation allows for further enrichment, using federated queries (Tachmazidis et al. 2017).

2.4 Extracted Data

Data for several automated traffic counting locations within the SimplyfAI study area is available. The data indicates the number of vehicles passing the site in the previous 5 minute period. A separate count for each road is provided as a datastream of the feed. The feeds are available via the data catalogue (e.g., search for "Automatic Traffic Count"). Access to the data itself is currently restricted but can be made available to users on request. Once subscribed, users can access the data via URIs such as the following (a URL including the feed id for the Automatic Traffic Count site in question):

http://api.bt-hypercat.com/sensors/feeds/aecb7ce3-d537-436f-a485-7f8f7436cdad/

Each feed has datastreams corresponding to each road e.g., for site 1202 in Dawson St there are two datastreams:

Stream id: 1 **Data**: Northbound
Stream id: 2 **Data**: Westbound

The result of the http request will show the most recent datapoint for the specified datastream i.e., the number of vehicles in the last 5 minutes. The following XML response shows that there were 180 vehicles in the preceding 5 minute period:

```
<datastream id="2">
        <tag>Westbound</tag>
        <current_time>Fri, 06 May 2016 15:57:31 GMT</current_time>
        <current_value>180</current_value>
</datastream>
```

A query giving an example of how the system can gather information from sensors (the example query is a federated query spanning over different SPARQL endpoints and it retrieves all sensor measurements close to a specific point of interest), which will be subsequently used for populating the dynamic data of the Planning Agent's input file, is as follows:

```
PREFIX geo: <http://www.w3.org/2003/01/geo/wgs84_pos#>
PREFIX hypercat: <http://portal.bt-hypercat.com/ontologies/bt-hypercat#>
PREFIX naptan: <http://transport.data.gov.uk/def/naptan/>
PREFIX skos: <http://www.w3.org/2004/02/skos/core#>
SELECT distinct ?d ?at_time ?western_longitude ?southern_latitude
?eastern_longitude ?northern_latitude ?stop ?lat ?long
WHERE {
SERVICE <http://gov.tso.co.uk/transport/sparql>
{
?stop a naptan:CustomBusStop;
naptan:naptanCode ?naptanCode;
naptan:stopValidity ?stopValidity;
naptan:street "Kingswood Road";
geo:lat ?lat;
geo:long ?long.
?stopValidity naptan:stopStatus ?stopStatus.
```

```
?stopStatus skos:prefLabel "Active"@en.
}
SERVICE <http://portal.bt-hypercat.com/BT-SPARQL-Endpoint/sparql>
{
?d a hypercat:Datapoint.
?d hypercat:datapoint_at_time ?at_time.
?d hypercat:datapoint_western_longitude ?western_longitude.
?d hypercat:datapoint_southern_latitude ?southern_latitude.
?d hypercat:datapoint_eastern_longitude ?eastern_longitude.
?d hypercat:datapoint_northern_latitude ?northern_latitude.
FILTER (?western_longitude > ?long - 0.1)
FILTER (?southern_latitude > ?lat - 0.1)
FILTER (?eastern_longitude < ?long + 0.1)
FILTER (?northern_latitude < ?lat + 0.1)
}
FILTER(BOUND(?d))
}
```

Data are extracted from the data hub using SPARQL queries. We relied on the technique presented by Tachmazidis et al. (2017), where federated queries are allowed via the definition of a number of different types of SPARQL endpoints. The previous query retrieves sensor measurement data close to a specific bus stop (this is a federated query spanning over different SPARQL endpoints). It is worth reminding that in this work we do not define a mapping to a domain-specific ontology (e.g., a Planning ontology): such a mapping must be defined for each application domain, followed by extraction of relevant data using the application specific ontology. Instead, here we rely on the previously presented domain-independent ontology for the extraction of relevant data, that is then appropriately wrapped and encoded – as described in Section 3.2– in order to allow the Planning Agent to perform automated reasoning.

3 Utilisation of Semantic Data in a Strategy-Generating Planning Agent

The overall concept in the improvement of traffic management is to utilise the semantically enriched data referred to in the previous section, to enable the use of an intelligent function which requires both the integration of traffic data from disparate sources, and the transformation of the data into a predicate logic level, in order to operate. The intelligent function was to create traffic signal strategies in real time to solve challenges caused by exceptional or unexpected conditions. Conventional road traffic signal management techniques, such as traffic-responsive systems like SCOOT (Taale et al. 1998) and SCATS (Chong-White et al. 2012), or fixed time light strategies optimised using historical data, work reasonably well in normal or expected conditions. When a capacity changing event such as a road closure occurs, however, they are known not to be optimal. Additionally, they are not helpful if the transport manager would like to change the distribution of traffic for some other reason than to minimise delay. For example, given certain weather conditions and traffic concentrations, it can be predicted if the legal limit of pollutants may be breached in some traffic region. Transport operators would then set the goal of the intelligent planning agent as the lowering of traffic concentrations in this region to avoid the pollution event.

The reasoning system employed within the agent is capable of generating signal strategies using planning (Ghallab et al. 2004). The planning agent is provided with a general description of the causality in the environment by knowledge engineers. This contains a formal description of available actions, processes and event and is called the “domain model” (refer to the architectural view of Figure 2). Also, the planning agent has situational awareness provided by the data hub (Dynamic Data in Figure 2) and existing databases (Static Data in Figure 2), with goals provided by the transport operators. The agent will synthesise strategies from groundings of the actions in space and time in

order to achieve the goals (for example changing the timings of traffic signals at certain intersections in the network at certain time points in the future).
The operational concept is that, when an exceptional situation or predicted problem was spotted, the transport operators engage the agent to output a solution strategy. In the future, working in a fully autonomous mode, we foresee that this may be replaced by a trigger which automatically engages the agent. In order for the agent to do this the data hub would supply all the relevant dynamic data describing its current situation. The agent is invoked to solve the problem, and a solution strategy is generated (tests below show this generation takes up to 30 seconds), then the normal fixed time strategy would be turned off, and replaced by the agent-generated strategy. When the strategy had achieved the goal, the fixed time strategy would be turned back on. To ensure that the strategy (which may take minutes or hours to enact) was up-to-date, the data hub provides the real time data to effect monitoring. This means that the agent's strategy's simulated effect would be compared to the actual situation on the network. If this comparison returned a close similarity, the strategy would continue. If the simulated effect and the actual situation on the network were very different, then the agent would be re-tasked to produce an updated strategy, and this would be enacted.

3.1 Input Data

The input data has to be gathered from a traffic region whose geographical limits are defined by the transport operators, wherein the problem event lies. The data for this region required by the planning agent falls into two types: the first type of data is real time situational data. In our case, for a specific time, the number of vehicles on each link and the current traffic signal position. This makes up the dynamic situational awareness of the agent, and is what, via its actions, the agent can have some effect on. The data gathering and semantic enrichment processes in the previous section needs to provide an accurate and complete description of this dynamic information, via the data hub, at a specific time. An example of how this dynamic data is extracted is given above in section 2.4.
The second is the description of the fixed environment of the region, such as road layout, signal locations and signal phase definitions. This gives a situational awareness of the fixed or static knowledge for an agent (fixed in the sense that the agent can not change it). In detail this contains:

1. the topology of the road links;
2. the physical vehicle capacity of all the road links (in numbers of "passenger car units" –pcu– which takes into account the differing size of vehicles);
3. intergreen timings between each of the phases of the signals. Intergreen intervals are used between two traffic light phases for clearing the intersection from vehicles, and allowing pedestrian crossings. The duration of intergreens often vary – they could be as much as 30 seconds for pedestrian crossings across a busy intersections, or 0 seconds if the difference between 2 phases was the green lighting of a filter arrow. Intergreens have a fixed length, which cannot be modified by the agent (or by traffic controllers);
4. the specification of the phases of signals (that is which paths across an intersections are on green light during each phase, and the minimum and maximum time that a signal phase can be set for;
5. the average traffic flows between links within each phase of traffic lights, in number of pcu's per second. In other words, for each phase, the average number of vehicles (in pcus) that cross between two links during a time interval;
6. the traffic flows into and out of a region.

The data for 1–4 is available to the agent through UTC from current data bases including historical data and SATURN data. The data for 6 can be computed from the known traffic demand via an original-destination (OD) matrix using historical data. The data for 5 can also be pre-computed, although the value we require is not straightforward. Two flow rates for each link to link flow through a intersection during a particular phase can be considered: (i) the maximum flow rate –that is the maximum number of pcus, taking into account any opposing flows, that can travel from one link to another. This maximum could be reached during periods of saturation which typically happen in times

of capacity-reducing events or events causing extra loading. (ii) The flow rate calculated by simulating the traffic flow using the OD matrix over the OD's period of time (typically one hour), and calculating the consequent flow rates across intersections during phases. This value translates the intentions of the traffic captured in the OD matrix to the individual flows between intersections, and will be some percentage of the maximum flow rate. In the case of our tests, given we were concentrating on solving problems during exceptional traffic flows, we chose to use value (i) for intersections which had waiting traffic.

These static and dynamic data provide the agent its situational awareness, and this is automatically translated into a state file in a very expressive formal language called PDDL+ (Fox & Long 2006). An overview of the SimplyfAI architecture, showing the position of the PDDL+ state, and transport operators, is shown in Figure 2.

[Figure 2 about here]

Goals are presented to the agent either by transport operators, or by a trigger that is made true by inspection of the situation. Currently we are experimenting with hand-created goals, as a first stage before the introduction of triggers. The goals are made up of numerical expressions corresponding to saturation levels on sets of links.

3.2 Construction of the Agent's Knowledge Base

In contrast to the situational information, the persistent knowledge used within the agent was engineered by planning experts to capture the physics of the traffic signals, and the progression of traffic flows across intersections. This knowledge was captured in a "domain model" using PDDL+, shown in Figure 2. This language has constructs to represent hybrid states (which have discrete and numeric state variables) and dynamical objects such as processes, events and actions. We formalise the traffic problem within PDDL+ as follows.

A region of the *road network* can be represented by a directed graph (the topology of Section 3.1, item 1. above), where edges stand for road links and vertices stand for *intersections*. One vertex is used for representing the *outside* of the modelled region. Intuitively, vehicles enter (leave) the network from road links connected with the outside. Each road link has a given maximum *capacity*, i.e. the maximum number of vehicles that can be in that link (introduced in Section 3.1, item 2. above), and the current number of vehicles of a road link, which is denoted as an *occupancy* (part of the dynamic data introduced in Section 3.1). In our formulation, we consider only intersections that are controlled by traffic lights, as they are those under the control of traffic controllers.

Traffic in intersections is distributed by *flow rates* that are defined between each couple of road links (introduced in Section 3.1, item 5. above). Given two road links r_x , r_y , an intersection i, and a traffic phase p such that r_x is an incoming road link to the intersection i, r_y is an outgoing road link from i, and the flow is active (i.e., has green light) during phase p. Flow rates stand for the maximum number of vehicles that can leave r_x (hence we use interpretation (i) of Section 3.1), pass through i and enter r_y per time unit. For the sake of simplicity, we assume that vehicles going in the same direction move into the correct lane, thus not blocking other vehicles going in the different directions.

Intersections are described in terms of a sequence of traffic phases. Specifically, intersections *contain* a traffic phase, and traffic phases are connected using a *next* predicate. According to the active traffic phase, one (or more) flow rates are activated, corresponding to the traffic lights that are turned green. For each traffic phase, the *minimum* and *maximum* phase length is specified (introduced in Section 3.1, item 4. above). Within this range, the agent can decide whether to stop the phase currently active, or not. Between two subsequent traffic phases, an *intergreen* interval is specified (introduced in Section 3.1, item 3. above).

The model was encoded so that intersections are either controllable (by the agent), or not under the control of the agent. Intersections are regulated using the following PDDL+ constructs:

- An action *switchPhase(p,i)* is used by the agent for stopping the currently active phase p in intersection i, if the intersection i is controllable, and minimum phase time of p (increased by

the *keepPhase* process) has been reached. This action is the "tool" allowing the agent to affect the traffic flows. The only effect of this action is of activating a *trigger* for the intersection *i*.

- An event *triggerCatcher(p,i)* is activated when the trigger of intersection *i* is activated, during the traffic phase *p*. The event stops the current traffic phase, resets the *phaseTime* to zero, and turns on the subsequent intergreen phase.
- A process *keepPhase(p,i)* is used for "keeping" the traffic phase *p* on intersection *i* active, and measuring the time the phase is kept on. This process is started when the *activePhase* predicate of *p* is set to true, and automatically stops when the phase time has reached the maximum allowed value, or the phase has been de-activated by the agent. Similarly, a *keepIntergreen(p,i)* process is used for keeping the intergreen, subsequent to the traffic phase *p*, active.
- An event *maxPhaseTimeReached(p,i)* is triggered when the phase *p* of intersection *i* reaches the maximum allowed time (the *keepPhase* process). The event activates the trigger predicate of *i* (in the same way as the *switchPhase* action does). A corresponding *maxIntergreenTimeReached(p,i)* is used for stopping an intergreen phase when the maximum time has been reached.
- A process *flowPhase(p,r1,r2)* is activated when the *keepPhase(p,i)* process is active. It is used for moving vehicles from road *r1* to road *r2* at the given flow rate. If there is no vehicle on *r1*, or *r2* is full (the number of the vehicles is the same as the capacity of *r2*), the process is stopped.

The PDDL+ encoding of the *switchPhase* action, the *triggerCatcher* event and the *keepInterGreen* process is as follows:

```
(:action switchPhase
        :parameters (?p - phase ?i - intersection)
        :precondition (and
                (controllable ?i)
                (activePhase ?p)
                (contains ?i ?p)
                (> (phaseTime ?i) (minPhaseTime ?p) ))
        :effect (and
                (trigger ?i) ))

(:event triggerCatcher
        :parameters (?p - phase ?i - intersection)
        :precondition (and
                (trigger ?i)
                (activePhase ?p)
                (contains ?i ?p))
        :effect (and
                (not (trigger ?i))
                (not (activePhase ?p))
                (activeIntergreenAfter ?p)
                (assign (phaseTime ?i) 0) ))

(:process keepInterGreen
        :parameters (?p - phase ?i - intersection)
        :precondition (and
                (activeIntergreenAfter ?p)
                (contains ?i ?p)
                (< (interGreenTime ?i) (interGreenLimit ?p)) )
        :effect (and
                (increase (interGreenTime ?i) (* #t 1 )) ))
```

A road link connected with the outside area can either have incoming or outgoing flows of vehicles. In the first case, vehicles from the outside region are entering the modelled area through the link, otherwise the road link is used by vehicles that are leaving the modelled area. Each road link connected with the outside has a corresponding *entering* (*leaving*) rate, that indicates the maximum flows of vehicles, in either direction, that can be served by the link. Vehicles that are going to enter the network are queued in the corresponding incoming road link, unless the road link is full. Flows of vehicles entering the network can be activated, or deactivated, using Timed Initial Literals (Fox & Long 2003).

3.3 Trials of the Approach

The data enrichment and strategy generation have been tested with real data and traffic scenarios utilising UTMC simulation software en route to progressing to physical trials. Hence, rather than taking in real-time current data, we adjusted the system so that what would be translated into the current state would be from input from the real *historical* data that was available. This would allow checking the performance of the system against the observed performance from historical data, in order to assess its feasibility, before deployment on the road system. The historic data includes sufficient information that can be processed into the input PDDL+ state, the same as a real-time version.

[Figure 3 about here]

As a basis for exploring exceptional or emergency traffic conditions, we chose to use historically averaged traffic data from a time/day when the road links were most congested: morning rush hour, between 8am and 9am on a non-holiday weekday. The modelled region chosen by the transport operator (Transport for Greater Manchester – TfGM) was in the Salford district of Manchester, UK, as shown in Figure 3, and abstracted in Figure 4. Intersections are identified following the IDs provided by TfGM. Directed links are identified by the concatenation of the names of their start and end intersections.

The agent was equipped with existing software called UPMurphi (Della Penna et al. 2009) which can generate strategies using the situational information, the agent's goals, and the agent's knowledge encased in PDDL+ (Fox & Long 2006). This existing software was itself encapsulated by a domain model and initial state processor which reduces the number of redundant states considered by the planner. We tested the agent on a range of classes of problems (i) to clear a saturated road link as soon as possible; (ii) to clear several saturated road links as soon as possible; (iii) to clear a region as soon as possible; and (iv) to clear a saturated road link with nearby road works.

All the goals in the tests below have the format:

$$X1 < N1 \ \& \ X2 < N2 \ \ldots$$

where Xi is the road link occupancy, and Ni is the desired occupancy level. Hence, in this context, clearing road links equates to lowering the occupancy to less than a certain –predefined– value, equivalent to a percentage of its saturation level.

UPMurphi was configured so that its heuristic was to minimise the values of link occupancy in the goal expression (X1, X2, ...). The tests completed generated strategies in less than 30 seconds, on a standard Linux PC with 2GB of memory. The strategies (plans) output from the agent were composed of sequences of the instantiation of the parametrised action in the PDDL+ model: to move on a traffic a signal phase on to the next phase (respecting intergreen intervals, of course). To investigate the scaling up of the method, we enlarged the scenario in Figure 4 in phases, eventually reaching a scenario containing 57 intersections and 125 links.

Validation of the generated strategies was carried out in several steps:

1. Comparison with what would be expected in a “common sense” solution.
2. Comparison of the effects of the generated strategies with a fixed strategy which had been optimised for the time of day by Transport Engineers, using simulation software (SUMO and AIMSUN). Clearly this fixed strategy was not generated to deal with the exceptional event, but nevertheless this was assumed a good comparison as that strategy would be operational when an event occurred.

[Figure 4 about here]

3.4 Results

The first experiment was in part intended to investigate the connection between the agent’s internal traffic model (based on flow values), the microsimulation model SUMO being utilised by Infohub, and the AIMSUN microsimulation package used by TfGM. We were aware that if the PDDL+ model was correct/sufficiently accurate, then the generated strategy was guaranteed to solve the goal when executed; and if the independent simulation tests showed that it does not, then we would conclude that the agent’s PDDL+ model was not correct or sufficiently accurate.

The first test was inspired by a possible scenario. Assume there was an extreme vehicle build upon a link (in our case 3966_1202) entering into the region, and the consequent air quality implications around the link were unacceptable. The problem to address would then be to clear the link as soon as possible. It is formalised by assuming the link contains at the initial state an unexpectedly large number of vehicles (in this case, 300), and the goal state is to reduce the number to less than 10. This scenario is also similar to the effect of car parks emptying into a link after the end of some large event. In the test scenarios, this was the only time that we introduced our own data into the problem formulation, in order to simulate the occurrence of some exceptional event.

The common sense, approximate strategy to solve this kind of problem would be as follows. At the intersection that the link leads out of (in this case 1202) called the “primary intersection”: give maximum green time to those light phases which allow vehicles to leave the link, and minimise those phases which do not, so that the lights will quickly cycle back to the phases letting out traffic. At the intersections that lead off from the primary intersection (in this case 6013 and 1349): give at least enough green time to the links leading in from the primary intersection to make sure that the links do not get congested and the increased level of traffic can go through them smoothly. This strategy may have to be repeated through intersections further away if necessary. To visually inspect the quality of the strategy, we checked that it was indeed close to this common sense solution. An example of the output provided by the planning agent is provided below. For each line of the plan, the first value indicates when the action has to be performed. Since the approach focuses on controlling traffic signals, the only available action for the agent is to switch red the current traffic light phase, in order to move to the next one. The action considers the current phase (first value) and the affected intersection (second value). For instance, the first line of the strategy shown below means that the currently active phase 2 of intersection 1353 has to be stopped after 130 seconds.

130.00: (switchphase J1353_p2 J1353) [0.000]
130.00: (switchphase J1352_p0 J1352) [0.000]
130.00: (switchphase J6013_p1 J6013) [0.000]
130.00: (switchphase J6014_p2 J6014) [0.000]
130.00: (switchphase J1349_p1 J1349) [0.000]
135.00: (switchphase J1867_p2 J1867) [0.000]
140.00: (switchphase J1353_p0 J1353) [0.000]
140.00: (switchphase J1352_p1 J1352) [0.000]
140.00: (switchphase J6013_p2 J6013) [0.000]
140.00: (switchphase J6014_p3 J6014) [0.000]
[...]

Considering the simulation, the traffic models (AIMSUN and SUMO) were run independently by the transport authority and the SME Infohub, respectively, using the planner-output strategy and the fixed optimised strategy. In the first test, after validating that the simulations were fairly consistent, the reduction in time to clear a intersections using the planner-output strategy was approximately 20% using the simulations. AIMSUN and SUMO gave similar results to each other, but tended to produce slightly longer times to clear congestion than the planner's own simulation, and tended to give better results for the planner-generated strategy than the planner's own simulation. Videos of the AIMSUN planner-generated[7] and fixed optimised[8] strategies are available online. This comparison shows a slightly longer makespan than the planner's internal simulator on both configurations (compare with results in Table 1, first row).

[Table 1 about here]

The results of the full range of tests are shown in Table 1: "3 Links" is to clear congestion from 3 road links leading into the intersections 1867, 1349 and 1202 shown in Figure 4, where an extra 600 vehicles are entering as a result of a disturbance in another region; "Saturated" is where all the links in the region of Figure 3 are at capacity, "Roadworks" is the same configuration as the initial test, but with roadworks severely limiting flow between intersections 1202 an 1349. In each case the figures in Table 1 are the times in seconds to decongest the roads involved using the optimised fixed strategy (first column) or the planner-produced strategy (second row) using the planner's simulator. All show a marked reduction in the case of the planner-generated strategies. A common sense, approximate strategy to solve the more complex problems (columns 2-4 in Table 1) is much more difficult to formulate than for the initial test (and hence one of the reasons for automation). However, a sensible pattern appeared to exist in the planner-generated strategies, to green light the correct intersections.
The results of the tests for investigating one aspect of scale-up (how big a region can the planning agent reason with) are shown in the right hand side of the table. The original problem was enlarged from 15 intersections (of which 7 are controllable) to 19 intersections (of which 10 are controllable). The latter scenario was then duplicated to provide the data for 38 and 57 intersection scenarios. The Generation Time is the time in seconds the planning agent took to generate a solution strategy in each case. Not surprising this value rises steeply, but in still manageable for 57 intersections. This compares very favourably to the size of area used in tests of the *decentralised* scheduling-basic traffic management system SURTRAC (Xie et al. 2012).

3.5 Discussion of Results and Future Directions

The tests confirmed that the use of a planning agent to generate strategies in real time in response to some exceptional or unusual congestion-causing event was feasible in terms of generation time. They also confirmed in a variety of scenarios that the quality of the strategies produced were superior in terms of makespan to those fixed time strategies already in operation.
In essence, connecting smart city data infrastructure in terms of semantically enriched and globally integrated data enabled the use of advanced AI technology to provide these solutions to a real UTMC challenge as demonstrated in our trials. Without the semantic level data, any deployment of such technology would require its own layer of software to extract data and would lack the urban-wide integration and reach delivered in the SimplyfAI project.
The main advantage of the approach appears to be its ability to generate a useful, readable strategy in real time to meet the needs of a new unexpected situation. This relies on the flexibility of the PDDL+ encoding, as well as the speed of an agent in dealing with the specified goals. Also, new effectors such as the exploitation of variable speed limits or variable message signs (affecting traffic flows) can be added to the agent's domain model modularly, meaning that new strategies generated will contain instances of those effectors if they help achieving a goal.

7 https://goo.gl/st149L
8 https://goo.gl/dNzByU

To deal with the *uncertainty* in real world scenarios, in future trials involving real physical infrastructure the method of implementation would incorporate monitoring and re-planning as follows. As the infrastructure implements the planning agent-generated strategy, the dynamic data from the current traffic situation will be retrieved via the queries discussed in Section 2, every period of time (typically 5 minutes). The planning agent's simulated results will be compared with this real data, and in cases where they diverge significantly, the planning agent will be forced to create a new plan using the original goal but starting from the advanced state that has just been sensed. As the CPU-time required by the agent during the trials to generate strategies is reasonable (under 30 seconds, even when dealing with 57 intersections) this makes re-planning in real time feasible.
Note also that, though the processes including the plan generating times are low (in terms of seconds), they are non zero. Hence once a strategy is generated for a particular goal and state, and the strategy has been passed to the hardware (signals) to be executed, the state would have changed. In order to take into account the latency in the system (let us assume a delay value in the system of 2 minutes, which takes into account the agent's strategy generation time plus any time for validation by simulation and transfer to hardware) the traffic situation can be very quickly simulated to give the situation after 2 minutes. The planning agent can then be invoked to solve the problem using this future point as its starting point, and the generated solution strategy will then be issued to start at this point in the future.

3.6 Comparison with a SCOOT-driven Control

SCOOT is a demand driven algorithm coping very well with cycle-to-cycle changes in demand. In the scenario where an input link to a intersection suddenly became saturated, its incremental changes would not move timings far from those predetermined –i.e. the presence of a SCOOT system would produce similar results to the optimised strategies identified by the traffic control centre. Furthermore, if the controlled region consists of several SCOOT regions, each SCOOT region is optimised on its own. In contrast, the agent-generated strategies work on the whole region giving an unlimited infrastructure horizon for optimisation.
A fundamental limitation of SCOOT is that it is dependent on its own local data sensors –the inductive loops embedded in the road surface. SimplyfAI's planning agent inputs enriched sources of data including data from inductive loops and more, hence this gives it the immediate advantage of higher quality data and a wider data view. Furthermore, SCOOT's gradual adaptation approach to adjusting the traffic signal timings does not compare well to the immediate adjustment made by the enactment of the agent-generated strategies.
In summary, simulating with SCOOT-based intersection control would force little change to the optimised control strategies to cope with the resulting step change of in-flows to intersections resulting from an exceptional event.

4 Related Work

While there are many examples of the application of general AI techniques to road traffic monitoring and management (Various 2007, Miles & Walker 2006), the generation of a complete temporal strategy to help in the management of road traffic is fairly novel (McCluskey & Vallati 2014, Cenamor et al. 2014). The most mature work based on AI Planning or Scheduling appears to be SURTRAC, a distributed scheduling system which controls traffic signals in urban areas (Xie et al. 2012). In SURTRAC, each intersection is controlled by a scheduling agent that communicates with connected neighbours to predict future traffic demand, and to minimise predicted vehicles waiting time at the traffic signal. It is currently being trialled in Pittsburgh, USA, with its distributed
approach suggesting good scale-up but less flexibility than a centralised AI planning agent.
A line of research in transportation is also devoted to optimise strategies for controlling specific set of intersections, that show some peculiar characteristics (Yang & Chang 2016,). Approaches to region-wide traffic control has been trialled using model predictive control (MPC) strategies and optimisation (Lin 2011, Dotoli et al. 2006, van den Berg et al. 2004), and are able to take into account emissions (Han et al. 2016,). This line of research uses a control theory approach which, given an

adequate dynamical model, can be used to derive a solution that can give continuous responses to changing inputs. Under changing state conditions, researchers have designed MPC algorithms which can continuously adjust the controlled
features (here signal timings) to optimise some given goal in real time. This approach tends to be less flexible than our approach using UPMurphi, as a solution needs to be designed, implemented and tuned using a specific model of traffic flow and a specific objective function. Additionally it is less scrutable, as it generates plans over a restricted time horizon.
A different line of approaches to cope with traffic congestion is based on the idea of controlling the access of vehicles to the network. The work of Csikós et al. (2017), for instance, shows how the flow of vehicles can be optimised while constraints on the lenght of the queue waiting to gain access to the area are respected.
A number of approaches have been recently introduced specifically to handle exceptional events in urban area networks. Darmoul et al. (2017) suggested a multi-agent architecture to control interrupted flow at signalised intersections. Each agent a junction, and adapts to disturbances in traffic flows. Similarly, Aslani et al. (2017) exploit actor-critic method to perform adaptive signal traffic control, and show how their approach is able to cope with different traffic distruption events. The approach proposed by Chai et al. (2017) deals with exceptional and unexpected events by dynamically re-routing traffic; similar approaches are also developed for usual traffic conditions (Barthélemy & Carletti 2017). As it is apparent, the two lines of research (controlling traffic lights and routing traffic) are complementary, and it is envisaged that they will be combined in the future. Finally, it is worth noting that also social media strategies can be exploited to affect the behaviour of traffic and transport users, as well as transport policies, particularly in the case of large events (Cottrill et al. 2017, Gal-Tzur et al. 2014).
The approach exploited in the SimplyfAI project is based on recent work from Vallati et al. (2016) (and subsequently extended in McCluskey & Vallati (2017)). It was inspired by works such as those from Lin (2011), and van den Berg et al. (2004), where traffic is modelled using “flows”, and then analysed through model-predictive controllers. The work of Vallati et al. (2016) exploit PDDL+ (Fox & Long 2006) for encoding a flow model of vehicles through traffic-light controlled intersections, though the scope of the work, which was carried out in an academic environment, was limited. Their experimental analysis, however, demonstrated that UPMurphi could solve traffic problems containing thousands of vehicles, in response to exceptional conditions (Vallati et al. 2016). They showed the efficacy of the resulting strategy by comparing its execution with a “fixed time” plan, and plans generated by a reactive approach, using SUMO. The largest scenario used was a hand simulation of a real problem, and the strategy generated by UPMurphi was shown by SUMO to be approximately twice as efficient as a fixed time and reactive-generated strategies.
Semantic Web technologies have been utilized in various road traffic applications. More specifically, Valle et al. (2011) presented a prototype for road traffic prediction and trip planning in the city of Milano (Italy). The prototype imports information from heterogeneous sources by converting it into RDF format and utilises conceptual (SPARQL) query answering for information retrieval. Lécué et al. (2012, 2014) developed a system that predicts the severity of road traffic congestion in the city of Dublin (Ireland). A major part of the system was the semantic integration of datastreams coming from different sources including road weather condition, weather information, Dublin bus stream, social media feeds, road works and maintenance, and city events. Used data were initially available in multiple formats such as CSV, XML, Tweets, PDF and ESRI SHAPE, but subsequently converted into OWL 2 EL and fed to the system. The system was further extended and named STARCITY (Lécué et al. 2014), while being deployed in cities including Dublin (Ireland), Bologna (Italy), Miami (USA) and Rio (Brazil).

5 Conclusions

In this paper we have described the operation and results of a collaboration between a transport authority, academics, a large technology provider and two SMEs to not only implement data infrastructure that underpins a Smart City, but to show how this can enable the use of an intelligent

agent to create traffic signal strategies in real time to solve challenges caused by exceptional or unexpected conditions. The data integration in the urban environment was enacted via a data hub, in order to integrate disparate types of data obtained from sensors and historical databases. Using data items that are based on ontologies, we elevated the data to become logical facts that can be consumed by a planning agent.

We embedded within the agent the UPMurphi software which is capable of generating plans of traffic signal changes to achieve desired goals in the presence of exceptional events. The trials involved using data describing the traffic and infrastructure in the region of a large city. The strategies (timing changes of traffic signals) output were judged to be useful for dealing with exceptional situations, using both hand inspecting the strategies to check that they were sensible and simulating their execution using two different traffic modelling software packages AIMSUN and SUMO. We believe that this is the first successful demonstration of using a planning agent to create useful strategies for UTC where the overall control for the region chosen, the nature of the data feeds, the planning of the project, and the validation of the end result was largely in the hands of non-academic stakeholders. Further, we believe that this is the first demonstration of its kind in the integration of the semantic layer with an AI agent. For future work, we plan to further develop the data hub platform, and field test the agent within physical trials.

6 Acknowledgements

This work was supported by the European COST ARTS Action 1102, and UK NERC grant NE/N007239/1 issued through Innovate UK. Many previous works made this trial possible. Firstly, we are indebted to the members of the SimplyfAI consortium, and in particular Sam Corns and Louis Burrows who conducted the comparative simulations with AIMSUN and SUMO respectively. We would like to thank Bart De Schutter for hosting Mauro Vallati's Short Term Scientific Mission in 2015, and Daniele Magazzeni for introducing us to the implementation and configuration of UPMurphi. Finally, we acknowledge the help of members of the COST ARTS Action too numerous to name who explained to us the finer points of Transport Engineering.

References

Antunes, F., Freire, M. & Costa, J. P. (2016), 'Semantic web and decision support systems', Journal of Decision Systems 25(1), 79–93.

Aslani, M., Saadi Mesgari, M. & Wiering, M. (2017), Adaptive traffic signal control with actor-critic methods in a real-world traffic network with different traffic disruption events, Transportation Research Part C: Emerging Technologies, (85) 732-752.

Barthélemy, J. & Carletti, T. (2017), A dynamic behavioural traffic assignment model with strategic agents, Transportation Research Part C: Emerging Technologies, (85) 23-46.

Beart, P. (2016), 'Hypercat 3.00 specification'.

Berners-Lee, T., Hendler, J. & Lassila, O. (2001), 'The semantic web', Scientific American pp. 29–37.

Bizer, C., Heath, T. & Berners-Lee, T. (2009), 'Linked data - the story so far', Int. J. Semantic Web Inf. Syst. 5(3), 1–22.

Blomqvist, E. (2014), 'The use of semantic web technologies for decision support–a survey', Semantic Web 5(3), 177–201.

Cenamor, I., Chrpa, L., Jimoh, F., McCluskey, T. L. & Vallati, M. (2014), Planning & scheduling applications in urban traffic management, in 'Proceedings of PLANSIG'.

Chai, H., Zhang, H.M., Ghosal, D. & Chuah, C. (2017), Dynamic traffic routing in a network with adaptive signal control, Transportation Research Part C: Emerging Technologies, (85) 64-85.

Chong-White, C., Millar, G. & Shaw, S. (2012), SCATS and the environment study: definitive results, in 'Proceedings of the 19th World Congress on Intelligent Transportation Systems (ITS)'.

Csikós, A., Charalambous, T., Farhadi, H., Kulcsár, B. & Wymeersch, H. (2017), Network traffic flow optimization under performance constraints, Transportation Research Part C: Emerging Technologies, (83) 120-133.

Cottrill, C., Gault, P., Yeboah, G., Nelson, J.D., Anable, J. & Budd, T. (2017), Tweeting Transit: An examination of social media strategies for transport information management during a large event, Transportation Research Part C: Emerging Technologies, (77) 421-432.

d'Aquin, M., Davies, J. & Motta, E. (2015), 'Smart cities' data: Challenges and opportunities for semantic technologies', IEEE Internet Computing 19(6), 66–70.

Darmoul, S., Elkosantini, S., Louati, A. & Ben Said, L. (2017), Multi-agent immune networks to control interrupted flow at signalized intersections, Transportation Research Part C: Emerging Technologies, (82) 290-313.

Della Penna, G., Magazzeni, D., Mercorio, F. & Intrigila, B. (2009), UPMurphi: A tool for universal planning on PDDL+ problems, in 'Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS)'.

Dotoli, M., Fanti, M. P. & Meloni, C. (2006), 'A signal timing plan formulation for urban traffic control', Control Engineering Practice 14(11), 1297–1311.

Fox, M. & Long, D. (2003), 'Pddl2. 1: An extension to pddl for expressing temporal planning domains.', Journal of Artificial Intelligence Research 20, 61–124.

Fox, M. & Long, D. (2006), 'Modelling mixed discrete-continuous domains for planning', Journal of Artificial Intelligence Research 27, 235–297.

Gal-Tzur, A., Grant-Muller, S.M., Minkov, E. & Nocera, S. (2014), The Impact of Social Media Usage on Transport Policy: Issues, Challenges and Recommendations, Procedia - Social and Behavioral Sciences, (111) 937-946.

Ghallab, M., Nau, D. & Traverso, P. (2004), Automated planning: theory & practice, Elsevier.

Han, K., Liu, H., Gayah, V.V., Friesz, T.L. & Yao, T. (2016), A robust optimization approach for dynamic traffic signal control with emission considerations, Transportation Research Part C: Emerging Technologies, (70) 3-26.

Hayes, P. (2004), Rdf semantics, in 'W3C Recommendation'.

Hitzler, P., Krötzsch, M., Parsia, B., Patel-Schneider, P. F. & Rudolph, S., eds (27 October 2009), OWL 2 Web Ontology Language: Primer, W3C Recommendation. Available from http: //www.w3.org/TR/owl2-primer/.

Krajzewicz, D., Erdmann, J., Behrisch, M. & Bieker, L. (2012), 'Recent development and applications of SUMO - Simulation of Urban MObility', International Journal On Advances in Systems and Measurements 5(3&4), 128–138.

Lécué, F., Schumann, A. & Sbodio, M. L. (2012), Applying semantic web technologies for diagnosing road traffic congestions, in 'The Semantic Web - ISWC 2012 - 11th International Semantic Web Conference, Proceedings, Part II', pp. 114–130.

Lécué, F., Tucker, R., Bicer, V., Tommasi, P., Tallevi-Diotallevi, S. & Sbodio, M. L. (2014), Predicting severity of road traffic congestion using semantic web technologies, in 'The Semantic Web: Trends and Challenges - 11th International Conference, ESWC 2014, Proceedings', pp. 611–627.

Lécué, F., Tucker, R., Tallevi-Diotallevi, S., Nair, R., Gkoufas, Y., Liguori, G., Borioni, M., Rademaker, A. & Barbosa, L. (2014), Semantic traffic diagnosis with STAR-CITY: architecture and

lessons learned from deployment in dublin, bologna, miami and rio, in ‘The Semantic Web - ISWC 2014 - 13th International Semantic Web Conference, Proceedings, Part II’, pp. 292–307.

Lin, S. (2011), Efficient model predictive control for large-scale urban traffic networks, TU Delft, Delft University of Technology.

McCluskey, T., Kotsialos, A., Müller, J., Klugl, F. & Schumann, R. (2016), Autonomic Road Transport Support Systems, Autonomic Systems, Springer, London, UK.

McCluskey, T. L. & Vallati, M. (2014), Extracting information from urban traffic data to improve road traffic management, in ‘Final Report for Transport Catapult Innovation Voucher’.

McCluskey, T. L. & Vallati, M. (2017), Embedding automated planning within urban traffic management operations, in ‘Proceedings of the Twenty-Seventh International Conference on Automated Planning and Scheduling, ICAPS 2017.’, pp. 391–399.

Miles, J. C. & Walker, A. J. (2006), ‘The potential application of artificial intelligence in transport’, Journal of Intelligent Transport Systems 153.

Taale, H., Fransen, W. & Dibbits, J. (1998), The second assessment of the SCOOT system in Nijmegen, in ‘IEEE Road Transport Information and Control’, number 21-23.

Tachmazidis, I., Batsakis, S., Davies, J., Duke, A., Vallati, M., Antoniou, G. & Clarke, S. S. (2017), A hypercat-enabled semantic internet of things data hub, in ‘The Semantic Web - 14th International Conference, ESWC 2017, Proceedings, Part II’, pp. 125–137.

Tachmazidis, I., Davies, J., Batsakis, S., Antoniou, G., Duke, A. & Clarke, S. S. (2016), Hypercat RDF: Semantic Enrichment for IoT, in ‘Semantic Technology - 6th Joint International Conference, JIST 2016, Singapore, Singapore, November 2-4, 2016, Revised Selected Papers’, pp. 273–286.

Townsend, A. (2013), Smart Cities: Big Data, Civic Hackers, and the Quest for a New Utopia, WW Norton & Company.

Vallati, M., Magazzeni, D., De Schutter, B., Chrpa, L. & McCluskey, T. L. (2016), Efficient macroscopic urban traffic models for reducing congestion: a pddl+ planning approach, in ‘The Thirtieth AAAI Conference on Artificial Intelligence (AAAI)’.

Valle, E. D., Celino, I., Dell’Aglio, D., Grothmann, R., Steinke, F. & Tresp, V. (2011), ‘Semantic traffic-aware routing using the larkc platform’, IEEE Internet Computing 15(6), 15–23.

van den Berg, M., De Schutter, B., Hegyi, A. & Hellendoorn, J. (2004), Model predictive control for mixed urban and freeway networks, in ‘Proceedings of the 83rd Annual Meeting of the Transportation Research Board’, Vol. 19.

Various (2007), ‘Artificial intelligence in transportation’, Transportation Research Circular E-C113, Transport Research Board.

Zhang, L., Song, Z., Tang, X. & Wang, D. (2016), Signal coordination models for long arterials and grid networks, Transportation Research Part C: Emerging Technologies, (71) 215-230.

Xie, X.-F., Smith, S. & Barlow, G. (2012), Schedule-driven coordination for real-time traffic network control, in ‘Proceedings of the 22nd International Conference on Automated Planning and Scheduling (ICAPS)’.

Yang, X. & Cheng, Y. (2017), Development of signal optimization models for asymmetric two-leg continuous flow intersections, Transportation Research Part C: Emerging Technologies, (74) 306-326.