Multi-Paradigm Reasoning for Access to Heterogeneous GIS

François Bry
Institute for Informatics
Ludwig-Maximilians University
Oettingenstr.67
D-80538 Munich
bry@ifi.lmu.de

François-Marie Colonna
LSIS - CNRS
Avenue Escadrille
Normandie Niemen
13013 Marseille, France
colonnaf@lsis.org

Bernhard Lorenz
Institute for Informatics
Ludwig-Maximilians University
Oettingenstr.67
D-80538 Munich
lorenz@pms.ifi.lmu.de

ABSTRACT

Accessing and querying geographical data in a uniform way has become easier in recent years. Emerging standards like WFS turn the web into a geospatial web services enabled place. Mediation architectures like VirGIS overcome syntactical and semantical heterogeneity between several distributed sources. On mobile devices, however, this kind of solution is not suitable, due to limitations, mostly regarding bandwidth, computation power, and available storage space. The aim of this paper is to present a solution for providing powerful reasoning mechanisms accessible from mobile applications and involving data from several heterogeneous sources. By adapting contents to time and location, mobile web information systems can not only increase the value and suitability of the service itself, but can substantially reduce the amount of data delivered to users. Because many problems pertain to infrastructures and transportation in general and to way finding in particular, one cornerstone of the architecture is higher level reasoning on graph networks with the Multi-Paradigm Location Language MPLL. A mediation architecture is used as a "graph provider" in order to transfer the load of computation to the best suited component – graph construction and transformation for example being heavy on resources. Reasoning in general can be conducted either near the "source" or near the end user, depending on the specific use case. The concepts underlying the proposal described in this paper are illustrated by a typical and concrete scenario for web applications.

Categories and Subject Descriptors

G.2.2 [**Discrete Mathematics**]: Graph Theory—*Network problems, Path problems*; H.2.4 [**Database Management**]: Systems—*Distributed databases, Query processing*; I.2.3 [**Artificial Intelligence**]: Deduction and Theorem Proving—*Deduction, Resolution*

General Terms

Graph networks, Database integration

Keywords

graphs, deductive languages, database, integration

Acknowledgements

This research has been funded in part by the European Commission and by the Swiss Federal Office for Education and Science within the 6th Framework Programme project REWERSE number 506779 (cf. `http://rewerse.net`) and by the Centre de Coopération Universitaire Franco-Bavarois (CCUFB) (cf. `http://www.bfhz.uni-muenchen.de`) within the project *Multi-Paradigm Location Reasoning* (MPLR). We also thank Hans Jürgen Ohlbach, University of Munich, for his contributions to section 3.1.

1. INTRODUCTION

Nowadays querying a single geographic data source from a mobile device is easily possible and most GIS' proprietary solutions facilitate these mobile functionalities [12, 13]. With widespread use of commercial or open sources GIS, the amount of available spatial data has grown substantially. Although many of the sources for geospatial data are accessible via the web, due to the multitude of proprietary standards, formats and protocols, standardised and uniform access is the exception. So, interoperability has been and still is a great GIS challenge, not only on traditional workstations, but especially on mobile devices such as cellular phones or PDAs. There is a demand for new technologies allowing the exploitation of heterogeneous and distributed sources in general and in particular on mobile devices.

1.1 Common integration approaches

Warehousing and mediation are two common approaches used for database integration. The first approach is based on data rewriting, while the second concentrates on query rewriting. Both come with their own inherent advantages and drawbacks. The warehousing architecture is not equally suitable on all computing platforms, especially on mobile ones, as it requires a large amount of local data storage. Furthermore, it also requires real time updates of data which is costly due to bandwidth limitations. The combination of low bandwidth and insufficient network coverage and reliability hampers the transfer of large amounts of data. Mediation engines are based on query rewriting. The mapping of the global schema to the local sources is an important step; the specification of these correspondences will determine the difficulty of query rewriting and adding/removing sources to/from the system. Using **G**lobal **A**s **V**iew [26, 28], relationships on global schema are considered as "views" on the local ones. Adding or removing a source forces to completely revise the global schema, but query rewriting leads to simple rule unfolding (as classical views' execution in databases). With **L**ocal **A**s **V**iew [15, 19, 20], all relationships on local sources are defined as views on the global schema. The complexity of query rewriting is increasing, but adding and removing sources is easier, as the global schema is set only once.

Most of the time, the user only wants to state a simple question, and then refine the result, by adding some more filters. For this purpose, mediation is the best choice, as map consultation is a *read only* process for most users. The various integration techniques used in [6] helped in building such a geospatial web service; our solution for building a web based mediator using WFS [23] partially solved the syntactical and semantical problem of integration. This article describes specific problems which arise from the mobile context. On mobile devices, an efficient solution is to use lightweight data structures and few network communications. It is not possible to use GML and it's strong encoding directly, as the equivalent of a small XML-DBMS needs to be embedded. Reasoning at a high level is part of the solution we propose. Advanced reasoning capabilities on spatial concepts are needed on embedded GIS systems, and we use the Multi-Paradigm Location Language MPLL [2, 7] as the core of the reasoning system, in order to produce valid destinations and optimised routes.

1.2 The Semantic Web

The Semantic Web is an endeavour aiming at enhancing Web data with meta-data and data processing, as well as processing methods specifying the "meaning" of such data and allowing Web-based systems to take advantage of "intelligent" capabilities. In a Scientific American article [3] which has diffused the Semantic Web vision, this endeavour is described as follows:

> "The Semantic Web will bring structure to the meaningful content of Web pages, creating an environment where software agents roaming from page to page can readily carry out sophisticated tasks for users."

Reasoning is central to the Semantic Web vision since reasoning is central to processing *declarative* data and specifying *intelligent* forms of data processing. In the above-mentioned article, this central role of reasoning for realizing the Semantic Web vision is stressed as follows:

> "For the semantic web to function, computers must have access to [. . .] sets of inference rules that they can use to conduct automated reasoning." [3]

Inference rules operate on facts and axioms. Axioms specify in an abstract way a model of the world. For example, the axiom $\forall x\ motorway(x) \Rightarrow road(x)$ says something about the relation between the words 'motorway' and 'road'. The most detailed axiomatisations which are currently being used for the semantic web are ontologies. They are formulated in logical formalisms like Description Logics [1] or OWL [31] and describe more or less complex relationships between different notions (concepts and relations) used in particular domains. Pure logical formalisms have a somewhat one-track style of expressiveness, so logical axiomatisations often give only a very coarse picture of the world. A web service, for example, which computes the shortest way to get from Munich to Hamburg needs a much more detailed picture of the world, namely digital road maps, than any pure logical axiomatisation is able to provide.

To complement standard, general purpose, logic-based data modelling and reasoning methods, as e.g. offered by RDF and OWL and reasoners for these languages, geospatial reasoning with topographical data is best tackled using graphs for data modelling and well-established graph algorithms for handling inference – the purpose of MPLL.

Completely general reasoning techniques must, by their very nature, be weakly committed to any particular class of problem and are thus unable to take advantage of any particular properties of that class. We therefore claim not only that the class of geospatial reasoning problems requires equally specific reasoning methods but that logic-based, general-purpose methods could never properly, intuitively, and efficiently realize what is best achieved using graphs and graph algorithms.

It has been claimed by Bry and Marchiori [8] that, on the Semantic Web, "theory reasoning" is a desirable complement to "standard reasoning". This articles substantiates this claim with respect to evidence from the practical case of geospatial reasoning for geographical guidance.

The paper is organised as follows: section 2 describes a concrete scenario which illustrates the shortcomings of the VirGIS mediation architecture and the improvements that are needed. Section 3 details the theoretical developments and components of the reasoning platform particularly adapted to mobile devices. Section 4 presents one of the possible solutions found for our application scenario. Finally, we conclude and open new perspectives of work in section 5.

2. APPLICATION SCENARIO

A person is on vacation in some foreign city on a Sunday afternoon. For some reason the person needs a certain medication rather urgently, therefore a handheld device with wireless access to the Web is used to send a query similar to the following: "Where is the nearest pharmacy with medication xyz in stock?". A map illustrating this scenario is shown in Fig. 1.

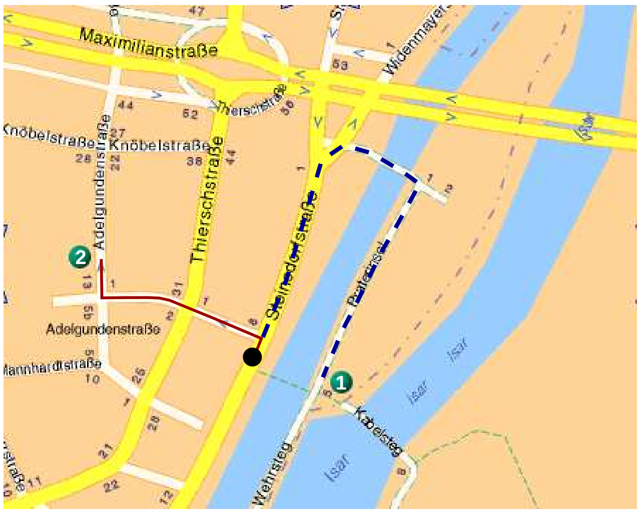

Figure 1: Where is the nearest pharmacy?

In this case, the term *nearest* pharmacy does not pertain to metric distances, but to a routing problem depending partly on user preferences and other constraints, such as *mode of transportation is "on* *foot"*, *special qualities of the pharmacy are "currently open" and "selling medication in question"*, etc. Producing the correct answer with common "human" reasoning capabilities (and the usual aids like a map or directions) is rather straightforward, doing the same using automated reasoning requires higher level reasoning techniques.

The *topology* of the surrounding area and the resulting semantics is a key factor. In this case the *semantic meaning* of the map is that crossing the river is only possible at certain points where there is a bridge (or an underpass). The same applies to a number of other features, such as railway lines or highways [18] – depending on the mode of transportation. Symbolic aspects, such as the borders of countries or districts for example, can have similar influence, although physical counterparts need not exist.

Simple, or naive, solutions already exist for this problem, for example as shown in [22], where the user can specify some preferences for filtering the answer. The results are given ordered according to the linear distance from current user position. The nearby services presented as additional informations on the maps are "pre-computed", as users can only choose within a given set : {`hotels, restaurants, tourist attractions, car parks`}. We can then suppose that there is a specific optimisation while accessing the database, using clusters or indexes. Achieving this would not be efficient if the set of services would be much larger, as numerous optimisations like indexes also add overhead to the database system as a whole, and so should be used sensibly.

2.1 Sketching a concrete Use Case

Input of the query is done (depending on device capabilities) for example via speech recognition or keyboard input, in form of natural language. After a short while the device displays one or more choices of destinations, i.e. pharmacies with suitable opening hours which have the requested item in stock. In addition, the specificities of the respective routes are displayed according to a personalised ranking system. Routes are for example ordered by travel times, cost or other factors. The user either accepts the suggested "best" route or chooses one of the alternatives by hand. If no route is deemed suitable by the user (or the routing system), user preferences or search criteria could be readjusted. While following the route, navigation instructions are given step by step as natural language, optionally aided by a graphical map, until the destination is reached.

2.2 Typical solution with real or virtual GIS

In the following, we call "real GIS" solutions like [11], and "virtual GIS" mediation based systems, where data is not stored permanently, but accessed at query-time [6]. With a single (real or virtual) GIS we can only write a query taking into account the parameters. In Fig. 1, we suppose that the user position is represented by the black circle, and pharmacies by the points with number 1 and 2. Let $U(x_u, y_u)$ user's Cartesian coordinates, $P1(x_{p1}, y_{p1})$, $P2(x_{p2}, y_{p2})$ the coordinates of pharmacies 1 and 2 respectively, and $distance : (pt_1, pt_2) \rightarrow d$ the function computing distance between two points. If the user asks for the nearest pharmacy, a possible execution would be first the calculus of $d1 = distance(U, P1)$ and $d2 = distance(U, P2)$. Then as $d1$ is shorter than $d2$, we obtain by comparison the correct result: *pharmacy 1*. Using the VirGIS platform, this can be expressed by the query shown in Fig. 2, using a dedicated language for GML [5]. The answer would be *pharmacy 1*. Or, a single look at the map shows that nearest pharmacy is not

```
let $x:=current_pos
for $y in document(''pharmacies'')//pharmacy
where distance($x,$y) <= min((list))
return $y/adress;
```

where **`(list)`** is a list computed by expression:

```
for $z in document(''pharmacies'')//pharmacy
return distance($x,$z);
```

Figure 2: Typical geographic query

number one but two, located on the same river side. The correct solution is shown as the solid line, not the dotted one.

2.3 Proposed Solution

Experience [14] shows that developing a tool for mobile GIS access is not a trivial task. The limited capabilities of both the network and the device have to be taken into account, as well as the fact that the input of complex queries is often not possible. Further problems have already been detailed in section 1.

As classic reasoning is not sufficient, we propose an architecture based on reasoning techniques at a higher level. Based mainly on MPLL, wrappers for graphs structures and VirGIS, this system also adds a degree of supplementary refinement to solve geospatial problems using context information. This "context" includes among other things the current position, user's preferences, and device capabilities. Subsequently, not only Euclidean distances (e.g. [21]) can be calculated, but specific limitations can be taken into account: a pedestrian can't walk along the highway, cars cannot pass through pedestrian zones, etc. The MPLL reasoning language is coupled with the VirGIS system in the case where graphs extracted from heterogeneous and distributed data sources are necessary. MPLL is not designed to solve such integration problems, which falls into the domain of VirGIS. Mediation aids in grouping sources, from which the corresponding graph structures are extracted. MPLL sees the VirGIS engine as a provider of geospatial entities, corresponding to the ones described in the GeoOntology. Details on each architecture layer can be found in the following section.

3. GLOBAL ARCHITECTURE

From top to bottom, the model can be divided into three parts, as shown in Fig. 3: the user interface (a complete GUI or a simple input form), the reasoning engine, and local heterogeneous sources seen as a unique database through VirGIS. The *Graph Wrapper* can be considered part of the MPLL functionality, the *Context* information is just another source of (mostly non-geospatial) information pertaining to the reasoning layer.

The access to geospatial data source by MPLL is done either through VirGIS or directly, depending on the data provider, formats and other factors. Essentially, the graph representations needed by MPLL are generated between the data sources and the reasoning layer. That also means that MPLL is not depending for example on VirGIS to provide the correct data format, although the ability of VirGIS to substantially reduce data sent to MPLL by generating the graph representations near the source is a substantial advantage. The output format could be GXL or GraphML if the transformation has already been done, GML or another format otherwise.

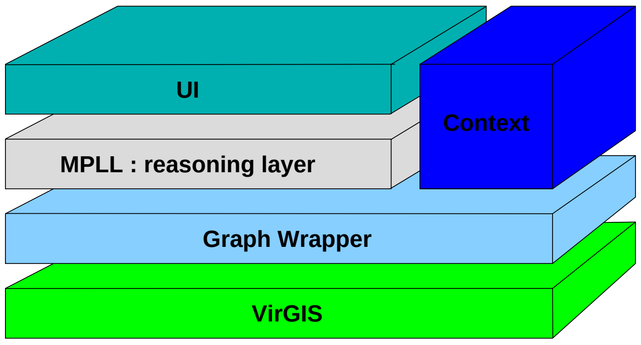

Figure 3: Layers of the system

Reasoning is then done on the MPLL internal data structures which are built by parsing the graph data either generated by MPLL or input directly from VirGIS. The internal representation is of secondary importance for the inter-working components but can be considered close to the graph sources. Theoretically, MPLL can itself act as an intermediate layer, if another instance of MPLL sent a query for some graph structure only – this might be necessary in cases where a number of different networks have to be processed by different instances of MPLL, whereas one instance is responsible for combining results and/or higher level reasoning.

Results are then presented (on an intermediate level, not necessarily meaning textual or graphical presentation at this point) in form of a plan language. The rough idea behind this language is the ability to have a high level description of a plan (i.e. also a route to follow) which can be interpreted by different devices depending on their specific abilities. A mobile phone for example might rely on audio capabilities only (e.g. using Voxx [2] for spoken or written output), while a device with a bigger screen might additionally display graphical elements or even complete maps. Another important advantage is the possibility to use parts of the result, i.e. parts of a plan, to send queries for more detailed information. If a travel plan includes the way to the airport in the user's hometown for example, there is rarely a need for very detailed route planning – the user probably has been to the airport many times already. The same section of a plan covering the local transport from the destination airport to, say, a conference venue, very probably has to be much more detailed, it might be the user's first visit. If the plan is presented as a combination of a number of segments (e.g. home – airport – flight – airport – hotel), then certain segments could be re-queried for more detail, while others are left to be higher level descriptions.

3.1 MPLL

Reasoning on locations normally operates at a numerical level (e.g. coordinates) or at a symbolic level (e.g. graphs). Extensive research has been conducted in either case [10], hence there is a broad choice of proven sets of calculi and algorithms to solve the respective tasks. As stated in section 2, whenever possible, reasoning should be conducted on higher level data, which is usually less heavy on the device's resources. The fundamental insight is that many queries pertaining to location information are closely related to the problem of route planning and way finding, which reduces the problem domain considerably. There are two reasons for this. First, whenever a certain location is sought after, chances are, that the inquirer intends to visit the location. Cases like these result in classic route planning tasks. Second, when people refer to the "distance" between two locations in the sense of locomotion, nearly never are they talking about distances per se (metres, kilometres) but the time needed to overcome these distances ("a ten minute walk" or "half an hour by train"). In fact, in many scenarios the exact distance between two points has a rather subordinate meaning from a traveller's viewpoint, especially in urban environments. MPLL (*Maple*) aims for producing a set of reasoning techniques and for providing a framework for Semantic Web applications. This framework will include suitable data types and ontologies for processing and presenting location data.

From GML to Graphs

Routing and way finding usually involves algorithms which operate on graph structures representing the real world. Data about real world entities come in different flavours. To be more precise, the great variety of (proprietary) standards for GIS data, deriving from the specificity of the respective systems (one GIS is optimised for cadastral applications, another for surveying, a third one for construction, etc.) has been a substantial problem in data integration. The *Geography Markup Language* (GML) [24] provides a unified format to facilitate data interchange between different systems. MPLL provides input mechanisms for GML, while other formats can be supported by just providing a customised import filter.

Internally however, the form of data structures used follows their purpose, or rather the purpose of the reasoning processes operating on the data. In the case of MPLL these processes are mainly graph (generic search, shortest paths, and related) algorithms. The preferred data structure for these tasks is a graph structure representing the real world network structures. A simple example is a network of streets which are connected to each other at junctions. A user can move along this network by travelling via streets and changing directions at junctions – in the graph analogue this would be traversing along edges and changing directions at nodes.

To generate graph structures from GML (or other formats) we therefore need mechanisms to achieve this in an automated fashion. Several issues arise in this process:

Unified Representation of Graphs. The structures at the different levels of the hierarchy are all graphs. Therefore there should be a unified representation of these graphs. The graphs need, however, be represented in different forms.

- We need a persistent representation of graphs which can be stored in files or databases.
- We need an in-memory representation of the graphs with a well defined application programming interface, probably similar to the DOM structures of XML data.
- We also need geometric representations of the graphs which can be used to display the graphs on the screen. As long as the nodes of the graph have coordinates, this is not a big problem. Graphs at the symbolic level of the hierarchy usually don't have coordinates. Fortunately there are well developed graph layout algorithms which we can use here.

Since graphs at different levels of the hierarchy can represent the same objects, road crossings, for example, it is very important to maintain the links between the same objects in the different graphs.

These links enable algorithms to choose the level of detail they need for doing their computations.

It must also be possible to use the transition links between different graphs of the same level to join several graphs into one graph. For example, a route planner for somebody without a car may need a combined graph of all public transport systems.

As mentioned above, it should be possible to add extra information to the graphs, which is not derivable from graphs at the lower levels. In order to do this, we need to develop an *editor* for the graphs. ■

'Geospatial' Ontology. We need to develop an ontology of interesting structures which can occur within graphs (road crossings, roundabouts, floors, train stations etc.). Such an ontology would be the anchor point for various auxiliary structures and algorithms, in particular:

- patterns which allow one to identify the structure in a graph, a roundabout, for example;
- transformation algorithms which simplify the structures to generate the nodes and edges in the graphs at the higher levels of the hierarchy;
- transformation algorithms which generate a graphical or verbal representation of the structures on the screen.

The ontology will also be used to annotate the structures in the graphs. ■

Ontology of Graph Types. The graphs at the different levels of the hierarchy provide the data for solving different kinds of problem. We need to classify the graph types, such that it is possible to choose the right graph for a given problem. Examples for this (e.g. "driver level" vs. "planning level") can be found in [30].

Geospatial data is available in many different formats which contain information of different levels of detail, coarseness and general structure. Just as there is no single ideal format or structure, there also doesn't exist an ideal level of detail. Each type of application might require a certain level of complexity and/or resolution of detail, the two of which in general are complementary. Complex objects or features are used in order to hide their inherent complexity and to simplify their handling. These complex objects are composed of simple (or simpler, if there exist multiple different levels) features, which in turn contain information of higher detail. On a larger scale therefore, complex objects facilitate efficient handling and serve on an overview level. Whenever the application operates on a comparably large scale, processing is not hampered by a huge amount of highly detailed, smaller scale information - which is at this point most likely not even of any use. Rendering a cadastral map of a town which shows streets, property boundaries, and positions of buildings is normally not reliant on sub-centimetre resolution. Moving to a smaller scale environment, however, reveals the importance of higher detail. Whatever superfluous information could be disregarded before might be essential for smaller scale processing. Same cadastral data, when processed for example for taxation purposes, better be highly accurate.

The following sections discuss representations of the same object. Adhering to the underlying problem domain we take a closer look at an intersection of streets which is modelled at three different levels of detail. Subsequently we dicuss possible transitions between these levels.

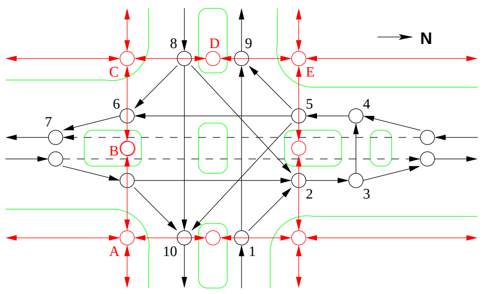

Figure 4: High detail

Fig. 4 shows a richly detailed representation of an intersection of two streets, including an underpass (dashed lines), street segments (numbered nodes), and pedestrian pathways (alphabetised nodes). Even some road furniture – in this case signs which constrain the direction of traffic flow – is implicitly given by the directed edges between the nodes. For example getting from node 1 to node 7 is achieved by traversing the nodes 2,3,4,5,6 in between, whereas from nodes 9 or 10 there exists no feasible traversal to node 7 within the limits of this particular section of the graph. Similar rules can be deduced for the pedestrian network.

An intermediate, simplified version of this crossing is shown in Fig. 5. It contains enough information at the planning level for standard navigation purposes.

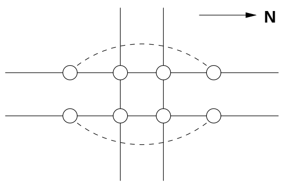

Figure 5: Medium detail

Finally, one can collapse the whole road crossing into a single node of the road network as seen in Fig. 6. This is sufficient for path planning on a larger scale.

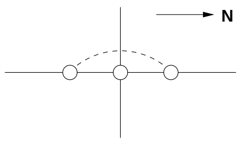

Figure 6: Low detail

The definition of these different levels is depending highly on the specific purpose. A conceptual model proposed for routing purposes of car traffic on a highway network [29] consists of three different levels: *planning*, *instruction* and *driving*. In this model, some operations on one level can be broken down into corresponding operations at a lower level. Thus, the instruction *"take exit 510a"* corresponds to a series of driving operations similar to *"change to the right lane, slow down to 25 MPH, take exit lane at exit 510a"*. Note that lower-level operations do not necessarily need to have a higher-level counterpart, as for example the driving instructions *"slow down"* or *"accelerate"* cannot be found on the planning level. In this specific conceptual model, this constraint, due to the inherent top-down approach, can be properly handled. Other models could favour a bottom-up or other approach. ■

Ontology of Means of Transportation. A graph in general represents only routes, i.e. the roads of a road network or the tracks of a railway network. Because of their importance for the routing process, a lot of additional information is needed, though. Characteristics of a connection, such as speed limits, number of lanes, etc. are equally important as train schedules or travel times. Many other attributes not only pertaining to the network characteristics (e.g. signage, traffic laws and regulations, road capacities, etc.) but also to other elements, such as vehicles and their characteristics have to be taken into account.

Therefore we need to develop an ontology for all kinds of objects which are connected with the graphs. Section 3.2 describes work in progress dealing with the expansion and further development of an existing ontology. ■

Context Modelling. In the introductory examples we showed that queries which require 'locational reasoning' need to take into account the context of the user. We must therefore develop a formal model of the context. The context can, for example, be the current situation of a human user: whether he has a car or not, whether he has luggage or not, his age and sex, and many other factors. ■

Customised Graph Construction. As we have seen in the introduction, many 'locational reasoning' problems require the solution of shortest path problems in a graph. The concrete graph which is relevant for the given problem, may, however, not be one of the graphs which are permanently available. It may be a combination of subgraphs from different graphs, and the combination may be determined by the context of the problem. Therefore we need to develop mechanisms for determining and constructing for a given problem the right combination of subgraphs as the input to the relevant problem solving algorithm.

Aiding in graph construction and also in solving the problem of level of detail would be mechanisms, which transform underlying data to whatever level of detail is needed for a certain task. Depending on the underlying model, either or both automated generation of *more* detail and *less* detail than the original data provide is desirable. Whenever necessary, the level of detail could be stated in conjunction with a query and subsequent queries should be possible with either the same or increased/decreased level of detail. While increasing the coarseness of data seems like a feasible, albeit non-trivial, task, the reverse process seems far less intuitive.

Stell and Worboys [27] propose generalisation of graph like structures by defining *selection* and *amalgamation* as two different kinds of operations which are combined in a so called *simplification*. Partially based on or influenced by work of Puppo and Dettori [25] and Bertolotto [4] they apply their concepts on examples of railway networks in Britain. In this paper we try to incorporate different ideas and concepts, including the ones mentioned above, into a more general approach which is suitable for our specific problem domain of routing and way finding. ■

The Main Problem Solvers. Finally we need to adapt or develop the algorithms for solving the main problems. These range from 'shortest path in a graph' algorithms to logical calculi for reasoning with symbolic information. Fortunately most of these algorithms are well developed and can, hopefully, be taken off the shelf. ■

Graphs

Apart from classic examples based on road traffic as seen in Fig. 4, the very same principles can be applied under very different circumstances and even on a symbolic level.

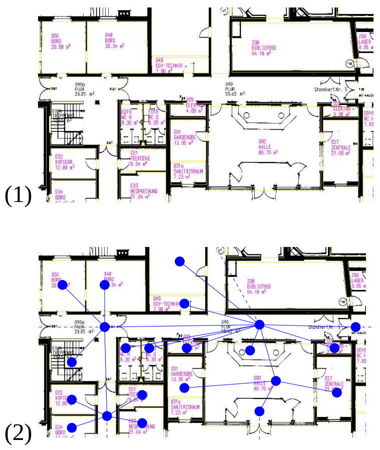

Figure 7: Plain Floor Plan without and with Network Overlay

Indoor navigation of autonomous vehicles requires a detailed floor plan, as shown in figure (1) of Fig. 7. In order to plan a way from, say, the entrance of the building to a particular office, such a detailed floor plan is not necessary. A simplified net plan, such as shown in picture (2) of Fig. 7 is much more suitable for this purpose. The simplified plan can be generated from the detailed floor plan.

Finally, one can collapse the whole building to a single node in a bigger city map. The node is sufficient for planning a path through the city to this building. In a similar way, symbolic data can be represented.

The left hand side of Fig. 8 shows the boundaries of two of the German states, and some cities. The boundaries can be represented as polygons, and these are again just graphs. In the right picture the polygons are collapsed into single nodes of a graph. The relation 'polygon A is contained in polygon B' is turned into an NTTP edge (Non Tangential Proper Part) of the new graph. The relation 'polygon A touches polygon B' is turned into an EC edge (Externally Connected) of the new graph. For more information on these RCC8 relations see [9].

Sample Queries

Apart from the query underlying the main scenario presented in section 2, we present a few examples which illustrate the different application possibilities.

Example 1. Consider the query "give me all cities *between* Munich and Frankfurt". What does *between* mean here? If we take a map of Germany and draw a straight line from Munich to Frankfurt, it does not cross many cities. A more elaborate (and still

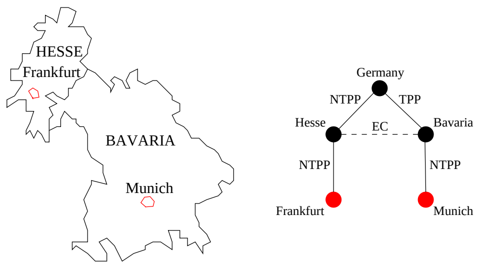

Figure 8: Symbolic Data Representation

too simple) formalisation of *between* could be: in order to check whether a city B is between the cities A and C, compute the shortest route R_1 from A to B, the shortest route R_2 from B to C and the shortest route R_3 directly from A to C. If the extra distance $d = length(R_1) + length(R_2) - length(R_3)$, I need to travel from A to C via B, compared to the direct route from A to C, is smaller than a certain *predefined threshold value*, B can be considered to be located *between* A and B. Of course, the threshold value is of discretionary nature. Also, solutions based on fuzzy values might prove useful here. In any case, the distance d could be used to order the answers to the query. ■

Example 2. Suppose a company looks for a building site for a new factory. The site should be *close to* the motorway. "Close to" does in this case of course not mean the geographic distance to the motorway. It means the time it takes for a vehicle to get to the next junction of the motorway. The length of the shortest path to the next junction can be turned into a fuzzy value which, in turn can be used to order the answers to the query.

We turned the relation $close\_to(point, line)$ into a shortest path problem whose result is then turned into a fuzzy value as the result of $close\_to(point, line)$. ■

Example 3. Suppose the database contains a road map, together with dynamic information about, say, traffic jams. The information about traffic jams is usually not very precise. It could be something like "there is a traffic jam on the M25 2 miles long between junction 8 and junction 10".

If the M25 is taken as a straight line then the traffic jam is a one-dimensional interval whose location is not exactly determined. Instead, we have some constraints: length = 2 miles, start after coordinate of junction 8, and end before coordinate of junction 10.

So queries like "is there a traffic jam on the western part of the M25" give rise to a constraint-solving problem. ■

3.2 GeoOntology - OTN

As already mentioned in section 3.1 a transport network oriented ontology is needed. MPLL uses this ontology for different aspects of the reasoning process, such as deciding what networks to use, how to handle constraints and preferences (the user is travelling with a child and baby carriage or is unwilling to take a plane), and most importantly, how to calculate and rank different routes.

As modelling transport networks is not a radically new idea, the *Transport Network Ontology* (OTN) is based on prior work in this area. The ideas and principles underlying the *Geographic Data File* format (GDF) [17] serves as the basis for OTN. For a comprehensive introduction to GDF and technical issues between GDF and OTN see [16].

GDF was primarily designed to facilitate an accurate modelling of street networks with some extensions to waterways (in the sense of an extension to the street network). Since OTN shall not be restricted to road traffic, other modes of transportation had to be represented with the main focus on seamless integration of different specificities. The already comprehensive coverage of road networks is expanded with public transport networks, such as buses, trains and air planes.

The complete description of the current state of development of the ontology is beyond the scope of this paper, nevertheless the authors want to give one example for a necessary and useful extension in the sense of interaction between MPLL and the ontology. The following paragraph deals with an aspect of transport networks, which has previously not been covered by GDF, time tables and schedules for public transport systems. Other desirable extensions are for example modelling the use and pricing of taxicabs, different pricing schemes for public transport systems, dynamics of traffic flow including congestions, and many other aspects.

Time Tables – One of the most important extensions in OTN is the modelling of time tables, which contain the operative schedule of public transport, such as ferries, trains, etc. While GDF only allows for specifying the hours of operation (from - until), this part has been substantially extended for OTN to cater for the requirements of routing tasks. As the following example shows, all necessary details defining a connection can be represented.

Each connection, such as a ferry connection or a railway segment between two stations can own a *timetable*, which contains at least one *Timetable*. Each one is valid within a certain *validity_Period*, i.e. during this time, the service is operational. From *starting_Date* on, for a period of *time_duration* the service operates every *loop_Time*. The starting node is denoted by a reference in *starts_at*, the duration of travel is defined in *travel_Time*. Optionally, *waiting_Time* specifies the idle period before departure, for example for boarding or disembarking a ferry.

The following timetable defines a service operating hourly from 6:30 to 18:30, from node A to node B, which has a travel time of 30 minutes and a waiting time of 20 minutes before departure:

```
<Timetable rdf:ID="Timetbl_AB">
  <starts_at rdf:resource="#A"/>
  <waiting_Time>m20</waiting_Time>
  <loop_Time>h1</loop_Time>
  <travel_Time>m30</travel_Time>
  <validity_Period>
    <Validity_Period rdf:ID='validity_Timetbl_AB'>
      <time_duration>h12</time_duration>
      <starting_Date>h6m30</starting_Date>
    </Validity_Period>
  </validity_Period>
</Timetable>
```

3.3 Wrappers and Local sources

The reasoning layer queries deal with:

- different locations
- bounding boxes
- overlapping networks
- secondary data (landmarks, addresses, etc.)

Adaptation of the data is necessary, since we need graphs structures for working with MPLL (examples are given in section 4). Whatever the underlying data source may be, we need to translate the communication format into a MPLL compatible one. Ideally, graphs precomputed or generated on the fly for specific purposes. The necessary attributes are different regarding different network structures, i.e. different attributes are needed for pedestrian networks as opposed to car networks or railway networks. Although, inherently similar networks have similar attributes, e.g. the schedules of buses, trams, subways, and regional trains. The graph wrapper takes a query from MPLL, and sends the corresponding one to the mediation engine. Then he takes back the answer and sends it to the reasoning layer level. We avoid efficiency problem, as the translation step is located on the server side. If directly accessing a known database, we simply get the data, and then MPLL can work with it. Using VirGIS helps in getting a specific geospatial entity without having to deal with integration problem.

VirGIS is a geographic mediation system that provides an integrated view of the data together with a spatial query language. Results are provided using GML format, and then easily translatable, into graphs structure using XML suitable operators. MPLL works on one or more graphs. Each graph represents a homo-genic network structure with certain node and edge attributes which are necessary for routing (e.g. travel times, distances, speeds, congestion information, etc.). If more than one graph is needed, i.e. whenever different modes of transportation are combined, then there are specially designated transition points which facilitate changing between different graphs (i.e. networks). These transition point have special attributes and sometimes they represent a network themselves, for example the airport buildings, which are logically located between travel by plane and the subsequent travel by taxi or train.

As shown in Fig. 9, we use a mediation engine as the intermediate layer between the reasoning engine and the local sources. When dealing with an entity hosted in a single source, the mediator transmits queries and results. If the data is distributed over several databases, the mediation procedures help in splitting the global query received from the MPLL engine, and building an integrated response from local results. In practice, when using a mobile device to solve the problem presented in section 2, we may have to query both a single data source providing the `pedestrian streets` graph, and two heterogeneous sources providing `bridges`. If `bridges` are distributed as in Fig. 9, we face both with vertical (district 1 is in GIS1 and GIS3) and horizontal (districts 1 to 6 cover the whole town) integration. This would add an integration problem to the reasoning one. That's why we choose to use VirGIS for computing the `bridges` graph structure. Next section details the different steps of the query execution when solving the "pharmacies problem".

4. A SOLUTION FOR THE USE CASE

We detail below the example presented in Fig. 1. For this classic routing problem, we deal with:

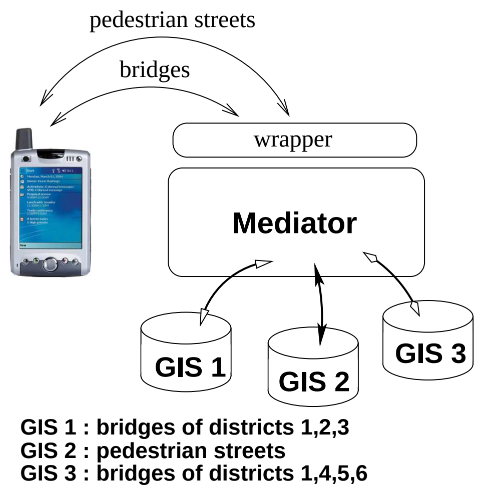
pedestrian streets
bridges
wrapper
Mediator
GIS 1
GIS 2
GIS 3
GIS 1 : bridges of districts 1,2,3
GIS 2 : pedestrian streets
GIS 3 : bridges of districts 1,4,5,6

Figure 9: Accessing the local sources

- Starting position (by context and/or device)
- Goal position(s) – this can be explicit or implicit, i.e. there might be a goal description which expresses certain qualities the goal must fulfil (e.g. opening hours, service provided, price, neighbourhood, etc. – there are countless possibilities)
- User preferences and context information

```
<GetFeature>
 <QuerytypeName="pedestrian_streets">
 <ogc:Filter><ogc:Or>
  <ogc:PropertyIsEqualTo>
  <ogc:PropertyName>district</ogc:PropertyName>
  <ogc:Literal>2</ogc:Literal>
  </ogc:PropertyIsEqualTo>
  <ogc:PropertyIsEqualTo>
  <ogc:PropertyName>district</ogc:PropertyName>
  <ogc:Literal>4</ogc:Literal>
  </ogc:PropertyIsEqualTo>
 </ogc:Or></ogc:Filter>
 </Query>
</GetFeature>
```

Figure 10: Local WFS query extracting pedestrian streets

Once the criterias are set at the higher level, we obtain a set of rules that are to be applied on graphs networks in relation with specific needs for answering the query. These rules will then be translated by the graph wrapper in order to query the right sources through VirGIS and get back the graphs. In the main example, we would need pedestrian streets and bridges covering districts 2 and 4. The query in Fig. 10 extracts suitable streets from a single data source.

Both queries in Fig. 11 are executed on distinct sources and their results are merged by the mediation engine to build a single graph, transmitted to the reasoning layer.

Global schema used by the mediation engine is based on geospatial entities described in the ontology, so we have no correspondence problems with features manipulated by the reasoning layer as explained in 3.1.

```
<GetFeature>
 <QuerytypeName="bridges">
  <ogc:Filter><ogc:PropertyIsEqualTo>
  <ogc:PropertyName>area</ogc:PropertyName>
  <ogc:Literal>2</ogc:Literal>
  </ogc:PropertyIsEqualTo>
  </ogc:Filter>
 </Query>
</GetFeature>
```

Local WFS query extracting bridges on GIS1

```
<GetFeature>
 <QuerytypeName="geo_feature_bridge">
  <ogc:Filter><ogc:PropertyIsEqualTo>
  <ogc:PropertyName>district</ogc:PropertyName>
  <ogc:Literal>4</ogc:Literal>
  </ogc:PropertyIsEqualTo>
  </ogc:Filter>
 </Query>
</GetFeature>
```

Local WFS query extracting bridges on GIS3

Figure 11: Fusion of data for building a single graph

5. CONCLUSIONS

One of the key features of the Semantic Web is that data on the web can be interpreted with respect to their meaning, their semantics. The meaning can be represented in various ways, as ontologies, as axioms in some logic, as rules in some rule language, and even with special purpose procedures. We show that for a number of problems the use of higher level reasoning mechanisms mainly based on graphs and graph algorithms is a very efficient solution, especially for mobile applications. The graph representations are to be found on different levels with respect to different applications. At the bottom end of the hierarchy we have detailed maps of the geographic entities (road maps, underground maps, floor plans etc.) At the upper end we have purely symbolic representations of concepts and relations. The correlation between the different levels is achieved by "mappings" of the lower level graphs to nodes or edges in the higher level graphs (road crossings, buildings, city boundaries etc.) These "mappings" are a part of MPLL which still needs to be fully specified and developed. The resulting description of a route, or more generally a plan, allows for rendering and presentation on all kinds of different devices with very different capabilites. Coupling the reasoning capabilities of MPLL with the VirGIS mediation engine is one of the key issues, which open new perspectives for the integrated approach. While this paper reports on work in progress, the broad possibilities for the concepts' applications in concrete systems as well as the refinement of different aspects in further research indicates its potential.

6. REFERENCES

[1] F. Baader, D. Calvanese, D. McGuinness, D. Nardi, and P. Patel-Schneider, editors. *The Description Logic Handbook*. Cambridge University Press, 2002.

[2] S. Berger, F. Bry, B. Lorenz, H. J. Ohlbach, P.-L. Pătrânjan, S. Schaffert, U. Schwertel, and S. Spranger. Reasoning on the Web: Language Prototypes and Perspectives. In *Proceedings of European Workshop on the Integration of Knowledge, Semantics and Digital Media Technology, London, U.K (25th–26th November 2004)*, pages 157–164. The Institution of Electrical Engineers, IEE, 2004.

[3] T. Berners-Lee, J. Hendler, and O. Lassila. The semantic web. *Scientific American*, 279(5):35–43, 2001.

[4] M. Bertolotto. *Geometric Modeling of Spatial Entities at Multiple Levels of Resolution*. PhD thesis, Dipartimento di Informatica e Scienze dell'Informazione, Universita di Genova, Genova, Italy, 1998.

[5] O. Boucelma and F.-M. Colonna. GQuery: A Query Language for GML. In M. R. Elfriede Fendel, editor, *24th Urban Data Management Symposium, UDMS 2004. Chioggia (Italie), 27–29 Octobre 2004.*, pages 23–32, 2004.

[6] O. Boucelma, F.-M. Colonna, and M. Essid. The VirGIS Geographic Integration System. In *Vingtièmes Journées Bases de Données Avancées, BDA 2004, Montpellier, France.*, 19–22 Octobre 2004.

[7] F. Bry, B. Lorenz, and S. Spranger. Calendars and Topologies as Types - A Programming Language Approach to Modelling Mobile Applications. In *Proceedings of 9th International Conference on Knowledge-Based Intelligent Information & Engineering System, Melbourne, Australia (14th–16th September 2005)*, 2005.

[8] F. Bry and M. Marchiori. Ten Theses on Logic Languages for the Semantic Web. In *Proceedings of W3C Workshop on Rule Languages for Interoperability, Washington D.C., USA (27th–28th April 2005)*. W3C, 2005.

[9] A. G. Cohn, B. Bennett, J. Gooday, and N. M. Gotts. Qualitative spatial representation and reasoning with the region connection calculus. *GeoInformatica*, 1(3):275–316, 1997.

[10] M. Escrig, F. Toledo, and A. Pobil. An Overview to Qualitative Spatial Reasoning. In *Current Trends in Qualitative Reasoning and Applications*, pages 43–60. International Center for Numerical Methods in Engineering, Barcelona, 1995.

[11] ESRI. GIS and mapping software. *http://www.esri.com/software/arcgis/*, 1995-2005.

[12] ESRI. ArcPad, Mobile GIS Software. *http://www.esri.com/software/arcgis/arcpad/index.html*, 2005.

[13] GeoConcept. GeoConcept Pocket 2.0. *http://www.geoconcept.com/article.php3?id_article=223*, 2005.

[14] G. Grand, G. Schiano, G. DiSapio, and M. Manent. GisMO: Mobile Access to GIS. *http://www.lsis.org/ colonnafm/projets.php*, Semester Project of Master of Computer and Information Technology.

[15] Hebrew University of Jerusalem. Lectures notes on data integration and transformation. *http://www.cs.huji.ac.il/course/2004/dit/*, 2005.

[16] F. Ipfelkofer. Basisontologie und Anwendungs-Framework für Visualisierung und Geospatial Reasoning. diploma thesis, University of Munich, Institute for Informatics, 2004.

[17] ISO/TS Standard 14825: Intelligent transport systems – Geographic Data Files (GDF) – Overall data specification. `http://www.iso.org`, February 2004.

[18] Kevin Lynch. *The Image of the City*. MIT Press, June 15, 1960.

[19] A. Y. Levy, A. Rajamaran, and J. Ordille. Querying heterogeneous information sources using source description. *In Proc. VLDB'96*, pages 252–262, 1996.

[20] I. Manolescu, D. Florescu, and D. Kossmann. Answering XML queries on heterogeneous data sources. *In Proc. VLDB'01*, pages 241–250, 2001.

[21] Mappy. Multi-media service for route planning and map editing. *http://www.mappy.com*, 1987-2005.

[22] Michelin. Via Michelin: Routes, charts and plans, tourism, hotels and restaurants in Europe. *http://www.viamichelin.com*, 2001-2005.

[23] Open GIS Consortium, `http://www.opengis.org`. *OpenGIS Web Feature Server Implementation Specification (WFS)*, 2001.

[24] Open GIS Consortium, `http://www.opengis.org`. *Geography markup language (GML) 3.0*, 2003.

[25] E. Puppo and G. Dettori. Towards a formal model for multiresolution spatial maps. In J. R. H. Max J. Egenhofer, editor, *Advances in Spatial Databases*. Springer-Verlag, Lecture Notes in Computer Science, N.951, 1995.

[26] M. Roth and P. Schwarz. Don't scrap it, wrap it! A wrapper architecture for data sources. *In Proc. VLDB'97*, pages 266–275, 1997.

[27] J. G. Stell and M. F. Worboys. Generalizing graphs using amalgamation and selection. *Lecture Notes in Computer Science*, 1651:19–34, 1999.

[28] M. Templeton, H.Henley, E.Maros, and D. V. Buer. InterViso: Dealing with the complexity of federated database access. *VLDB Journal*, 4:287–317, 1995.

[29] S. Timpf and W. Kuhn. Granularity transformations in wayfinding. In *Spatial Cognition*, pages 77–88, 2003.

[30] S. Timpf, G. S. Volta, D. W. Pollock, and M. J. Egenhofer. A conceptual model of wayfinding using multiple levels of abstraction. In I. Campari, A. U. Frank, and U. Formentini, editors, *Theories and Methods of Spatio-Temporal Reasoning in Geographic Space*, volume 639, pages 348–367. Springer-Verlag, Lecture Notes in Computer Science, 1992.

[31] W3C, `http://www.w3.org/TR/owl-guide`. *OWL Web Ontology Language*, 2005.