Knowledge-Based Spatiotemporal Linear Abstraction

Yuval Shahar[1] and Martin Molina[2]

[1] Address of corresponding author:
Section on Medical Informatics,
251 Campus Drive, Medical School Office Building (MSOB) X-215
Stanford University,
Stanford, CA 94305, USA
Tel.: +1-650-725-3393
email: shahar@smi.stanford.edu
fax: +1-650-725-7944

[2]Department of Artificial Intelligence,
Technical University of Madrid,
Madrid,
Spain
Tel +34-1-352-48-03
email mmolina@dia.fi.upm.es
fax +34-1-352-48-19

Running title: Knowledge-Based Spatiotemporal Linear Abstraction

Shahar, Y., & Molina, M. (1998). Knowledge-based spatiotemporal linear abstraction. Pattern Analysis and Applications, 1(2), 91-104.

Knowledge-Based Spatiotemporal Linear Abstraction

Abstract

We present a theoretical framework and a case study for reusing the same conceptual and computational methodology for both temporal abstraction and linear (unidimensional) space abstraction, in a domain (evaluation of traffic-control actions) significantly different from the one (clinical medicine) in which the method was originally used. The method, known as knowledge-based temporal abstraction, abstracts high-level concepts and patterns from time-stamped raw data using a formal theory of domain-specific temporal-abstraction knowledge.

We applied this method, originally used to interpret time-oriented clinical data, to the domain of traffic control, in which the monitoring task requires linear pattern matching along both space and time. First, we reused the method for creation of unidimensional spatial abstractions over highways, given sensor measurements along each highway measured at the same time point. Second, we reused the method to create temporal abstractions of the traffic behavior, for the same space segments, but during consecutive time points. We defined the corresponding temporal-abstraction and spatial-abstraction domain-specific knowledge.

Our results suggest that (1) the knowledge-based temporal-abstraction method is reusable over time and unidimensional space as well as over significantly different domains; (2) the method can be generalized into a knowledge-based linear-abstraction method, which solves tasks requiring abstraction of data along any linear distance measure; and (3) a spatiotemporal-abstraction method can be assembled from two copies of the generalized method and a spatial-decomposition mechanism, and is applicable to tasks requiring abstraction of time-oriented data into meaningful spatiotemporal patterns over a linear, decomposable space, such as traffic over a set of highways.

Key Words: Temporal abstraction, Spatial abstraction, Traffic control, Knowledge-based systems

Knowledge-Based Spatiotemporal Linear Abstraction

1. Introduction: Spatiotemporal Abstraction and knowledge reuse

In this paper, we present a case study with both theoretical and experimental aspects. The case illustrates the reuse of a conceptual and computational methodology, originally used for the task of interpretation of time-stamped data in a clinical domain, for the purpose of solving a task involving both temporal and (unidimensional) spatial pattern matching in a very different domain, monitoring of traffic- control actions. The method we reused is the *knowledge-based temporal-abstraction method* [1], which solves the task of abstraction of high-level concepts and patterns from time-oriented data, and which was applied previously mainly to clinical domains [2]. We first show how we have applied that method to solve both the temporal- and spatial-abstraction tasks in the traffic-control domain. We then generalize our results by showing how, using an existing methodology for construction of knowledge-based systems, a *spatiotemporal–abstraction method* can be configured from a generalized (linear) version of the temporal-abstraction method and a spatial-decomposition mechanism.

1.1 A guide to the paper

In Section 2, we present the domain of traffic control and the task of monitoring control-actions. Section 3 summarizes briefly the knowledge-based temporal-abstraction method. Section 4 shows how the knowledge-based temporal-abstraction method was reused to perform two different subtasks: spatial reasoning and temporal reasoning. Section 5 describes how we can configure the spatiotemporal–abstraction method from a more general version of the knowledge-based temporal-abstraction method, the *knowledge-based linear-abstraction method*. Section 6 summarizes the experiment and its conclusions.

2. The Traffic-Control Domain

The past two decades have experienced a significant demand for advanced information technology in road-transport management. Control centers for traffic management are connected on-line to devices such as detectors on roads, cameras, traffic lights, etc. Thus, operators can supervise the state of the road by consulting data bases with recent information from detectors and can affect the state of various control devices. The use of these traffic monitoring and management facilities requires sophisticated support tools for on-line operators, to help them in dealing with the information complexity and diversity of sensors and control devices. In particular, expert systems for decision support have recently been successfully introduced in this field. Existing systems include for example TRYS [3], KITS [4], ARTIST [5], SAPPORO [6], which employs a blackboard system specialized to the representation of traffic knowledge, CLAIRE [7], a context-free supervisor for traffic control, and other systems focused on traffic monitoring and control [8,9]. A useful collection of work in the first half of the 1990s on applications of Artificial Intelligence to traffic engineering can be found in Bielli, Ambrosino, and Boero's book [10].

The goal of a real-time traffic decision-support system is to propose, to traffic management center operators, control actions to eliminate or reduce problems according to the global state of the traffic. The particular traffic network we used for the modeling phase was the one used in major cities in Spain, such as in Madrid.

The type of traffic-control decision-support systems we had analyzed receive as input the following input:

1. Data from *sensors* that are located on all major roads, recording several traffic-oriented magnitudes such as speed (km/h), flow (vehicle/h) and occupancy (percentage of time the sensor is occupied). The distance between successive sensors on a freeway is around 500m. Information arrives periodically (e.g., every minute). Some of the sensors, however, might not be working all the time, so missing data at certain spatial or temporal points is a possibility.

2. Information about the current state of *control devices*. Control devices (or *control actions*) include traffic signals at intersections, traffic signals at sideways on-ramps, changeable message signs that present different messages to motorists (e.g., warning about existing congestions or alternative path advice), radio advisory systems to broadcast messages to motorists, and reversible lanes (i.e. freeway lanes whose direction can be selected according to the current and expected traffic demand).

The system supplies, as output, answers to the following questions:

1. *What happens?* The system interprets sensor data and detects the presence of a problem and a possible cause. Problems are congestions at certain locations caused by lack of capacity due to accidents, excess of demand (like rush hours), etc. The system might also supply complementary information such as the severity of a problem, the number of lanes blocked, etc., which helps to understand the problem.

2. *What to do?* The system proposes recommendations how to solve or reduce the problem. For instance it may recommend increasing the duration of a phase (e.g., green time) at a traffic signal, or it may suggest showing certain messages on some message signs to divert traffic. The system also gives explanations about why it recommends those control actions.

3. *What if?* The system answers what would happen if the operator choose to implement a particular action. The operator may propose a control action and the system suggests whether the control action might have some influence on an existing problem.

Typically, a city network is divided into two different but related subnetworks, which are supervised by different control centers: the surface streets and the freeways. In this paper the focus will be on freeway management. The expert system supervising the whole freeway network analyzes each direction of each freeway separately in such a way that the whole knowledge model is divided into submodels. Each submodel is specialized for detecting and solving problems in the particular area under control. Thus, the global expert system is composed of different specialists, called traffic-control agents, coordinated by an additional agent, called the coordinator, that integrates local proposals.

2.1. The Traffic Control-Action Monitoring Task

One of the tasks of the system is to monitor current control actions (e.g., warnings or path recommendations) to be sure that they are performing as expected and they are consistent with the traffic state. The reason is that when a problem occurs, the system proposes solutions making heuristic assumptions about the effect of the proposed control actions, but once the control action is implemented, its real effect may be different.

For example, consider the recommendation of alternative paths. In congested locations, the system might propose presenting messages to drivers recommending an alternative path. The system assumes that a certain percentage of traffic (e.g., 10% to 30%) will be diverted to the new path. However, once the messages are implemented different situations may happen due to unexpected conditions. For instance, if only few drivers follow the recommendation, it is better to remove the messages to come up with another solution. Alternatively, if the number of drivers following the recommendation is bigger than expected, a new congestion may appear as a consequence of the massive diversion. In this case, it is important to remove immediately the messages.

The task of monitoring effects of traffic control actions is called *control-action monitoring* and receives as input the recent evolution of different road parameters (speed, flow and occupancy at every sensor location) and recent control actions. It returns answers about the adequacy of particular control actions. Output answers include: (1) the effect of the control action is *appropriate* and the action must be maintained on the road because it is working properly, (2) the control action is *useless* because since the control action was implemented, traffic has not changed its behavior, (3) the control action has a *negative* side effect because a new problem has appeared as a consequence of the action, (4) the effect of the control action is still *unknown* because the action was recently implemented and it needs some more time to act, (5) the control action has already *solved* the problem and must be removed, and (6) the action has some effect but is *insufficient* to solve the problem.

Typical reasoning includes whether the number of drivers taking a certain exit is decreasing, whether the length of an existing queue is increasing, whether in a nearby location (such as related surface streets) the flow is close to a critical value and a new problem may be expected, etc. Solving this task requires temporal reasoning (e.g., about durations, rates, and trends) as well as one-dimensional spatial reasoning (e.g., about queue lengths) and a limited amount of two-dimensional spatial reasoning (e.g., about crossing of highways). Thus, control-action monitoring requires both spatial and temporal reasoning.

3. The Knowledge-Based Temporal Abstraction Method

Many domains require the collection of substantial numbers of data over time and the abstraction of those data into higher-level concepts, meaningful for that domain. The **temporal-abstraction (TA) task** concerns the specific temporal-reasoning task of context-sensitive abstraction and interpretation of time-stamped data.

For instance, in the domain of medical care, most clinical tasks require measurement and capture of numerous time-oriented patient data. It is highly desirable for an automated knowledge-based decision-support tool that assists physicians who monitor patients over significant periods to provide short, informative, context-sensitive summaries of time-oriented clinical data stored on electronic media. Such a tool should be able to answer queries at various levels of abstraction about abstract concepts that summarize the data. Data summaries are

valuable to the physician, support diagnostic or therapeutic recommendations by an automated system, and monitor plans suggested by the physician or by the decision-support system. A meaningful summary cannot use only time points, such as data-collection dates; it must be able to characterize significant features over *periods* of time, such as "2 weeks of grade-II bone-marrow toxicity in the context of therapy for potential complications of a bone-marrow transplantation event" (Figure 1) and more complex patterns. The TA task is thus an interpretation task: given time-stamped data and external events, produce context-specific, interval–based, relevant abstractions of the data (a somewhat more formal definition of the task is given later on in this section).

The TA task implies several conceptual and computational requirements:

1. both the input data and the required output abstractions might include several data types (e.g., symbolic, numeric) and can exist at various abstraction levels;

2. input data might arrive out of temporal order, and existing interpretations must be revised nonmonotonically;

3. several alternate interpretations might need to be maintained and followed over time;

4. parameters have context-specific temporal properties, such as expected persistence of measured values and classification functions (e.g., the meaning of the value LOW of the hemoglobin-state abstraction depends on the context);

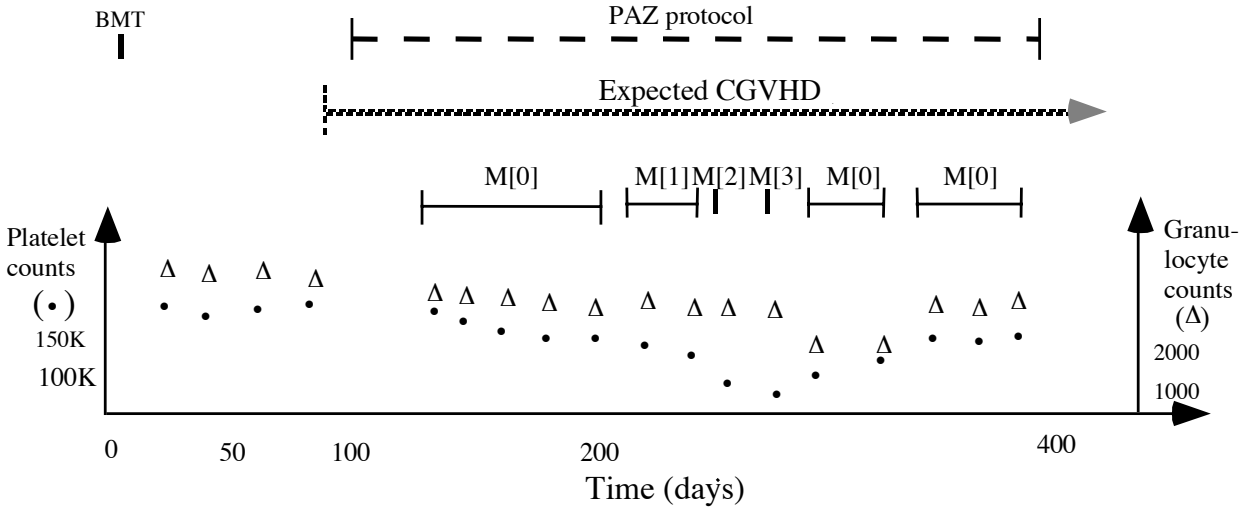

Figure 1: Abstraction of platelet and granulocyte values during administration of the PAZ clinical protocol for treating patients who have chronic graft-versus-host disease (CGVHD). The time line starts with a bone-marrow transplantation (BMT) event. The platelet and granulocyte count primitive (raw-data) parameters and the PAZ event are typical inputs to the temporal-abstraction task. The abstraction intervals and context intervals are typically part of the output. Intermediate-level abstractions such as platelet-state and granulocyte-state abstractions are not shown in this figure. • = platelet counts; Δ = granulocyte counts; ⊢ ⊣ = event; ⊢→ = open context interval; ⊢⊣ = closed abstraction interval; M[n] = myelotoxicity (bone-marrow–toxicity) grade n.

5. acquisition of knowledge from domain experts and maintenance of that knowledge should be facilitated. The method should enable *reusing* its *domain-independent* knowledge for solving the TA task in other domains, and enable *sharing* of *domain-specific* knowledge with other tasks in the same domain.

The framework we have chosen for solving the TA task is a general **problem-solving method** [11; 12] for interpreting data in time-oriented domains, with clear semantics for both the *method* and its domain-specific *knowledge* requirements: the **knowledge-based temporal-abstraction** (**KBTA**) **method** [1]. In this section, we describe briefly the fundamental principles of the KBTA method, focusing mainly on its input and output data and its required knowledge, but not on its underlying theoretical model and computational mechanisms [1], implementation [13], or evaluation [2].

The KBTA method comprises a knowledge-level representation of the TA task and of the knowledge required to solve that task. The KBTA method has a formal model of input and output entities, their relations, and their properties—the **KBTA ontology** [1].

In the TA ontology, input entities (see Figure 1) include external *event propositions* (e.g., administration of medications) interpreted over time intervals (i.e., *event intervals*), and measurable (*primitive*) or computed (*abstract*)*parameter* values (e.g., hemoglobin values), also interpreted over intervals, possibly zero-length intervals (i.e., time points). Parameters are not under control directly, and are modifiable only through external events that have an affect on them. Events can be of several *types*; each type can have a list of *arguments* that can be instantiated with values (e.g., dose). Output (and sometimes, input) entities include also *abstractions* of parameters, which may be of type *state*, *gradient*, *rate* or the more general *pattern*. An abstraction of a parameter also is a parameter (e.g., the *state* of the hemoglobin value). Abstractions must be defined within an *interpretation context* (e.g., the effect of administration of insulin or therapy by chemotherapy). *Parameter propositions* include a parameter (e.g., hemoglobin level), an abstraction type (e.g., state) a value (e.g., grade_II_toxicity) and an interpretation context (e.g., being treated by the CCTG-522 AIDS-therapy protocol). Parameter propositions are interpreted over some time interval (an ordered pair of time stamps), thus forming a *parameter interval*. Interpretation contexts are *induced* by external events, by certain parameter propositions, by the abstraction process's goals, and by certain combinations of these entities, but are not necessarily contemporaneous to the inducing entities [14].

The **TA task** is thus the following: Given a set of event, parameter, and goal intervals and the domain's TA ontology, produce an interpretation—a set of new abstractions that can answer any temporal query about all the abstractions derivable from the transitive closure of the input data and the domain's TA ontology. (A *temporal query* is a set of temporal and value constraints over a set of parameter, event, and context intervals.)

The KBTA method decomposes an external TA task internally into five parallel *subtasks* (Figure 2):

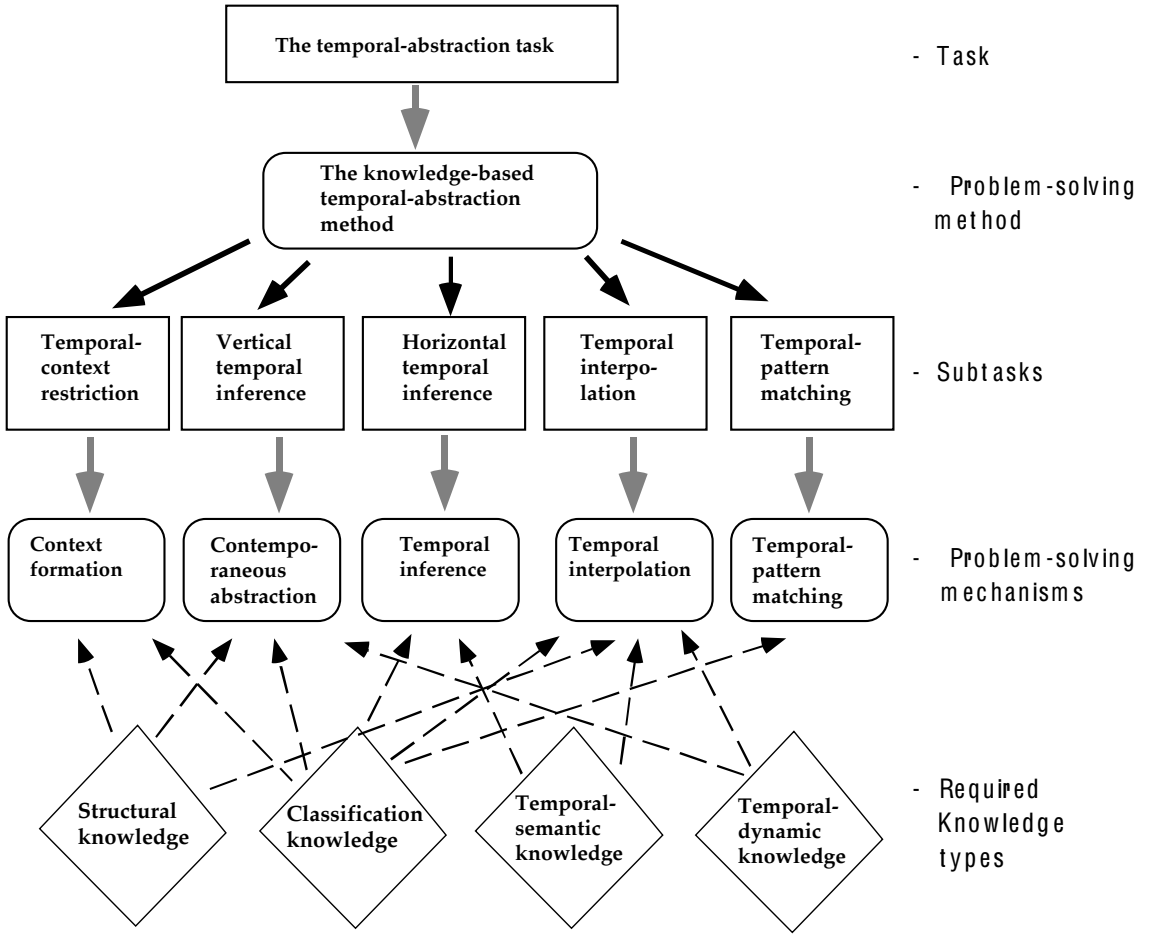

Figure 2: The knowledge-based temporal-abstraction method. The temporal-abstraction *task* is solved by this method by decomposing it into five parallel *subtasks*. Each subtask can be solved by one of five temporal-abstraction *mechanisms*. The temporal-abstraction mechanisms depend on four domain- and task-specific *knowledge types*. —> = DECOMPOSED-INTO relation; ⟶ = SOLVED-BY relation; --> = USED-BY relation.

1. **temporal-context restriction**: creation of contexts relevant for data interpretation (e.g., effect of a drug), to focus and limit the scope of the inference. The computational mechanism that solves this task assumes that the temporal scope of the interval-based interpretation contexts induced by various parameter and event propositions is represented as a set of distance constraints relative to the temporal scope of the proposition inducing the interpretation context [14].

2. **vertical temporal inference**: inference from values of contemporaneous input data or abstractions (e.g., results of several blood tests conducted during the same day) into values of higher-level concepts (e.g., classification into bone-marrow toxicity Grade II). The computational mechanism that solves this task assumes that classification functions are given.

3. **horizontal temporal inference**: inference from similar-type propositions that hold over different time intervals (e.g., joining different-value abstractions of the same parameter that hold over two meeting time intervals and computing the new abstraction's value). The computational mechanism that solves this task assumes access to knowledge about certain types of temporal-semantic constraints, such as what types of

propositions can be joined, and when does the truth of a proposition over a time interval imply the truth of that proposition during every subinterval of the original interval.

4. **temporal interpolation**: bridging of gaps between similar-type but temporally disjoint point- or interval-based propositions to create longer intervals (e.g., joining two disjoint episodes of anemia, occurring during different days, into a longer episode). The computational mechanism that solves this task assumes, in addition to other knowledge types, that part of the input is a set of domain-specific and context-specific persistence functions. Each function denotes (given the parameter, its value, and a context), for each length of parameter interval before a gap and length of parameter interval after a gap, the maximal allowed length of the gap between the two intervals that can be bridged when data are missing (usually, when no measurements exist) [15].

5. **temporal-pattern matching:** creation of intervals by matching patterns over a set of disjoint intervals over which hold propositions of various types. Patterns encode constraints regarding values, temporal relations, and temporal distances. Constraints are typically *local and global constraints* on parameter and event intervals. *Local constraints* specify, for each interval, the earliest starting shift (ESS), the latest starting shift (LSS), the earliest finishing shift (EFS), the latest finishing shift (LFS), the minimal duration (MinDu) and the maximal duration (MaxDu). Temporal shifts are measured in time units from a given absolute or relative reference time point (the default is the start of the current time line). Thus, a local temporal constraint is written as ([ESS, LSS], [EFS, LFS], [MinDu, MaxDu], REFERENCE). Local constraints include also restrictions over the parameter's value or the event's arguments' values. *Global constraints* specify qualitative temporal relations among the pattern's input intervals, such as that one interval overlaps another, quantitative constraints such as that the first interval ends at least 100 days before the second starts, and value constraints involving more than one interval, such as that the values of the bone-marrow toxicity grades in the second and third parameter intervals are both greater than that of the first parameter interval.

The five subtasks of the KBTA *method* are solved by five **temporal-abstraction mechanisms** (nondecomposable computational modules) (see Figure 2). The TA mechanisms depend on four well-defined domain-specific **knowledge types**:

1. *structural* knowledge (e.g., IS-A, PART-OF, CONTEXT-INDUCING, and ABSTRACTED-INTO relations);

2. *classification* (functional) knowledge (e.g., how raw hemoglobin values are mapped into hemoglobin states);

3. *temporal-semantic* (logical) knowledge (e.g., the CONCATENABLE property [16], which enables the potential join of meeting proposition intervals, depending on the proposition type; two consecutive weeks of coma can be summarized as a 2-week interval of coma, but 2 consecutive episodes of a 9-month pregnancy cannot be summarized as an 18-month pregnancy);

4. *temporal-dynamic* (probabilistic) knowledge (e.g., temporal persistence functions, which imply context-specific constraints with respect to bridging of gaps between temporally disjoint intervals [15]).

Values for the four knowledge types are specified as the domain's **temporal-abstraction ontology** when developing a TA system for a particular domain and task. The TA ontology is a domain-independent, but task-specific theory of the domain-specific entities, properties, and relations relevant for the TA task in each domain.

The KBTA method has been implemented as the **RÉSUMÉ** system [13] and evaluated with encouraging results in several different medical domains, such as oncology, therapy of AIDS patients, monitoring of children's growth, and monitoring of diabetes patients [2].

4. From Temporal to Spatial Abstraction

The KBTA method performs makes only few assumptions regarding the structure of time, along which it creates interval-based abstractions. For instance, it assumes for each time line a set of totally ordered *time stamps*, one of which must be a zero point, and a *time measure* with predefined granularity units (e.g., HOUR); adding or subtracting a time measure to or from a time stamp returns a time stamp.

Thus, the KBTA method can easily be applied to a domain with different linear-distance stamps and measures, as long as they comply with the algebraic constraints imposed on time lines. By changing the interpretation of the distance stamps, measure, and units to those relevant to the spatial dimension, we obtain a **knowledge-based spatial-abstraction (KBSA) method** that is useful for the control-action monitoring task.

Only few changes to the TA ontology had to be made to apply it to the spatial dimension. The most significant change was knowledge about the distance units. In particular, the time-units (temporal-granularities) conversion table had to be substituted with the corresponding spatial-metrics version. Special time stamps, such as PRESENT,

although theoretically meaningful for certain spatial tasks, were not found to be useful either (for practical purposes, it could be replaced by END). Thus, we obtained a **spatial-abstraction (SA) ontology**).

With respect to the TA computational mechanisms using the SA ontology, several slots that have certain defaults or a list of allowed time-unit symbols needed to be changed as well. Note that, although the TA mechanisms can function perfectly well in a spatial domain with internal time-oriented distance symbols such as start-time (as long as the input data is mapped to such terms from terms such as start-distance), their use for an SA task is obviously facilitated by these minor changes. Less obvious is the fact that an automated knowledge-acquisition tool generated automatically from the SA ontology, which can be created and customized using tools such as those of the PROTÉGÉ-II system [17], and which would acquire knowledge from domain experts to be used by the TA mechanisms, would use highly nonintuitive time-oriented terms if the SA ontology and corresponding links to what would be now SA mechanisms were not modified.

One can easily see at this point that a more general method ontology and related computational mechanisms could be constructed without any prior commitments as to the distance measure used, and even with several different distance measures (e.g., both time and space) within each parameter interval (see Section 2). This option will be presented when discussing the *knowledge-based linear-abstraction method* in Sections 5 and 6.

4.1. Spatial Abstraction in Traffic Domain

Given the KBSA method, we can consider each highway as a linear space. Primitive parameters (sensor measurements) along the highway, all measured at the same time, are abstracted over spatial intervals into values of abstract parameters; the relationship is represented in the traffic-domain's SA ontology (Figure 3).

The *primitive parameters* and respective units in the traffic domain are the three basic magnitudes recorded by sensors which are the inputs of the system: Speed (*km/h*), Flow (*vehicle/hr*), and Occupancy (*percentage*).

The *abstract parameters* are high-level qualitative variables representing the state of the traffic and several intermediate variables in the abstraction process, such as Saturation degree (*percentage*), Circulation (on of FLUID, UNSTABLE, CONGESTED), and Saturation level (one of FREE, CRITICAL). The Saturation level (one of FREE, CRITICAL) abstract parameter is abstracted from the Saturation degree. The Circulation parameter is abstracted from Speed and Occupancy using a 2:1 classification table.

Interpretation contexts include, for example, the number of lanes of the highway. Interpretation contexts are induced also by *events* such as the presence of an accident blocking one or more lanes, a road construction reducing the capacity of a freeway section, or the state of a reversible lane. Interpretation contexts determine how parameters such as the Saturation degree should be abstracted (e.g., the Saturation degree for one lane is 100 x Flow/1600, but is 100 x Flow/3500 for two lanes).

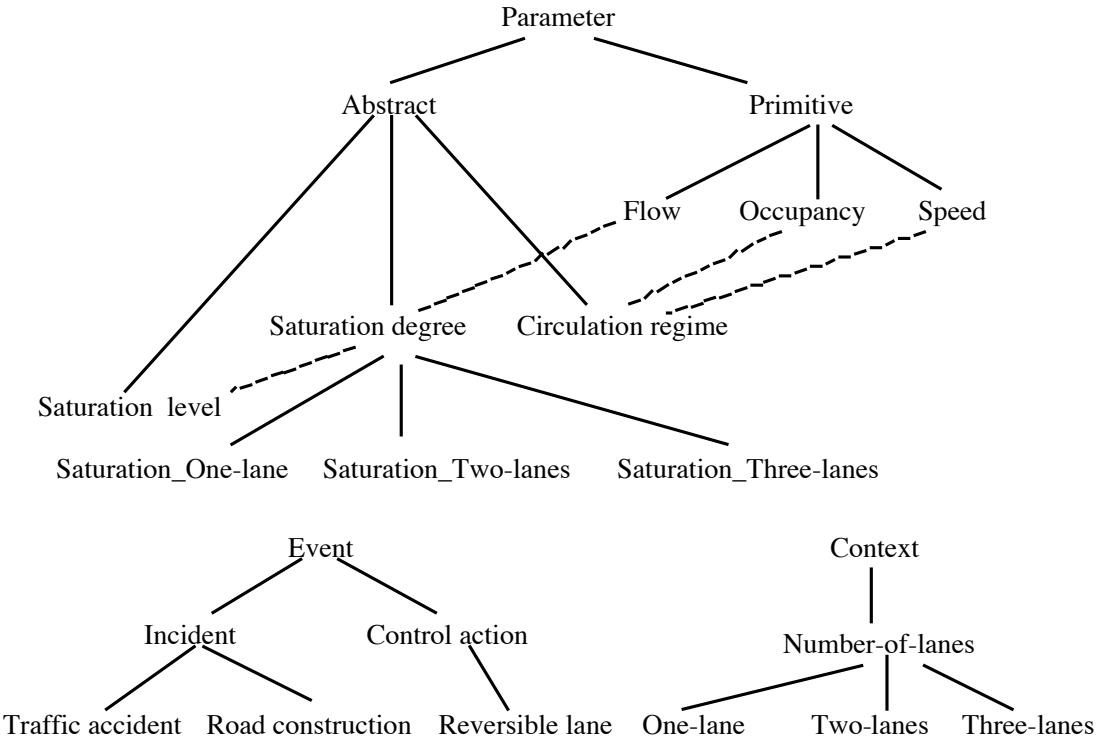

Figure 3: Part of the spatial-abstraction ontology for the traffic domain. Names represent classes, lines represent IS-A relations, dashed lines represent ABSTRACTED-INTO relations.

Other types of knowledge are represented in the traffic-domain's SA ontology, besides vertical-classification knowledge. One is the Δ *(maximal-gap) persistence function*, a dynamic knowledge type (see Section 2) which expresses the maximal distance between successive disjoint parameter intervals that still allows joining them into a new parameter interval through interpolation. Thus, in the case of the Circulation parameter and the CONGESTED value, this distance could be established as *3 km* (i.e., two Circulation-parameter intervals with the CONGESTED value would be joined into a longer interval when the distance between the endpoint locations was less than 3 km; if the distance was bigger, they would be interpreted as two different problems). This particular feature of the KBSA method is especially useful in the traffic domain since sometimes sensors do not work, certain data are missing, and the system must be able to interpolate using other sensors and heuristics.

Values for the knowledge type depend on particular highways. One approach is to consider each highway as a different interpretation context, and specialize the SA ontology by these contexts, as is done in the TA ontology and implemented in RÉSUMÉ [2]. The other approach involves defining different instances of the SA ontology for each highway, as would be done using the Knowledge Structure Manager (KSM) [3] (see Section 5) *knowledge-units*.

Using an appropriate SA ontology, the KBSA method was used to create spatial abstractions using the spatial version of RÉSUMÉ (i.e., the SA mechanisms) and values from simulated highway data sensors (Figure 4).

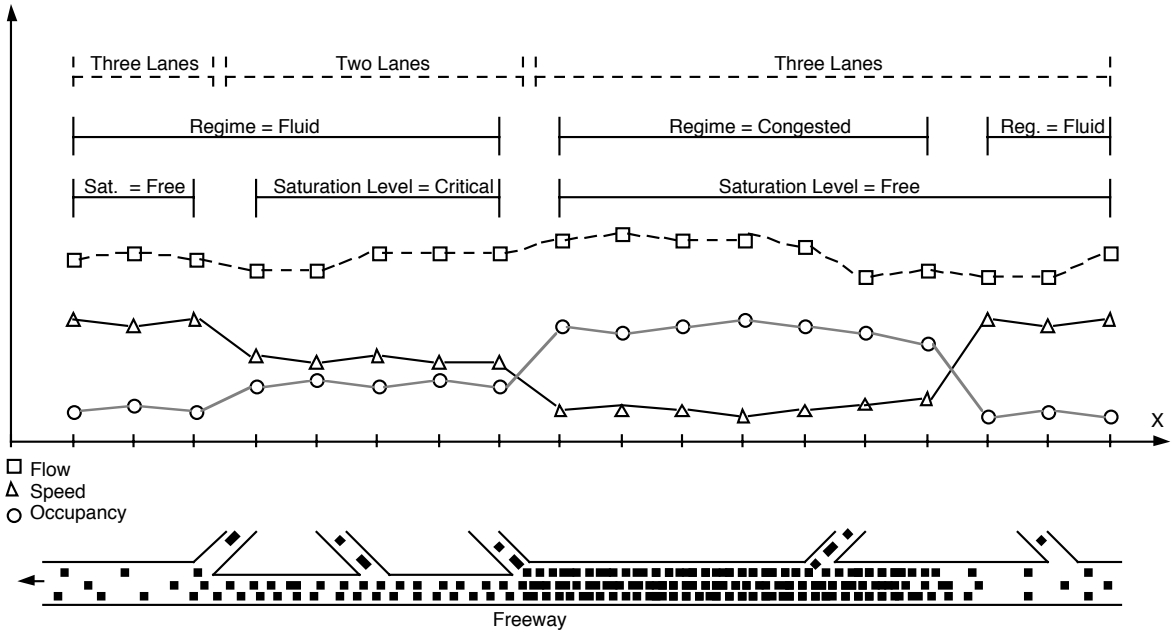

Figure 4: Spatial abstraction of a highway section. At the bottom, a scheme of the highway is presented showing different densities of traffic. Above that, respective values are presented for different magnitudes recorded by sensors at consecutive locations (speed, occupancy and flow). At the top, different intervals are shown as a result of the spatial abstraction. The figure shows only top-level abstraction intervals although different intermediate intervals are also created during the inference process. r = Flow; • = Speed; ° = Occupancy; ⊢ —| = context; |—| = closed abstraction interval.

In summary, the first part of the solution for the traffic-control task (i.e., the solution of the SA task), shows how the KBTA method and its TA ontology were transformed (within a few days) into the KBSA method and its SA ontology, and were applied to traffic control. The KBSA method provides a rich representation and inference to easily model the knowledge involved in traffic domain for linear spatial abstraction.

4.2 Temporal Abstraction in The Traffic Domain

In addition to the SA task, the control-monitoring task requires also a solution for a TA task to determine conclusions the adequacy of control actions. This subtask receives as input a set of qualitative instantaneous views of the highway (Figure 5) which are the output of the KBSA module, corresponding to consecutive time instants, and determines the adequacy of the current control action by abstracting these views over time.

In this case, the primitive parameters include values provided by the output of the SA task and values provided by sensors at critical points outside the highway, such as ramps or intersections: Congestion length (*meters*), Flow at point *P*i (*vehicles/hr*) (the number *i* of these points is usually less than 5 per highway).

For the sake of clarity, we assume that a highway can have at most one problem at a time. In fact this is normally true. However, reasoning about multiple problems is not difficult; several *zones*, as they are often called, must be defined, with each zone corresponding to a spatial interval between two consecutive message-sign devices. Zones can be represented as *subcontexts* (a part of the TA and SA interpretation-context ontologies). A traffic queue usually has a fixed starting point where there is a lack of capacity (an accident, a bottleneck, etc.) and the end of the congested area evolves according to the demand. This means that if there were several problems on the same highway, each one could be identified by the zone of their starting point.

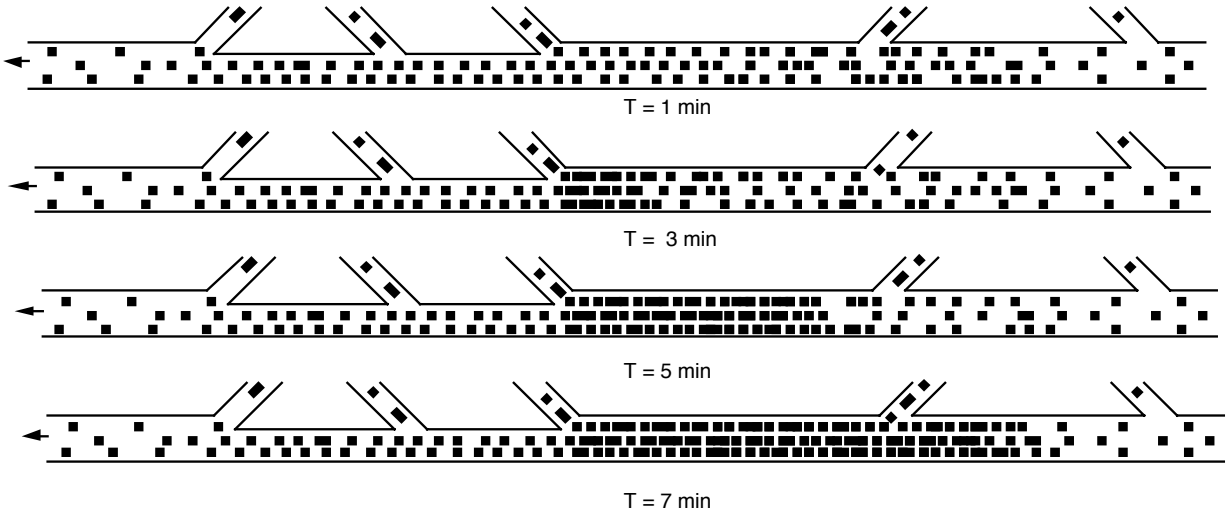

Figure 5: The input for the temporal abstraction task in the domain of traffic is a sequence of qualitative instantaneous views of the same segment of the highway. The figure presents the temporal evolution of the state of a highway in which the congestion-length gradient in a certain zone can be abstracted as INCREASING over successive temporal snapshots (at T=1min, T= 3min, T= 5min, and T= 7min).

The abstract parameters of the TA ontology for the control-monitoring task in the traffic domain include: Congestion-length gradient (one of INCREASING, DECREASING, CONSTANT); Flow gradient at point Pi (one of INCREASING, DECREASING, CONSTANT); Saturation level at point Pi (one of FREE, CRITICAL) (Figure 6).

The Congestion-length gradient is necessary to decide if the control action is having an effect on the existing problem. Flow gradients monitor whether control actions such as diversion are followed by drivers. The Saturation level at critical points is useful to determine whether a new problem may appear as a consequence of the control action. Vertical-classification tables for the Saturation-level are specialized by each subcontext created by each point *Pi*.

Interpretation contexts are also induced by events (execution of traffic-control actions), such as a turning on a congestion warning at a certain zone or creating a path diversion.

The horizontal-inference knowledge for gradient interpolation includes values of variations significant to the values of the parameters abstracted (e.g., 1000m for Congestion length, 500 vehicles/hr for the Flow parameter).

Finally, to determine the adequacy of a control action it is necessary to define *temporal patterns*. Using the terminology introduced in section 3.1, for each control action we defined the following set of TA patterns representing its adequacy: APPROPRIATE, USELESS, NEGATIVE, UNKNOWN, SOLVED and INSUFFICIENT.

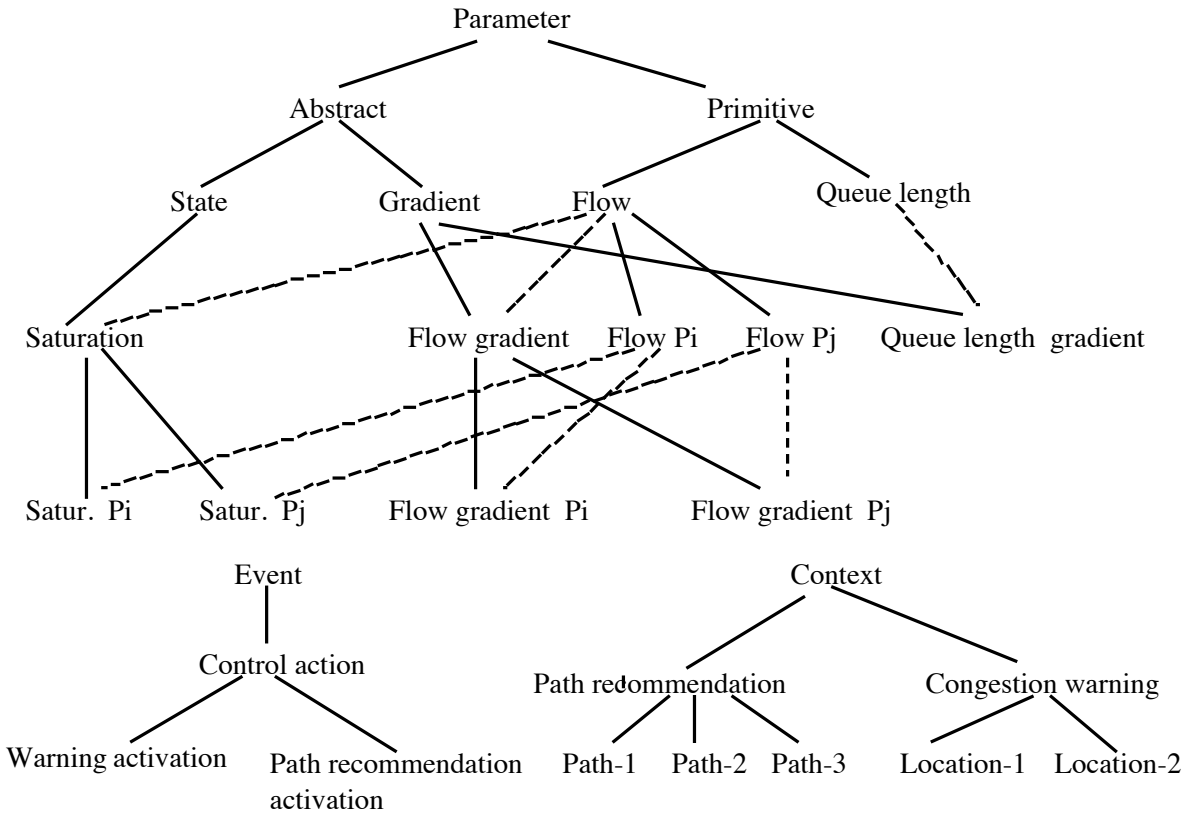

Figure 6: Part of the temporal-abstraction ontology for the traffic domain. Each name represents a class, a normal line represents an IS-A relation and a dashed line represents an ABSTRACTED-INTO relation.

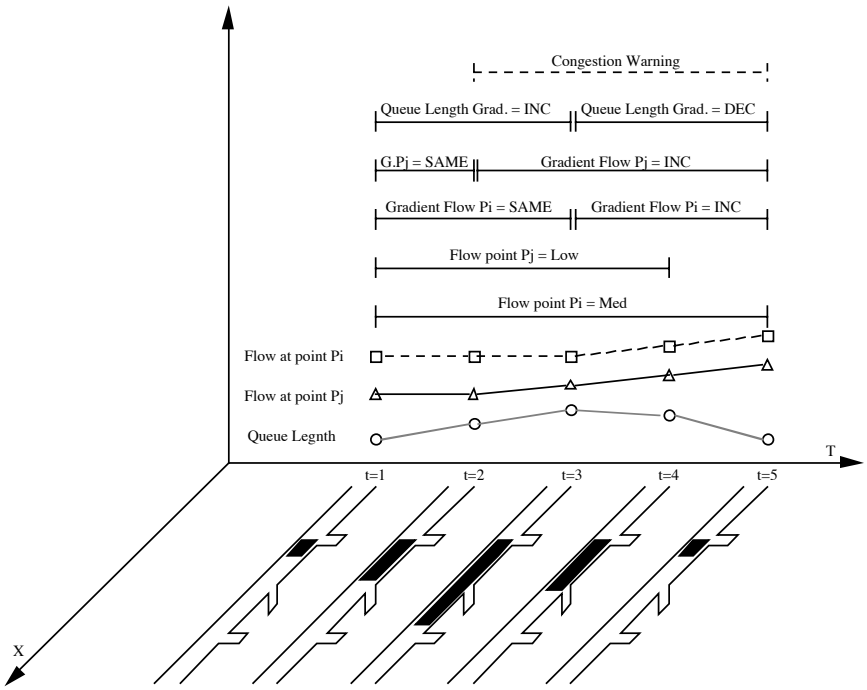

Figure 7: Temporal abstraction of a highway section. At the bottom is the evolution of the highway spatial-abstraction state over time; a queue first increases and then decreases. Above that, values are presented for the queue length and for values measured by sensors at critical spatial locations. At the top are inferred temporal abstraction intervals. ⊢ —| = context; |—| = closed abstraction interval.

Each pattern is expressed as a set of parameter intervals with temporal and value constraints among them. The values of these patterns (typically one of TRUE, FALSE) supplied the final answer to the control-action monitoring task.

Figure 7 shows an example of a temporal abstraction of the spatial data abstracted from one highway section, showing an evolution of its (abstract) parameters over time.

In summary, we reused the KBTA method again within the control-action monitoring task of the traffic domain., this time, to solve a TA task. The knowledge model uses the *TA ontology* of the traffic domain, including TA properties of parameters that are part of the *SA ontology* for that domain, to determine if executed control actions are adequate and consistent with the traffic behavior.

5. Assembling Problem-Solving Methods: The Spatiotemporal-Abstraction method

The previous sections have shown how the KBTA method was reused to perform two different subtasks of the control-action monitoring task in the traffic domain. Note that each of the two versions, spatial and temporal, has its own particular knowledge model (ontology). Thus, for a particular highway, the same method must be applied in two different ways. In addition, since the complete system supervises a set of highways, this process must be repeated several times, as many as the number of highways. This section shows how all those subtasks can be assembled to solve the control-action monitoring task using a new, higher-level problem-solving method. We will demonstrate this assembly using the PROTÉGÉ-II framework.

Knowledge-based architectures present unique opportunities for software reuse. Several theoretical and practical frameworks for knowledge reuse have been proposed. One such framework is **PROTÉGÉ-II** [17; 18; 12]. PROTÉGÉ-II uses a library of domain independent *problem-solving methods*, which can be reused to solve different domain-dependent tasks by defining explicit mappings between *method ontologies* (i.e. a theory of entities, their properties, and their relations) and *domain ontologies*. Graphical tools for acquiring the domain-specific knowledge required by the selected problem-solving method are then generated automatically, customized for the particular domain ontology.

Another approach for configuration of knowledge-based systems is the **Knowledge Structure Manager** (**KSM**) environment [19]. In KSM, the developer can create generic abstract knowledge structures that can be applied to different specific domains by duplicating and configuring their components. The **Knowledge Reuse Tool** (**KREST)** is another software-reuse environment, based on the components-of-expertise approach [20]. KREST presents a knowledge-level description of an application and assists non-programmer users in reusing parts to develop applications. Other approaches to the design and configuration of knowledge-based systems exist, such as the **KADS** project [21].

5.1. The PROTÉGÉ-II Approach

PROTÉGÉ-II is a development environment and methodology for the construction of knowledge-based systems with reusable components (see Section 1). This section shows first, how in PROTÉGÉ-II, knowledge components can be constructed by reusing other components and second, how to assemble them in order to solve the global task.

In PROTÉGÉ-II, a method is a domain independent description of how to solve a problem. Methods have their own ontologies (inputs, outputs and required knowledge). Methods decompose tasks into subtasks. A method that solves a task without further decomposition is a mechanism. The declarative domain knowledge (concepts, properties, and relations) is defined by a domain ontology. At least a part of the domain ontology is method independent, and can be used by different methods to carry out several tasks. New method-dependent concepts might be added to enable the application of a specific method, thus creating an application ontology. The KBTA method (see Figure 2) fits the PROTÉGÉ-II model well.

The developer uses PROTÉGÉ-II to define the application ontology. Given the structure of that ontology, a graphical *knowledge acquisition tool* is generated automatically (with optional customization) for the acquisition of the specific domain knowledge. In addition, the developer defines *mapping-relations* between the method ontology and the application ontology. Mapping relations explicitly show how the domain-independent method is applied to the particular domain to perform the task. Mapping-relations express the *role* that the relevant domain concepts play in the method. The clear distinction between methods and domains facilitates both reuse and sharing.

5.2 Generalization: The Knowledge-Based Linear-Abstraction Method

Given the initial results of the study in the clinical and traffic domains, we generalized the KBTA method into a slightly more general method, the **knowledge-based linear abstraction** (**KBLA**) **method**. In the KBLA method, knowledge about the dimension, or distance measure along which data should be abstracted (e.g., time, space), is an additional knowledge role to be mapped to the domain ontology. The KBLA method has a method ontology identical to the TA or SA ontologies, but uses a distance measure which must be linear and must comply with several formal properties of timelines, such as a zero point, a total order of distance stamps, and certain algebraic relations (e.g., adding a distance measure to a distance stamp results in a distance stamp). The KBLA method receives as input a set of values of parameters at different levels of abstraction with associated time intervals, and a set of event intervals (data can arrive out of order). The KBLA method returns abstractions and answers to pattern-matching queries over the predefined distance measure. There are also certain knowledge requirements that are part of the KBLA method's assumptions; for instance, there needs to exist domain-specific knowledge for abstracting parameter values into higher level parameter values, knowledge about relevant contexts and their relations, etc.

Thus, in order to model spatial and temporal abstraction in the traffic domain, the developer must define mapping-relations between the KBLA method and the traffic domain. For instance, speed, flow and occupancy play the role of primitive parameters, and the relation between speed, occupancy and circulation is mapped to a relation of type ABSTRACTED-INTO; the associated with it a 2:1 function is mapped into a vertical-classification table. One set of mapping relations describes how to perform the SA task and another set of mapping relations describes the TA task. Both subtasks use the same application ontology but with different mappings. In addition, if the final application supervises several areas (for example 20 sections of highways) there will be either up to 20 different contexts (or up to 20 different knowledge bases, if we were to use the KSM methodology). The next subsection shows how to assemble methods to build the final application.

5.3 Assembly with PROTÉGÉ-II

Although the current version of the PROTÉGÉ-II architecture does not yet provide automated support to developers for assembly of existing library problem-solving methods to create new ones, we can use the PROTÉGÉ-II theoretical approach (task-methods-subtasks-mechanisms) to show how to assemble components to create the final application. (In practice, the configuration process was performed manually.)

The model is presented in figure 8. The implicit domain-specific top-level task is *control-action monitoring* of a network of highways. We solve that task with a new method—the **knowledge-based spatiotemporal abstraction** (**KBSTA**) **method**, which formally solves the **multiple-region spatiotemporal-abstraction task**. The KBSTA method reasons over a set of linear spatial regions and performs both spatial and temporal abstractions. The KBSTA method decomposes the task it solves into three subtasks: *decomposition*, *spatial abstraction* and *temporal abstraction*. The first subtask is necessary to successively decompose the global abstraction process into spatial *regions* (each region will be mapped to a particular section of a highway). This subtask is performed by the **decompose by regions** mechanism. Then, both the SA and the TA tasks are solved by the KBLA method. The KBLA *method* will be considered for this example as a *mechanism*, i.e. it will not

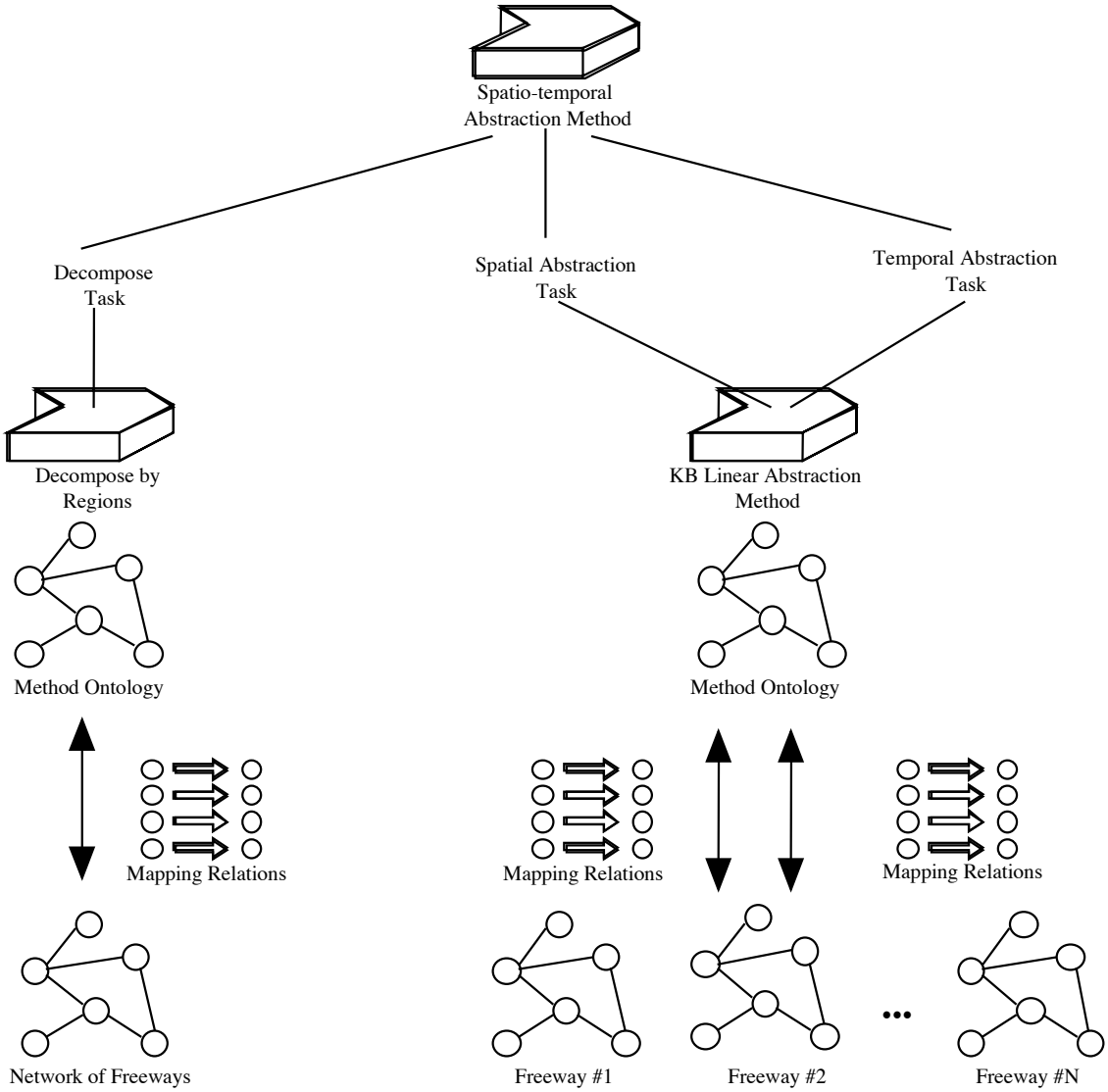

Figure 8: Assembly of the spatiotemporal-abstraction problem-solving method using PROTÉGÉ-II. The method decomposes its task into three subtasks: (spatial) decomposition, spatial abstraction and temporal abstraction. The last two subtasks are solved by knowledge-based linear abstraction, each time using a different set of mapping relations to the same domain, and thus a different knowledge base.

decompose its task into subtasks. Each of the three mechanisms has its own ontology which has to be mapped to the particular domain, in this case, the traffic domain. In the case of the KBLA mechanism there are two sets of mapping-relations, one for the SA task and another for the TA task. The final application will have several subparts of each ontology, modeled as specializing contexts, or several knowledge bases, one for each region. Thus, at runtime, the KBLA method will have to select the appropriate mapping and knowledge base.

The new, composite KBSTA method needs to include *control knowledge* that defines how to use the mechanisms during its reasoning. The control knowledge represents (1) how subtasks are connected, i.e. how outputs of some tasks are inputs of others (in our example for instance, the output of the spatial abstraction is the input of the temporal abstraction), (2) the way in which subtasks are executed (in this case it is a loop that performs sequentially in each time step the subtasks *decompose by regions*, SA, and TA), and (3) what mapping relations and ontologies must be used by methods each time. Thus, in each reasoning cycle, the KBLA method

must select one kind of mappings (spatial or temporal) and one region, or context (e.g., a highway section) in the appropriate ontology. The algorithm used for spatiotemporal abstraction, given a regional decomposition $\Re$ and a set of spatiotemporal measurements, can be summarized as follows:

1. For each region $R_i \in \Re$ and set of time stamps T_i during which the region R_i has been observed:
   1.1. For each time-stamp $\tau_{ij} \in T_i$,
      Apply the KBLA method to the (spatial) parameter values set measured over R_i at time τ_{ij}, each measured at spatial-distance measure D_i, using the knowledge of spatial-abstraction ontology SAO_i, to produce a set of regional spatial abstractions RSA_{ij} (i.e., for region R_i at time τ_{ij});
   1.2 Apply the KBLA method to the (temporal) parameter values set RSAij (where $\tau_{ij} \in T_i$) measured throughout the set of time stamps T_i, each measured at time τ_{ij}, using the knowledge of temporal-abstraction ontology TAO_i, to produce a set of regional temporal abstractions RTA_i (e.g., to evaluate traffic patterns in region R_i over time period T_i).
2. Match global spatiotemporal patterns defined over one or more regions $R_i \in \Re$, if existent, using the knowledge of temporal-abstraction ontology $TAO_{\Re}$, to produce a set of multiregional spatiotemporal abstractions (e.g., to evaluate control actions over several highways over a time period).

There are still several open problems for assembling tasks using PROTÉGÉ-II. One is the appropriate representation to model the control knowledge. Another is how to determine the method ontology of the global KBSTA method. The global ontology is not just the union of the included-methods' ontologies. With a simple union, in our example the spatial and temporal distinctions would disappear. This suggests an explicit mapping between tasks and method. This mapping would establishing the role of the method in a particular task.

6. Summary and Conclusions

This experiment and analysis demonstrate the high level of reusability of the KBTA method. The KBTA method has a domain-independent, but task-specific, model of TA constraints (the four knowledge types). The method has been shown to be reusable in very different domains for abstraction of high-level concepts from time-oriented data in a context-sensitive manner. Originally, it was created to be applied in clinical domains, where it was used to build several applications involving interpretation of clinical time-oriented data in the domains of oncology, experimental AIDS therapy, monitoring of children's growth , and diabetes care [[13; 2]. In this paper, we showed how it was applied to a very different domain—traffic management; in addition, the new domain involved reasoning along both the time and space dimensions. In order to do so, a significant step involved the translation of the TA reasoning into a new dimension, space instead of time, to solve the task of abstraction of unidimensional spatially-oriented data, creating the KBSA method.

The results of the initial experiments then led us to the proposal and configuration of the more general KBLA method, which abstracts data along any linear distance measure. Solving the traffic-domain control-action monitoring task requires two versions of this method, one for reasoning about time and another for space. These two versions can be assembled to create the new KBSTA method for spatiotemporal abstraction.

Besides demonstrating the reusability of the KBTA method, the results provide a good example for the different requirements of reuse and assembly of high level components, using as a paradigm the PROTÉGÉ-II task/method framework. The same generic method was mapped to two different dimensions (temporal and spatial), each version being used for a different task (SA, TA); finally, all of these components can be assembled to create the final application.

The detailed process of assembly in PROTÉGÉ-II is still an open question that for the moment is solved by *ad hoc* procedures. The example we use here illustrates well several interesting requirements for assembly to be used in the future for characterizing in detail this activity. One issue is that, during run time, since a particular method may be applied in different ways, control knowledge must indicate the knowledge base and mapping-relations to be used by the method each moment (while considering that this information may be the result of a previous task). Another requirement the example shows is that, in order to determine the ontology of a composite method, it may be necessary to define method-task mappings.

Finally, developers using the KBSTA method need to consider the issue of efficiency. In the traffic domain, a decision-support system must operate in real time, receiving every minute data from sensors. In our prototype model, whenever the method is executed (1) the particular knowledge base with mapping-relations is selected, (2) data are mapped from domain ontology to method ontology, (3) the method is executed, (4) results are mapped from the method ontology to the domain ontology. To be efficient, an appropriate technical solution for these four steps must be used. For instance, it may be best to store in memory several predefined mappings to knowledge bases to provide quick access.

In summary, this study has shown how a domain-independent conceptual and computational methodology for representation and use of temporal-abstraction knowledge was reused both for two very different domains (medicine and traffic control) and for two unidimensional distance measures (time and linear space). In the process, the temporal-abstraction method was generalized into a linear-abstraction one. In addition, using the methodology of the PROTÉGÉ-II framework for construction of knowledge-based systems, we have demonstrated that task-specific but domain-independent problem-solving methods, such as the KBTA method, provide high level building blocks that assist in both the development and maintenance of large knowledge-based applications, such as abstraction of meaningful, domain-specific, high-level spatiotemporal patterns.

Acknowledgments

This work has been supported by grants LM05305, LM05708, and LM06245 from the National Library of Medicine, and award IRI-9528444 from the National Science Foundation. We thank Samson Tu and Mark Musen for useful discussions.

References

[1] Shahar Y. A framework for knowledge-based temporal abstraction. Artificial Intelligence 1997; 90(1–2):79–133.

[2] Shahar Y and Musen MA. Knowledge-based temporal abstraction in clinical domains. Artif Intell Med 1996; 8(3): 267-298.

[3] Cuena J., Hernandez J., and Molina M. Using knowledge based models for adaptive traffic management

systems. Transportation Research 1995; part C.

[4] Cuena J, Ambrosino G, Boero M. A general knowledge-based architecture for traffic control: The KITS approach. In Proceedings of The International Conference on Artificial Intelligence Applications in Transportation Engineering, San Buenaventura, California, 1992.

[5] Deeter DL and Ritchie SG. A prototype real-time expert system for surface street traffic management and control, ASCE. In Proceedings of Third International Conference on Applications of Advanced Technologies in Transportation Engineering, Seattle, Washington, 1993.

[6] Wild, B. SAPPORO—a framework for intelligent integrated traffic management systems. In Bielli M, Ambrosino G, and Boero M (eds). Artificial Intelligence Applications to Traffic Engineering. VSP: AH Zeist, The Netherlands, 1994.

[7] Scemana, G. CLAIRE: a context-free Al based supervisor for traffic control. In Bielli M, Ambrosino G, and Boero M (eds). Artificial Intelligence Applications to Traffic Engineering. VSP: AH Zeist, The Netherlands, 1994.

[8] Bruno G and Improta, G. Urban traffic control: current methodologies. In Bielli M, Ambrosino G, and Boero M (eds). Artificial Intelligence Applications to Traffic Engineering. VSP: AH Zeist, The Netherlands, 1994.

[9] Bielli M, Ambrosino G, and Boero M (eds). Artificial Intelligence Applications to Traffic Engineering. VSP: AH Zeist, The Netherlands, 1994.

[10] Ambrosino G, Bielli M, and Boero M. Artificial intelligence approach to road traffic control. In Bielli M, Ambrosino G, and Boero M (eds). Artificial Intelligence Applications to Traffic Engineering. VSP: AH Zeist, The Netherlands, 1994.

[11] McDermott J. Preliminary steps toward a taxonomy of problem-solving methods. In: Marcus S (ed), Automating Knowledge Acquisition for Expert Systems. Kluwer: Boston, MA, USA,1988.

[12] Eriksson H., Shahar Y, Tu SW, Puerta AR, Musen, M.A. Task modeling with reusable problem-solving methods. Artificial Intelligence 1995; 79 (2):293–326.

[13] Shahar, Y., and Musen, M.A. RÉSUMÉ: A temporal-abstraction system for patient monitoring. Computers and Biomedical Research 1993; 26:255–273. Reprinted in van Bemmel, J.H., and McRay, A.T. (eds) (1994), Yearbook of Medical Informatics 1994, pp. 443–461, Stuttgart: F.K. Schattauer and The International Medical Informatics Association.

[14] Shahar Y. Dynamic temporal interpretation contexts for temporal abstraction. Annals of Mathematics and Artificial Intelligence, in press.

[15] Shahar Y. Knowledge-based temporal interpolation. Proceedings of The 1997 Fourth International Workshop on Temporal Representation and Reasoning (Time-97), IEEE Computer Society Press: Los Alamitos, CA, USA, 1997: 102–111.

[16] Shoham Y. Temporal logics in AI: Semantical and ontological considerations. Artificial Intelligence 1987; 33(1):89–104.

[17] Musen, MA, Gennari JH., Eriksson H, Tu SW, Puerta AR. PROTÉGÉ-II: Computer Support For Development Of Intelligent Systems From Libraries of Components. In Proceedings of MEDINFO '95, Eighth World Congress on Medical Informatics, , Vancouver BC, 1995, pp 766–770.

[18] Tu SW, Eriksson H, Gennari J, Shahar Y, Musen M.A. Ontology-based configuration of problem-solving methods and generation of knowledge-acquisition tools: Application of PROTÉGÉ-II to protocol-based decision support. Artificial Intelligence in Medicine 1995; **7**(3):257–289.

[19] Cuena J, and Molina M. A tool concept for knowledge oriented design of applications, Technical Report FIM 78.1 IA/93, Facultad de Informatica, Universidad Politecnica, Madrid, 1993.

[20] Steels L. Components of Expertise. AI Magazine 1990; 11 (2):29–49.

[21] Weilinga B, Schreiber AT, Breuker. KADS: A modeling approach to knowledge engineering. Knowledge-Acquisition **4**(1) (1992) 5–53.

Figure Captions

Figure 1: Abstraction of platelet and granulocyte values during administration of the PAZ clinical protocol for treating patients who have chronic graft-versus-host disease (CGVHD). The time line starts with a bone-marrow transplantation (BMT) event. The platelet and granulocyte count primitive (raw-data) parameters and the PAZ event are typical inputs to the temporal-abstraction task. The abstraction intervals and context intervals are typically part of the output. Intermediate-level abstractions such as platelet-state and granulocyte-state abstractions are not shown in this figure. • = platelet counts; • = granulocyte counts; ⊢ —| = event; |⟶ = open context interval; |—| = closed abstraction interval; M[n] = myelotoxicity (bone-marrow–toxicity) grade n.

Figure 2: The knowledge-based temporal-abstraction method. The temporal-abstraction *task* is solved by this method by decomposing it into five parallel *subtasks*. Each subtask can be solved by one of five temporal-abstraction *mechanisms*. The temporal-abstraction mechanisms depend on four domain- and task-specific *knowledge types*. —> = DECOMPOSED-INTO relation; ⟶ = SOLVED-BY relation; --> = USED-BY relation.

Figure 3: Part of the spatial-abstraction ontology for the traffic domain. Names represent classes, lines represent IS-A relations, dashed lines represent ABSTRACTED-INTO relations.

Figure 4: Spatial abstraction of a highway section. At the bottom, a scheme of the highway is presented showing different densities of traffic. Above that, respective values are presented for different magnitudes recorded by sensors at consecutive locations (speed, occupancy and flow). At the top, different intervals are shown as a result of the spatial abstraction. The figure shows only top-level abstraction intervals although different intermediate intervals are also created during the inference process. r = Flow; • = Speed; ° = Occupancy; ⊢ —| = context; |—| = closed abstraction interval.

Figure 5: The input for the temporal abstraction task in the domain of traffic is a sequence of qualitative instantaneous views of the same segment of the highway. The figure presents the temporal evolution of the state of a highway in which the congestion-length gradient in a certain zone can be abstracted as INCREASING over successive temporal snapshots (at T=1min, T= 3min, T= 5min, and T= 7min).

Figure 6: Part of the temporal-abstraction ontology for the traffic domain. Each name represents a class, a normal line represents an IS-A relation and a dashed line represents an ABSTRACTED-INTO relation.

Figure 7: Temporal abstraction of a highway section. At the bottom is the evolution of the highway spatial-abstraction state over time; a queue first increases and then decreases. Above that, values are presented for the queue length and for values measured by sensors at critical spatial locations. At the top are inferred temporal abstraction intervals. ⊢ —| = context; |—| = closed abstraction interval.

Figure 8: Assembly of the spatiotemporal-abstraction problem-solving method using PROTÉGÉ-II. The method decomposes its task into three subtasks: (spatial) decomposition, spatial abstraction and temporal abstraction. The last two subtasks are solved by knowledge-based linear abstraction, each time using a different set of mapping relations to the same domain, and thus a different knowledge base.