# Monitoring Mobile and Spatially Distributed Cyber-Physical Systems

Ezio Bartocci
Technische Universität Wien
Wien, Austria

Luca Bortolussi
Università di Trieste
Trieste, Italy

Michele Loreti
Università di Firenze
Florence, Italy

Laura Nenzi
Technische Universität Wien
Wien, Austria

## ABSTRACT

Cyber-Physical Systems (CPS) consist of collaborative, networked and tightly intertwined computational (logical) and physical components, each operating at different spatial and temporal scales. Hence, the spatial and temporal requirements play an essential role for their correct and safe execution. Furthermore, the local interactions among the system components result in global spatio-temporal emergent behaviors often impossible to predict at the design time. In this work, we pursue a complementary approach by introducing STREL a *novel spatio-temporal logic* that enables the specification of spatio-temporal requirements and their monitoring over the execution of *mobile* and *spatially distributed* CPS. Our logic extends the Signal Temporal Logic [15] with two novel spatial operators *reach* and *escape* from which is possible to derive other spatial modalities such as *everywhere, somewhere* and *surround*. These operators enable a monitoring procedure where the satisfaction of the property at each location depends only on the satisfaction of its neighbours, opening the way to future distributed online monitoring algorithms. We propose both a *qualitative* and *quantitative* semantics based on *constraint semirings*, an algebraic structure suitable for constraint satisfaction and optimisation. We prove that, for a subclass of models, all the spatial properties expressed with *reach* and *escape*, using euclidean distance, satisfy all the model transformations using rotation, reflection and translation. Finally, we provide an offline monitoring algorithm for STREL and, to demonstrate the feasibility of our approach, we show its application using the monitoring of a simulated mobile ad-hoc sensor network as running example.

## CCS CONCEPTS

• **Computer systems organization** → **Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability;

## KEYWORDS

Runtime Verification, Monitoring, Cyber-Physical Systems, Spatio-Temporal Logic.

## 1 INTRODUCTION

From micro- and nano-scale cyber and physical/biological materials to self-driving cars, smart factories and smart cities, cyber-physical systems (CPS) are reshaping the way in which we perceive and interact with our physical world, becoming ubiquitous in our society. CPS consist of collaborative, networked, spatially distributed, and tightly intertwined computational (logical) and physical components, each operating at different spatial and temporal scales. Therefore, the spatial and the temporal requirements are fundamentals for their safe and correct execution.

The openness of CPS with the possibility for new actors to join or to leave the system, the local interactions among the system components and the unknown environment in which they operate may cause undesired spatio-temporal emergent behaviours (i.e., congestion) often impossible to predict at the design-time. Indeed, their complexity restricts the exhaustive verification of their models runtime only to relatively small examples. Here, we pursue a complementary approach by introducing the *Spatio-Temporal Reach and Escape Logic* (STREL), a novel formal specification language that enables to express in a concise way complex spatio-temporal requirements and to monitor them for the first time (to the best of our knowledge) over the execution of *mobile and spatially distributed CPS.*

The idea of the proposed framework stems from the attempt to generalise and to overcome some limitations of the *Spatio-Signal Temporal Logic* (SSTL) previously introduced in [18]. SSTL extends the Signal Temporal Logic [15] with modalities (named *somewhere* and *surround*) to express also *spatial properties* and it is interpreted over a discrete model of the space, represented as a finite undirected graph. Each node represents a location in the space, characterised by a set of signals whose evolution can be observed in time, while each edge of the graph is labelled with a positive weight, that can be used to represent the distance between two nodes. This provides a metric structure to the space in terms of shortest path distances, enabling to monitor also spatial properties. However, since the topology of the graph in SSTL is assumed to be static, one main limitation is the impossibility to monitor nodes changing locations. Furthermore,

monitoring of spatial properties is performed on each location by changing the graph so to consider only the locations that satisfy the distance constraint. This means that, the monitoring results of a location cannot be *reused* in the monitoring of its neighbours. In this work, we decide to completely reformulate the spatial modalities changing the perspective: instead of searching locations satisfying properties within a certain distance using the shortest path, the satisfaction of a location can be obtained by *using* monitored values obtained from the directly connected locations.

In particular, STREL generalizes SSTL by considering two new operators, named *reach* and *escape*. These new operators simplify the monitoring procedure that can be computed *locally*: the satisfaction of the property at each location depends only on the satisfaction of its neighbours[1]. Furthermore, while SSTL operates on spatio-temporal models that are static (the locations do not change their positions), STREL can handle also mobile/dynamic CPS. We also prove that, for a subclass of models, all the spatial properties expressed with *reach* and *escape*, using euclidean distance, satisfy all the transformed models through rotation, reflection and translation.

Another important feature of our logic considered in this paper is that, following an approach similar to the one considered in [14], we do not rely on a specific domain for interpreting logical properties. Indeed, STL/SSTL semantics can be either *qualitative*, ranging over *boolean values*, or *quantitative*, ranging over *real values*. In this paper, we propose both qualitative and quantitative semantics based on *Constraint Semirings*. These are algebraic structures that consist of a domain and two operations named *choose* and *combine*. Constraint semirings have been shown to be very flexible, expressive and convenient for a wide range of problems, in particular for optimization and solving problems with soft constraints and multiple criteria [5]. The use of semirings allows the definition of a single monitoring procedure that, being parametric with respect to the class of data collected from devices and values produced as results, can be used with different purposes. We then provide an offline monitoring algorithm for STREL, and, to illustrate the main features of the proposed formal framework, we show its application using the monitoring of a simulated *Mobile Ad-hoc sensor NETwork* (MANET) as our running example.

We want to stress that STREL is a flexible framework to formulate properties of CPS: the ability of freely mixing spatial and temporal operators to build complex queries, and to automatically construct monitoring algorithms, marks a neat difference from other related approaches, like the development of ad hoc solutions for specific properties.

The rest of this paper is organized as follows. Section 2 discusses the related work. Section 3 introduces the model we consider to represent the spatio-temporal signals, while section 4 provides the syntax and the semantics of STREL. An offline monitoring algorithm and its implementation is then discussed in section 5. In section 6, we show the logic at work on some examples, in particular we consider a MANET as case study. Section 7 draws our conclusions and discusses future works.

---

[1]We will see in Section 5 that this feature is very important to define distributed and online monitor algorithms.

## 2 RELATED WORK

Monitoring spatial-temporal properties over CPS executions was first proposed in [19] where the author has introduced the notion of spatial-temporal event-based model for CPS. Events are triggered by the execution of actions, by the exchange of messages and by physical changes. Each generated event is labeled with time and space stamps and processed by a monitor. In [20], this concept is further elaborated, developing a spatial-temporal event-based model where the space is represented as a 2D Cartesian coordinate system with location points and location fields.

The approaches described in [19, 20] provide an algorithmic framework enabling a user to develop manually a monitor. However, they do not provide any spatio-temporal logic language enabling the specification and the automatic monitoring generation.

In the field of *collective adaptive systems* [8], other mathematical structures, such as *topological spaces*, *closure spaces*, *quasi-discrete closure spaces* and *finite graphs* [18], have been considered to reason about spatial relations, such as *closeness* and *neighborhood*. Despite these models are suitable for offline and centralised monitoring of model-based simulations, they do not scale well for the runtime monitoring of spatially distributed CPS.

Several logic-based formalisms have been proposed to specify the behavior and the spatial structure of concurrent systems [7] and for reasoning about the topological [4] or directional [6] aspects of the interacting entities. In topological reasoning [4], the spatial objects are sets of points and the relation between them is preserved under translation, scaling and rotation. In directional reasoning, the relation between objects depends on their relative position. These logics are usually highly computationally complex [6] or even undecidable [16].

Monitoring spatial-temporal behaviors has started to receive more attention only recently with SpaTeL [13] and SSTL [18]. The *Spatial-Temporal Logic* (SpaTeL) [13] is the unification of *Signal Temporal Logic* [15] (STL) and *Tree Spatial Superposition Logic* (TSSL) introduced in [2, 3] to classify and detect spatial patterns. TSSL reasons over quad trees, spatial data structures that are constructed by recursively partitioning the space into uniform quadrants. The notion of superposition in TSSL provides a way to describe statistically the distribution of discrete states in a particular partition of the space and the spatial operators corresponding to *zooming in and out in* a particular region of the space. By nesting these operators, it is possible to specify self-similar and fractal-like structures [12] that generally characterize the patterns emerging in nature. The procedure allows one to capture very complex spatial structures, but at the price of a complex formulation of spatial properties, which are in practice only learned from some template image.

Another important work to mention is Voltron [17], an open-source *team-level programming* system for drone's collaborative sensing. Voltron provides special programming constructs to reason about time and space and allows users to express sophisticated collaborative tasks without exposing them to the complexity of concurrent programming, parallel execution, scaling, and failure recovery. The spatial constructs are limited to operate on a set of locations of a given geometry (that the user needs to specify). The system is suitable more for programming than for monitoring. For

example, it does not allow to quantify how much the current CPS execution is close to violate a given requirement.

## 3 SPATIAL MODELS, SIGNALS AND TRACES

In this section, we introduce the model of space we consider, and the type of signals that the logic specifies.

### 3.1 Constraint Semirings

An elegant and general way to represent the result of monitoring is based on *constraint semiring*. This is an algebraic structure that consists of a domain and two operations named *choose* and *combine*. Constraint semirings are subclass of semirings which have been shown to be very flexible, expressive and convenient for a wide range of problems, in particular for optimisation and solving problems with soft constraints and multiple criteria [5], and in model checking [14].

*Definition 3.1 (semiring).* A *constraint semiring* (just *semiring* in the following) is a tuple $\langle A, \oplus, \otimes, \perp, \top \rangle$ composed by a set $A$, two operators $\oplus, \otimes$ and two constants $\perp, \top$ such that:

- $\oplus : 2^A \rightarrow A$ is an associative, commutative, idempotent operator to "choose" among values[2], with $\oplus(\emptyset) = \top$;
- $\otimes : A \times A \rightarrow A$ is an associative, commutative operator to "combine" values;
- $\otimes$ distributes over $\oplus$;
- $\perp \oplus a = a, \top \oplus a = \top, \top \otimes a = a, \perp \otimes a = \perp$ for all $a \in A$;
- $\sqsubseteq$, which is defined as $a \sqsubseteq b$ iff $a \oplus b = b$, provides a complete lattice $\langle A, \sqsubseteq, \perp, \top \rangle$.

We say that a *semiring* $A$ is *idempotent* if and only if for any $a \in A$ $a \oplus a = a \otimes a = a$. Moreover, we say that a *semiring* $A$ is *total* when $\sqsubseteq$ is a *total order*.

With an abuse of notation we sometimes refer to a semiring $\langle A, \oplus, \otimes, \perp, \top \rangle$ with the carrier $A$ and to its components by subscripting them with the carrier, i.e., $\oplus_A, \otimes_A, \perp_A$ and $\top_A$. For the sake of a lighter notation we drop the subscripts if clear from the context.

*Example 3.2.* Typical examples of semirings that we will use in this paper are[3]:

- the Boolean semiring $\langle \{true, false\}, \vee, \wedge, false, true \rangle$;
- the tropical semiring $\langle \mathbb{R}_{\geq 0}^{\infty}, min, +, +\infty, 0 \rangle$;
- the max/min semiring: $\langle \mathbb{R}^{\infty}, max, min, -\infty, +\infty \rangle$ ;
- the integer semiring: $\langle \mathbb{N}^{\infty}, max, min, 0, +\infty \rangle$.

Boolean, max/min and integer semirings are *idempotent* while tropical semiring is not. All the above semirings are *total*.

One of the advantages of *semirings* is that these can be easily composed. For instance, if $A$ and $B$ are two semirings, one can consider the *cartesian product* $\langle A \times B, (\perp_A, \perp_B), (\top_A, \top_B), \oplus, \otimes \rangle$ where operations are applied elementwise.

### 3.2 Spatial model

Space is represented via a graph with edges having a weight from a given semiring.

---

[2]We let $x \oplus y$ to denote $\oplus(\{x, y\})$.
[3]We use $\mathbb{R}^{\infty}$ (resp. $\mathbb{N}^{\infty}$) to denote $\mathbb{R} \cup \{-\infty, +\infty\}$ (resp. $\mathbb{N} \cup \{\infty\}$).

*Definition 3.3.* Let $\langle A, \oplus, \otimes, \perp, \top \rangle$ be a *semiring*, a $A-$*spatial model* $\mathcal{S}$ is a pair $\langle L, \mathbf{W} \rangle$ where:

- $L$ is a set of *locations*, also named *space universe*;
- $\mathbf{W} \subseteq L \times A \times L$ is a *proximity function* associating at most one label $w \in A$ with each distinct pair $\ell_1, \ell_2 \in L$.

We will use $\mathbb{S}_A$ to denote the set of *A-spatial models*, while $\mathbb{S}_A^L$ indicates the set of *A-spatial models* having $L$ as a set of locations. In the following, we will equivalently write $(\ell_1, w, \ell_2) \in \mathbf{W}$ as $\mathbf{W}(\ell_1, \ell_2) = w$ or $\ell_1 \overset{w}{\mapsto} \ell_2$, saying that $\ell_1$ is *next to* $\ell_2$ with weight $w \in A$.

A special class of spatial models are the ones based on *Euclidean spaces*.

*Definition 3.4 (Euclidean spatial model).* Let $L$ be a set of locations, $R \subseteq L \times L$ a (reflexive) relation and $\mu : L \rightarrow \mathbb{R}^2$ a function mapping each location to a point in $\mathbb{R}^2$, we let $\mathcal{E}(L, R, \mu)$ be the $\mathbb{R}^{\infty} \times \mathbb{R}^{\infty}$-spatial model[4] $\langle L, \mathbf{W}^{\mu, R} \rangle$ such that:

$$\mathbf{W}^{\mu, R} = \{(\ell_1, \mu(\ell_1) - \mu(\ell_2), \ell_2) | (\ell_1, \ell_2) \in R\}$$

Note that we label edges with a 2-dimensional vector $w$ describing how to reach $\ell_2$ from $\ell_1$, i.e., $\mu(\ell_1) + w = \mu(\ell_2)$. This obviously allows us to compute the euclidean distance between $\ell_1$ and $\ell_2$ as $\|w\|_2$, but, as we will see, allows us to compute the euclidean distance of any pair of locations connected by any path, not necessarily by a line in the plane.

*Example 3.5 (Mobile Ad hoc sensor NETwork).* A Mobile Ad-hoc sensor NETwork (MANET) is a sensor network that can consist of up ten thousands of mobile devices connected wirelessly. The devices are usually deployed to monitor environmental changes such as pollution, humidity, light and temperature. Each sensor node can be equipped with a sensing transducer, data processor, a radio transceiver and an embedded battery. It can move independently in any direction and change its links to other devices. Two nodes can communicate each other if their Euclidean distance is at most their communication range as depicted in Fig. 1 (right) . Moreover, the nodes can be of different type and their behaviour and communication can depend on their types.

When considering a MANET, we can easily define different proximity functions for the same set of locations, where each location represents a mobile device. Given a set of $n$ reference points in a two-dimensional Euclidean plane, a Voronoi diagram [1] partitions the plane into set of $n$ regions, one per reference point, assigning each point of the plane to the region corresponding to the closest reference point. The dual of the Voronoi diagram is the proximity graph or Delaunay triangulation [9]. In Figure 1 (left), we can see an example of Voronoi diagram (in blue) and proximity graph (in red). The proximity function can then be defined with respect to the Cartesian coordinates, as in Definition 3.4: $\mathbf{W}^{\mu, R}(\ell_i, \ell_j) = \mu(\ell_i) - \mu(\ell_j) = (x_i, y_i) - (x_j, y_j) = (x_i - x_j, y_i - y_j)$, where $(x_i, y_i)$ are the plane coordinates of the location $\ell_i$.

The proximity function can be also equal to a value that depends of other specific characteristics or behaviours of our nodes. For instance, Fig. 1 (right) represents the connectivity graph of MANET. In this case a location $\ell_i$ is next to a location $\ell_j$ if and only if they are within their communication range.

---

[4]$\mathbb{R}^{\infty}$ is the *min/max* semiring considered in Example 3.2.
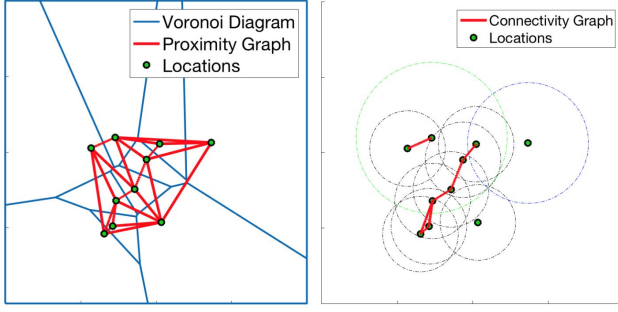
**Figure 1: Proximity graph (left) and Connectivity graph (right)**

Given an $A$-spatial model we can define *routes*.

*Definition 3.6.* Let $\mathcal{S} = \langle L, \mathbf{W} \rangle$, a *route* $\tau$ is an infinite sequence $\ell_0 \ell_1 \cdots \ell_k \cdots$ in $L^\omega$ such that for any $i \geq 0$, $\ell_i \xrightarrow{d} \ell_{i+1}$.

Let $\tau = \ell_0 \ell_1 \cdots \ell_k \cdots$ be a route, $i \in \mathbb{N}$ and $\ell \in L$, we use:

- $\tau[i]$ to denote the $i$-th node $\ell_i$ in $\tau$;
- $\tau[i..]$ to indicate the suffix route $\ell_i \ell_{i+1} \cdots$;
- $\ell \in \tau$ when there exists an index $i$ such that $\tau[i] = \ell$, while we use $\ell \notin \tau$ if this index does not exist;
- $\tau(\ell)$ to denote the first occurrence of $\ell$ in $\rho$:

$$\tau(\ell) = \begin{cases} \min\{i | \tau[i] = \ell\} & \text{if } \ell \in \tau \\ \infty & \text{otherwise} \end{cases}$$

We also use $Routes(\mathcal{S})$ to denote the set of routes in $\mathcal{S}$, while $Routes(\mathcal{S}, \ell)$ denotes the set of routes starting from $\ell \in L$.

We can use routes to define the *distance* among two locations in a *spatial model*. This distance is computed via an appropriate function $f$ that combines all the weights in a route into a value taken from an appropriate semiring $B$.

*Definition 3.7.* Let $\mathcal{S} = \langle L, \mathbf{W} \rangle$ be an $A$-spatial model, $\tau$ a route in $\mathcal{S}$, $\langle B, \oplus_B, \otimes_B, \perp_B, \top_B \rangle$ a *complete* semiring and $f : B \times A \to B$ a *distance monotone function* such that $b \sqsubseteq_B f(b, a)$, or $f(b, a) \sqsubseteq_B b$, for any $a \in A$ and $b \in B$. The distance $d_\tau^f[i]$ up-to index $i$ is:

$$d_\tau^f[i] = \begin{cases} \perp_B & i = 0 \\ f(d_{\tau[1..]}^f[i-1], w) & (i > 0) \text{ and } \tau[0] \xrightarrow{w} \tau[1] \end{cases}$$

Given a locations $\ell \in L$, the distance over $\tau$ up-to $\ell$ is then $d_\tau^f(\ell) = d_\tau^f[\tau(\ell)]$ if $\ell \in \tau$, or $\top_B$ otherwise.

*Example 3.8.* Considering again a MANET, one could be interested in different types of distances, e.g., *counting* the number of *hops*, or distances induced by the weights of the Euclidean space structure.

To count the number of hops, we can simply use the function $hops : \mathbb{N}^\infty \times \mathbb{R}_{\geq 0}^\infty \to \mathbb{N}^\infty$, taking values in the tropical semiring on $\mathbb{N}^\infty$:

$$hops(v, w) = v + 1$$

and in this case $d_\tau^{hops}[i] = i$.

Considering the proximity function $\mathbf{W}^{\mu, R}(\ell_i, \ell_j)$ computed from the Cartesian coordinates, we can use the distance induced by the function $\Delta$ defined as follow

$$\Delta(v, (x, y)) = v + \|(x, y)\|_2,$$

where $(x, y)$ are the coordinates of the vectors returned by $\mathbf{W}^{\mu, R}$ while $v$ is the distance incrementally computed by $\Delta$. It is easy to see that for any route $\tau$ and for any location $\ell \in L$ in $\tau$, the function $d_\tau^\Delta(\ell)$ yields the sum of lengths of the edges in $\mathbb{R}^2$ connecting $\ell$ to $\tau(0)$.

Both the functions *hops* and $\Delta$ are *monotone* and satisfy the constraints:

$$hops(v, w) \sqsubseteq_\mathbb{N} v \qquad \Delta(v, (x, y)) \sqsubseteq_{\mathbb{R}_{\geq 0}^\infty} v$$

The distance between two locations $\ell_1$ and $\ell_2$ is obtained by choosing the distance values along all possible routes starting from $\ell_1$ and ending in $\ell_2$, according to the $\oplus$ operation of the semiring $B$:

$$d_\mathcal{S}(\ell_1, \ell_2) = \oplus\{d_\tau(\ell_2) | \tau \in Routes(\mathcal{S}, \ell_2)\}.$$

*Example 3.9.* Consider again the distance functions defined for a MANETS. For *hops*, we are taking the minimum hop-length over all paths connecting $\ell_1$ and $\ell_2$, resulting in the shortest path distance. In the Euclidean case, the function $\Delta$ returns the same result along any path, which will also be our distance, due to idempotence of $\oplus$.

## 3.3 Spatio-Temporal Signals

*Definition 3.10.* A *signal domain* is a tuple $\langle D, \oplus, \otimes, \odot, \top, \perp \rangle$ where:

- $\langle D, \oplus, \otimes, \top, \perp \rangle$, is an *idempotent semiring*;
- $\odot : D \to D$, is a *negation function* such that:
  - $\odot \top = \perp$;
  - $\odot \perp = \top$;
  - $\odot(v_1 \oplus v_2) = (\odot v_1) \otimes (\odot v_2)$
  - $\odot(v_1 \otimes v_2) = (\odot v_1) \oplus (\odot v_2)$
  - for any $v \in D$, $\odot(\odot v) = v$.

In this paper, we will consider two *signal domains*:

- Boolean signal domain $\langle \{\top, \perp\}, \vee, \wedge, \neg \rangle$ for qualitative monitoring;
- Max/min signal domain $\langle \mathbb{R}^\infty, \max, \min, - \rangle$ for quantitative monitoring.

For signal domains we will use the same notation and notational conventions introduced for semirings.

*Definition 3.11.* Let $\mathbb{T} = [0, T]$ a time domain and $\langle D, \oplus, \otimes, \odot, \top, \perp \rangle$ a *signal domain*, a *temporal $D$-signal* $v$ is a function $v : \mathbb{T} \to D$. Consider a finite sequence:

$$\tilde{v} = [(t_0, d_0), \ldots, (t_n, d_n)]$$

such that, for any $i \in \{0, \ldots, n\}$, $t_i < t_{i+1}$ and $d_i \in D$. Usually, $t_0 = 0$. We let $\tilde{v}$ denote a *piecewise constant temporal $D$-signal* in $\mathbb{T} = [0, T]$, that is

$$\tilde{v}(t) = \begin{cases} \perp & \text{for } t < t_0, \\ d_i & \text{for } t_i \leq t < t_{i+1}, \\ d_n & \text{for } t_n \leq T; \end{cases}$$

Given a *piecewise constant temporal signal* $\tilde{v} = [(t_0, d_0), \ldots, (t_n, d_n)]$ we will use $\mathcal{T}(\tilde{v})$ to denote the set $\{t_0, \ldots, t_n\}$ of *time steps* in $\tilde{v}$; $start(\tilde{v})$ to denote $t_0$; while we will say that $\tilde{v}$ is *minimal* if and only if for any $i$, $d_i \neq d_{i+1}$. We will also let $\tilde{v}[t = d]$ to denote the signal obtained from $\tilde{v}$ by adding the element $(t, d)$. Finally, if $v_1$ and $v_2$ are two $D$-temporal signals, and $op : D \times D \to D$, $v_1 \ op \ v_2$ denotes

the signal associating with each time $t$ the value $v_1(t)$ $op$ $v_2(t)$. Similarly, if $op : D_1 \rightarrow D_2$, $op$ $v_1$ denotes the $D_2$−signal associating with $t$ the value $op$ $v_1(t)$.

*Definition 3.12.* Let $L$ be a *space universe*, and $\langle D, \oplus, \otimes, \odot, \top, \bot \rangle$ a signal domain. A *spatial D-signal* is a function $\mathbf{s} : L \rightarrow D$.

*Definition 3.13 (Spatio-temporal D-signal).* Let $L$ be a space universe, $\mathbb{T} = [0, T]$ a time domain, and $\langle D, \oplus, \otimes, \odot, \top, \bot \rangle$ a signal domain, a spatio-temporal D-signal is a function

$$\sigma : L \rightarrow \mathbb{T} \rightarrow D$$

such that $\sigma(\ell) = v$ is a temporal signal that returns a value $v(t) \in D$ for each time $t \in \mathbb{T}$. We say that $\sigma$ is *piecewise constant* when for any $\ell$, $\sigma(\ell)$ is a *piecewise constant temporal signal*. Piecewise constants *spatio-temporal* signal are denoted by $\tilde{\sigma}$.

Given a spatio-temporal signal $\sigma$, we will use $\sigma@t$ to denote the *spatial signal* at time $t$, i.e. the signal $\mathbf{s}$ such that $\mathbf{s}(\ell) = \sigma(\ell)(t)$, for any $\ell \in L$. Different kinds of signals can be considered while the signal domain $D$ is changed. Signals with $D = \{true, false\}$ are called *boolean signals*; with $D = \mathbb{R}^\infty$ are called real-valued or *quantitative signals*.

*Definition 3.14 (D-Trace).* Let $L$ be a space universe, a *spatio-temporal D-trace* is a function

$$\vec{x} : L \rightarrow \mathbb{T} \rightarrow D^n$$

such that for any $\ell \in L$ yields a vector of temporal signals $\vec{x}(\ell) = (v_1, \ldots, v_n)$. In the rest of the paper we will use $\vec{x}(\ell, t)$ to denote $\vec{x}(\ell)(t)$.

We plan to work with spatial models that can dynamically change their configurations. For this reason, we need to define a function that returns the spatial configuration at each time.

*Definition 3.15 (Location service).* Let $L$ be a spatial universe, a *location service* is a function $\lambda : \mathbb{T} \rightarrow \mathbb{S}_A^L$ associating each element in the time domain $\mathbb{T}$ with a spatial model $\mathbb{S}_A^L$ that describes the spatial configuration of locations.

*Example 3.16.* Let us considering a MANET with a proximity graph. A $\mathbb{R}^\infty$−spatio temporal signal $\sigma : L \rightarrow \mathbb{T} \rightarrow \mathbb{R}^\infty$ associates a temporal signal $\sigma(i) = v$ of real-values at each location $\ell \in L = \{\ell_1, \ldots, \ell_7\}$; $\sigma@t$ instead corresponds to the spatial signal at time $t$, i.e. it is a function that returns a value $\sigma(\ell_i)(t)$ for each location $\ell_i$ at time t. We can see the use of the location service in the figure below. The plot shows two different spatial configurations of the model for time $t_1$ and $t_2$. We can see that locations $\ell_1$ and $\ell_2$ change their position, this changes also the Voronoi diagram and the proximity graph. We have then two different proximity functions on the same space universe $L$, i.e. $\lambda(t_1) = \langle L, \mathbf{W}_1 \rangle$, $\lambda(t_2) = \langle L, \mathbf{W}_2 \rangle$.
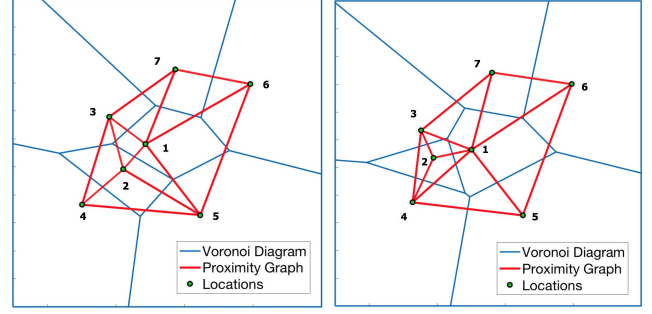


**Figure 2: Two snapshots of a spatial model with 7 locations $\ell_1, \ldots, \ell_7$ that move in a 2D Euclidian space. The plane is partitioned using a Voronoi Diagram (blue). In red we have the proximity graph.**

# 4 SPATIO-TEMPORAL REACH AND ESCAPE LOGIC

In this section, we present the *Spatio-Temporal Reach and Escape Logic* (STREL), an extension of the *Signal Temporal Logic*. We define the syntax and the semantics of STREL, describing in detail the spatial operators and their expressiveness.

## 4.1 Syntax

The syntax of STREL is given by

$$\varphi := \mu \mid \neg\varphi \mid \varphi_1 \wedge \varphi_2 \mid \varphi_1 \, U_{[t_1, t_2]} \, \varphi_2 \mid \varphi_1 \, S_{[t_1, t_2]} \, \varphi_2 \mid \varphi_1 \, \mathcal{R}_d^f \, \varphi_2 \mid \mathcal{E}_d^f \, \varphi$$

where $\mu$ is an *atomic predicate* (*AP*), *negation* $\neg$ and *conjunction* $\wedge$ are the standard Boolean connectives, $U_{[t_1, t_2]}$ and $S_{[t_1, t_2]}$ are the *Until* and the *Since* temporal modalities, with $[t_1, t_2]$ a real positive closed interval. These are the standard temporal operators of STL, and we refer the reader to [11, 15] for more details. The spatial modalities are the *reachability* $\mathcal{R}_d^f$ and the *escape* $\mathcal{E}_d^f$ operators, with $f$ a *Distance Function*, (we call *DF* their collection), described in the previous section, and $d$ a *Distance Predicate* (from a set *DP* of predicates), e.g., inequalities[5]. The exact meaning of $f$ and $d$ depends on specific interpretation functions. This because, the monitored value associated with a formula $\varphi$ depends on the considered domain. We impose that any $d$ occurring in a operator $\mathcal{R}_d^f$ is $\sqsubseteq -closed$, i.e., if $x$ satisfies $d$ and $y \sqsubseteq x$ then $y$ satisfies $d$. This because predicate $d$ represents an *upper bound* on a distance.

The reachability operator $\phi_1 \mathcal{R}_d^f \phi_2$ describes the behavior of reaching a location satisfying property $\phi_2$ passing only through locations that satisfy $\phi_1$, through nodes whose distance from the initial location satisfy the predicate $d$. The escape operator $\mathcal{E}_d^f \phi$, instead, describes the possibility of escaping from a certain region passing only through locations that satisfy $\phi$, via a route with distance satisfying the predicate $d$. Differently from $\mathcal{R}$, in $\mathcal{E}_d^f$ the predicate $d$ represents a *lower bound*. For this reason we assume that any interpretation of $d$ is $\sqsupseteq -closed$, i.e., if $x$ satisfies $d$ and $x \sqsubseteq y$ then $y$ satisfies $d$[6].

---

[5]With an abuse of notation, we will denote by $\neg d$ the predicate that complements $d$.
[6]$d$ is $\sqsubseteq -closed$ if and only if $\neg d$ is $\sqsupseteq -closed$.

As customary, we can derive the *disjunction* operator $\vee$ and the future *eventually* $F_{[t_1,t_2]}$ and *always* $G_{[t_1,t_2]}$ operators from the until temporal modality, and the corresponding past variants from the since temporal modality, see [15] for details. We can define also other three derived spatial operators: the *somewhere* and the *everywhere* that describe behaviors of some or of all locations at a certain distance from a specific point, and the *surround* that expresses the topological notion of being surrounded by a $\phi_2$-region, while being in a $\phi_1$-region, with additional metric constraints. A more thorough discussion of the spatial operators will be given after introducing the semantics.

## 4.2 Semantics

The semantics of STREL is evaluated point-wise at each time and each location. We stress that each STREL formula $\varphi$ abstracts from the specific domain used to express the satisfaction value of $\varphi$ as well as there is not explicit reference to the semiring used in the spatial model to express *weights* associated with edges. These, of course, are needed to define the semantics. In the following, we assume that $D_1$ is the domain of the spatio-temporal traces, while $D_2$ is the semiring where the logic is evaluated. Furthermore, $A$ is the semiring of weights, and $B$ is the semiring in which distance functions take values. To define the semantics, we also need three auxiliary functions. The *signal interpretation function* $\iota : AP \times D_1^n \to D_2$ permits to translate the input trace in a different $D_2$-spatio temporal signal, for each atomic proposition in $AP$, which will be the input of the monitoring procedure. The function $\gamma : DF \to (B \times A \to B)$ is used to interpret function symbols as proper distance functions, while $\delta : DB \to (B \to \{true, false\})$ maps distance predicate symbols into proper predicates.

*Definition 4.1 (Semantics).* Let $A$ and B be two semirings, and $D_1$ and $D_2$ two signal domains. Let $L$ be a *space universe*, $\vec{x}$ be a *spatio-temporal $D_1$-trace* for $L$ and $\lambda : \mathbb{T} \to \mathbb{S}_L^A$ the *location service* associating an A-spatial model $\mathbb{S}_A^L$ at each time in $\mathbb{T}$. Let $\iota$, $\gamma$, and $\delta$ be the functions introduced above. The $D_2$-monitoring function $\mathbf{m}$ of $\vec{x}$ is recursively defined in Table 1.

Given a formula $\phi$, the function $\mathbf{m}(\lambda, \vec{x}, \phi, t, \ell)$ corresponds to the evaluation of the formula at time $t$ in the location $\ell$. The choice of $B, D_2, \iota, \gamma$ and $\delta$ produces different types of semantics. As described in Section 3, we consider two signal domains: $\mathbb{B}$ and $\mathbb{R}^\infty$, giving rise to qualitative and quantitative monitoring, correspond respectively to a Boolean answer value and real satisfaction value. We describe the semantics for the Boolean signal domain ($D_2 = \langle \{\top, \bot\}, \vee, \wedge, \neg \rangle$). We say that $(\lambda, x(\ell, t))$ satisfies a formula $\phi$ if $\mathbf{m}(\lambda, \vec{x}, \phi, t, \ell) = \top$. The procedure will be exactly the same for different choices of the formula evaluation domain, just operators have to be interpreted according to the chosen semirings and signal domains. We use the following example as the system on which we specify our properties, in particular we will use the graph in Figure 3 to describe the spatial operators.

*Example 4.2 (ZigBee protocol).* In Fig. 3, the graph represents a MANET. In particular, we consider the nodes with three different roles such as the ones implemented in the ZigBee protocol: *coordinator*, *router* and *EndDevice*. The Coordinator node (*coord*),

represented in green color in the graph, is unique in each network and is responsible to initialize the network. After the initialisation, the coordinator behaves as a router. The Router node (*router*), represented in red color in the graph, acts as a intermediate router, passing on data from other devices. The EndDevice node (*end_dev*), represented in blue, can communicate only with a parent node (either the Coordinator or a Router) and it is unable to relay data from other devices. Nodes move in space and the figure corresponds to the spatial configuration at a fixed time $t$. As trace and location service, let us consider a $\mathbb{R}^\infty$-spatial model as the proximity graph presented in Example 3.5 and a $\mathbb{B}$-trace over this graph $\vec{x} : L \to \mathbb{T} \to \mathbb{B}^3$ denoting the kind of node, i.e. $\vec{x}(\ell, t) = (\top, \bot, \bot)$ if $\ell$ is a coordinator, $\vec{x}(\ell, t) = (\bot, \top, \bot)$ if $\ell$ is a router, and $\vec{x}(\ell, t) = (\bot, \bot, \top)$ if $\ell$ is an end node.

**Atomic Proposition.** $\mathbf{m}(\lambda, \vec{x}, \mu, t, \ell) = \iota(\mu, \vec{x}(t, \ell))$. Different types of atomic propositions and signal interpretations are admissible. We can simply consider a finite set $\{p_1, \ldots, p_n\} = AP$ and an interpretation function $\iota(p_i, \vec{x}(\ell, t)) = \top$ iff $x_i(\ell, t) = \top$. E.g., in Fig. 3, we can consider atomic propositions describing the type of node, i.e., the boolean propositions $\{coord, router, end\_dev\}$ are true if the node is of the corresponding type. In case of real valued signals and of a quantitative interpretation of the logic ($D_2$ being in this case the real valued max/min semiring), we can consider inequalities $\mu = (g(\vec{x}) \geq 0)$ for some real function $g$ and define $\iota(\mu, t, \vec{\ell}) = g(\vec{x}, t)$.

**Negation.** $\mathbf{m}(\lambda, \vec{x}, \neg\varphi, t, \ell) = \neg\mathbf{m}(\lambda, \vec{x}, \varphi, t, \ell)$

**Conjunction.** $\mathbf{m}(\lambda, \vec{x}, \varphi_1 \wedge \varphi_2, t, \ell) = \mathbf{m}(\lambda, \vec{x}, \varphi_1, t, \ell) \wedge \mathbf{m}(\lambda, \vec{x}, \varphi_2, t, \ell)$

**Until.** $\mathbf{m}(\lambda, \vec{x}, \varphi_1 U_{[t_1,t_2]}\varphi_2, t, \ell) = \bigvee_{t' \in t+[t_1,t_2]}(\mathbf{m}(\lambda, \vec{x}, \varphi_2, t', \ell) \wedge \bigwedge_{t'' \in [t,t']} \mathbf{m}(\lambda, \vec{x}, \varphi_1, t'', \ell))$. As customary, $(\lambda, x(\ell, t))$ satisfies $\varphi_1 U_{[t_1,t_2]}\varphi_2$ iff it satisfies $\varphi_1$ from $t$ until, in a time between $t_1$ and $t_2$ time units in the future, $\varphi_2$ becomes true. Note how the temporal operators are evaluated in each location separately.

**Since.** $\mathbf{m}(\lambda, \vec{x}, \varphi_1 \ S_{[t_1,t_2]} \ \varphi_2, t, \ell) = \bigvee_{t' \in t-[-t_2,-t_1]} \left(\mathbf{m}(\lambda, \vec{x}, \varphi_2, t', \ell) \wedge \bigwedge_{t'' \in [t',t]} \mathbf{m}(\lambda, \vec{x}, \varphi_1, t'', \ell)\right)$. $(\lambda, x(\ell, t))$ satisfies $\varphi_1 \ S_{[t_1,t_2]} \ \varphi_2$ iff it satisfies $\varphi_1$ from now since, in a time between $t_1$ and $t_2$ time units in the past, $\varphi_2$ was true.

Except for the interpretation function, the semantics of the boolean and the temporal operators is directly derived from and coincident with that of STL (qualitative for Boolean signal domain and quantitative for an $\mathbb{R}^\infty$ signal domain), see [11] for details.

**Reachability.** $\mathbf{m}(\lambda, \vec{x}, \varphi_1 \ \mathcal{R}_d^f \ \varphi_2, t, \ell) = \bigvee_{\tau \in Routes(\lambda(t), \ell)} \bigvee_{\ell' \in \tau : \delta(d)\left(d_\tau^{\gamma(f)}(\ell')\right)}(\mathbf{m}(\lambda, \vec{x}, \varphi_2, t, \tau(\ell')) \wedge \bigwedge_{j < \tau(\ell')} \mathbf{m}(\lambda, \vec{x}, \varphi_1, t, \tau[j]))$

$(\lambda, x(\ell, t))$ satisfies $\varphi_1 \ \mathcal{R}_d^f \ \varphi_2$ iff it satisfies $\varphi_2$ in a location $\ell'$ reachable from $\ell$ through a route $\tau$, with a length $d_\tau^{\gamma(f)}(\ell')$ satisfying the predicate $\delta(d)$, and such that $\tau[0] = \ell$ and all its elements with index less than $\tau(\ell')$ satisfy $\varphi_1$. In Figure 3, we report an example of reachability property, considering f as the *hops* function described in Example 3.9. In the graph, the location $\ell_6$ (meaning the trajectory $\vec{x}$ at time t in position $\ell_6$) satisfies *end_dev* $\mathcal{R}_{m \leq 1}^{hops}$ *router*, with distance predicate $d = m \leq 1$ being true if the distance is less than or equal to 1 units. Indeed, there exists a route $\tau = \ell_6\ell_5$ such that $d_\tau^{hops}[1] = 1$, where $\tau[0] = \ell_6$, $\tau[1] = \ell_5$, $\tau[1]$ satisfies the red property (it is a router) and all the other elements of the route satisfy the blue property (they are end-devices). Instead, for example,

$$\mathbf{m}(\lambda, \vec{x}, \mu, t, \ell) = \iota(\mu, \vec{x}(t, \ell))$$

$$\mathbf{m}(\lambda, \vec{x}, \neg\varphi, t, \ell) = \odot_{D_2} \mathbf{m}(\lambda, \vec{x}, \varphi, t, \ell)$$

$$\mathbf{m}(\lambda, \vec{x}, \varphi_1 \wedge \varphi_2, t, \ell) = \mathbf{m}(\lambda, \vec{x}, \varphi_1, t, \ell) \otimes_{D_2} \mathbf{m}(\lambda, \vec{x}, \varphi_2, t, \ell)$$

$$\mathbf{m}(\lambda, \vec{x}, \varphi_1 \, U_{[t_1, t_2]} \, \varphi_2, t, \ell) = \bigoplus_{D_2 t' \in [t+t_1, t+t_2]} \left( \mathbf{m}(\lambda, \vec{x}, \varphi_2, t', \ell) \otimes_{D_2} \bigotimes_{D_2 t'' \in [t, t']} \mathbf{m}(\lambda, \vec{x}, \varphi_1, t'', \ell) \right)$$

$$\mathbf{m}(\lambda, \vec{x}, \varphi_1 \, S_{[t_1, t_2]} \, \varphi_2, t, \ell) = \bigoplus_{D_2 t' \in [t-t_2, t-t_1]} \left( \mathbf{m}(\lambda, \vec{x}, \varphi_2, t', \ell) \otimes_{D_2} \bigotimes_{D_2 t'' \in [t', t]} \mathbf{m}(\lambda, \vec{x}, \varphi_1, t'', \ell) \right)$$

$$\mathbf{m}(\lambda, \vec{x}, \varphi_1 \, \mathcal{R}_d^f \, \varphi_2, t, \ell) = \bigoplus_{D_2 \tau \in Routes(\lambda(t), \ell)} \bigoplus_{D_2 \ell' \in \tau : \delta(d)} \left( d_\tau^{\gamma(f)}(\ell') \right) \left( \mathbf{m}(\lambda, \vec{x}, \varphi_2, t, \ell') \otimes_{D_2} \bigotimes_{D_2 j < \tau(\ell')} \mathbf{m}(\lambda, \vec{x}, \varphi_1, t, \tau[j]) \right)$$

$$\mathbf{m}(\lambda, \vec{x}, \mathcal{E}_d^f \, \varphi, t, \ell) = \bigoplus_{D_2 \tau \in Routes(\lambda(t), \ell)} \bigoplus_{D_2 \ell' \in \tau : \delta(d)} \left( d_{\lambda(t)}^{\gamma(f)}(\ell, \ell') \right) \bigotimes_{D_2 i \leq \tau(\ell')} \mathbf{m}(\lambda, \vec{x}, \varphi, t, \tau[i])$$

**Table 1: Monitoring function.**

the location $\ell_8$ does not satisfy the property because it does not satisfies the blue (end-device) property.

**Escape.** $\mathbf{m}(\lambda, \vec{x}, \mathcal{E}_d^f \, \varphi, t, \ell) = \bigvee_{\tau \in Routes(\lambda(t), \ell)} \bigvee_{\ell' \in \tau : \delta(d)} \left( d_{\lambda(t)}^{\gamma(f)}(\ell, \ell') \right) \bigwedge_{i \leq \tau(\ell')} \mathbf{m}(\lambda, \vec{x}, \varphi, t, \tau[i])$. $(\lambda, x(t, \ell))$ satisfies $\mathcal{E}_d^f \, \varphi$ if and only if there exists a route $\tau$ and a location $\ell' \in \tau$ such that $\tau[0] = \ell$ and $d_S(\tau[0], \ell')$ satisfies the predicate $\delta(d)$, while $\ell'$ and all the elements $\tau[0], \ldots \tau[k-1]$ (with $\tau(\ell') = k$) satisfy $\varphi$. In Fig 3, we report an example of escape property. In the graph, the location $\ell_{10}$ satisfies $\mathcal{E}_{m \geq 2}^{hops} \, \neg end\_dev$. Indeed, there exists a route $\tau = \ell_{10}\ell_7\ell_8$ such that $\tau[0] = \ell_{10}, \tau[2] = \ell_8, d_S^{hops}(\ell_{10}, \ell_1) = 2$ and $\ell_{10}, \ell_7$ and $\ell_8$ do not satisfy the blue property, i.e. they are not end-devices. Note that the route $\ell_{10}\ell_{11}\ell_{16}$ is not a good route to satisfy the property because the distance $d_S^{hops}(\ell_{10}, \ell_{16}) = 1$.



**Figure 3: Example of spatial properties. Reachability:** $end\_dev \, \mathcal{R}_{m \leq 1}^{hops} \, router$. **Escape:** $\mathcal{E}_{m \geq 2}^{hops} \, \neg end\_dev$. **Somewhere:** $\diamondsuit_{m \leq 4}^{hops} coord$. **Everywhere:** $\boxdot_{m \leq 2}^{hops} router$. **Surround:** $(coord \vee router) \, \odot_{m \leq 3}^{hops} \, end\_dev$.

We can also derive other three spatial operators: *somewhere*, *everywhere* and *surround*.

**Somewhere.** $\diamondsuit_d^f \varphi := true \mathcal{R}_d^f \varphi$ is satisfied by $(\lambda, x(t, \ell))$ iff there exists a location that satisfies $\varphi$ reachable from $\ell$ via a route $\tau$ with a distance satisfying the predicate $\delta(d)$. This length is computed via the function $\gamma(f)$. In Fig. 3, all the locations satisfy the property

$\diamondsuit_{m \leq 4}^{hops} coord$ because, for all $\ell_i$, there is always a path $\tau = \ell_i \ldots \ell_{10}$ with a length $d_\tau^{hops}[k] \leq 4$, where $\tau[0] = \ell_i, \tau[k] = \ell_{10}$, and $\ell_{10}$ satisfies the green property, i.e. it is a coordinator node.

**Everywhere.** $\boxdot_d^f \varphi := \neg \diamondsuit_d^f \neg \varphi$ is satisfied by $(\lambda, x(t, \ell))$ iff all the locations reachable from $\ell$ via a path, with length satisfying the predicate $\delta(d)$, satisfy $\varphi$. In Fig. 3, there are no locations that satisfy the property $\boxdot_{m \leq 2}^{hops} router$ because for all the locations $\ell_i$ there is a path $\tau = \ell_i \ell_j$ s.t. $\ell_j$ is not a router.

**Surround.** $\varphi_1 \, \odot_d^f \, \varphi_2 := \varphi_1 \wedge \neg(\varphi_1 \mathcal{R}_d^f \neg(\varphi_1 \vee \varphi_2) \wedge \neg(\mathcal{E}_{\neg d}^f \varphi_1)$ expresses the topological notion of being surrounded by a $\varphi_2$-region, while being in a $\varphi_1$-region, with an additional metric constraint. The operator has been introduced in [8] as a basic operator, while here it is a derived one. The idea is that one cannot escape from a $\varphi_1$-region without passing from a location that satisfies $\varphi_2$ and, in any case, one has to reach a $\varphi_2$-location via a path with a length satisfying the predicate $d$. In Fig. 3, the location $\ell_{10}$ satisfies the property $(coord \vee router) \odot_{\leq 3}^{hops} end\_dev$. In fact, it is coordinator, it cannot reach a location that does not satisfy the the $coord \vee router$ or the $end\_dev$ property via a path with length lesser or equal to 3 and it cannot escape through a path satisfying the $coord \vee router$ property at a distance more than 3.

The operators can be arbitrarily composed to specify complex properties as we will see in Section 6. Furthermore, they can be evaluated both on indirect and on direct graphs.

## 5 MONITORING STREL

In this section, we present a monitoring algorithm that can be used to check if a given signal satisfies or not a STREL property. The proposed algorithm follows an *offline* approach. Indeed, it takes as input the complete spatio-temporal signal together with the property we want to monitor. At the end of this section, we will also briefly discuss a possible alternative approach that can lead to a distributed and *online* monitoring procedure. In this case, the spatio-temporal signal is not known at the beginning, it is discovered while data are collected from the system during its execution.

### 5.1 Offline monitor

Offline monitoring is performed via the function monitor that takes as inputs a location service $\lambda$, a trace $\vec{x}$ and a formula $\phi$ and returns

the *piecewise constant spatio-temporal signal* $\tilde{\sigma}$ representing the monitoring of $\phi$. The function also relies on parametrised with respect to functions $\iota, \delta$ and $\gamma$, used to interpret symbols in formulas, and operators $\oplus_{D_2}, \otimes_{D_2}$ and $\odot_{D_2}$ of *signal domain*, used to represent satisfaction values.

The function monitor is defined by induction on the syntax of the formula[7]. The spatio-temporal signal resulting from the monitoring of atomic proposition $\mu$ is just obtained by applying function $\iota(\mu)$ to the trace $\mathbf{x}$. The spatio-temporal signals associated with $\neg \varphi$ and $\varphi_1 \wedge \varphi_2$ are obtained by applying operators $\odot_{D_2}$ and $\otimes_{D_2}$ to the signals resulting from the monitoring of $\varphi$ and from the monitoring of $\varphi_1$ and $\varphi_2$.

Monitoring of temporal properties, namely $\varphi_1 U_{\leq t} \varphi_2$ and $\varphi_1 S_{\leq t} \varphi_2$, can be done by using the same approach used in [11] and [15]. However, while their monitoring relies on classical boolean and arithmetic operators, here the procedure is parametrised with respect to operators $\oplus_{D_2}$ and $\otimes_{D_2}$ of the considered semiring.

To monitor $\varphi_1 \mathcal{R}_d^f \varphi_2$ first the signals $\mathbf{s}_1$ and $\mathbf{s}_2$ resulting from the monitoring of $\varphi_1$ and $\varphi_2$ are computed. After that, the final result is computed by aggregating the spatial signals $\mathbf{s}_1 @ t$ and $\mathbf{s}_2 @ t$ at each time $t \in \mathcal{T}(\mathbf{s}_1) \cup \mathcal{T}(\mathbf{s}_2)$ with function reach, defined in Algoritm 1. This function also takes as parameters the spatial model $(L, \mathbf{W})$ at time $t$ (obtained from the *location service*), the function $f : B \times A \to B$ used to compute the distances over paths, and the predicate $d$ describing the reachability bound. In function reach, the data structure $r$ is iteratively computed. This data structure associates each location $\ell$ with a set of triples $(\ell', v, w)$. Intuitively, $(\ell', v, w)$ is in $r[\ell]$ after $i$ iterations if and only if: $\ell$ can reach $\ell'$ with at most $i$-steps with a distance at least $w$ ($w$ satisfying $d$) and a monitored value $v$. At the beginning $r[\ell]$ is initialised to $\{(\ell, \mathbf{s}_2(\ell), 0)\}$. Moreover, at each iteration, the values in $r[\ell]$ are updated by considering the elements in $r[\ell']$, for any $\ell'$ next to $\ell$. The loop continues until a fix point is reached. Note that, termination of the algorithm is guaranteed by the fact that $D_2$ is an *idempotent semiring* and from the fact that, for any $(\ell_1, v_1, w_1), (\ell_2, v_2, w_2) \in r[\ell]$, if $\ell_1 = \ell_2$ and $v_1 = v_2$ then $w_1 = w_2$. The result spatial signal associates each location $\ell$ with the value $\bigoplus_{D_2}(\{v | (\ell', v, w) \in r[\ell]\})$.

Monitoring algorithm for $\mathcal{E}_d^f \varphi$ is reported in Algorithm 2, where function escape is defined. Given a space model at time $t$, a distance function $f$, a distance predicate $d$ and a spatial signal, it computes the spatial signal representing the monitoring value of $\mathcal{E}_d^f \varphi$ at time $t$. Function escape iteratively computes the data structure obtained by $e$ that associates each location $\ell$ with a set of triples of the form $(\ell', v, w)$ representing the fact that $\ell$ can *escape* in $\ell'$ with a distance $w$ and a total value $v$. At each iteration, these values are updated by considering the values in the neighbours in each location. Similarly to function reach, this computation continues until a fixpoint is reached. After that, the monitored value associated with each location $\ell$ is computed as $\bigoplus_{D_2}(\{v | (\ell', v, w) \in e[\ell] \wedge d(w)\})$.

**Remark.** The offline monitoring iteratively computes the monitor value at a location by considering the values of monitoring in the previous iteration. This approach easily enables the definition of a

---

[7]This definition is straightforward and, for the sake of readability, we only report it in Appendix, available in the extend version of this article at https://github.com/Quanticol/strel

---

**Algorithm 1** Function reach

1: inputs: $(L, \mathbf{W})$, $f : B \times A \to B$, $d : B \to \{true, false\}$, $\mathbf{s}_1, \mathbf{s}_2$
2: $\forall \ell \in L.r[\ell] = \{(\ell, \mathbf{s}_2(\ell), 0)\}$
3: $stable = false$
4: **while** $\neg stable$ **do**
5: 　　$stable = true$
6: 　　$r' = r$
7: 　　**for all** $\ell_1 \in L$ **do**
8: 　　　　**for all** $\ell_2 : \ell_1 \overset{w}{\mapsto} \ell_2$ **do**
9: 　　　　　　$N = \{(\ell, v \otimes_{D_2} \mathbf{s}_1(\ell_1), f(w', w)) | (\ell, v, w') \in r[\ell_2] \wedge d(f(w', w))\}$
10: 　　　　　　**for all** $(\ell, v, w) \in N$ **do**
11: 　　　　　　　　**if** $\exists(\ell, v, w') \in r'[\ell]$ **then**
12: 　　　　　　　　　　$r'[\ell] = r'[\ell] - \{(\ell, v, w')\} \cup \{(\ell, v, w \oplus_B w')\}$
13: 　　　　　　　　**else**
14: 　　　　　　　　　　$r'[\ell] = r'[\ell] \cup \{(\ell, v, w)\}$
15: 　　　　　　　　**end if**
16: 　　　　　　**end for**
17: 　　　　**end for**
18: 　　　　**if** $r'[\ell] \neq r[\ell]$ **then**
19: 　　　　　　$stable = false$
20: 　　　　**end if**
21: 　　**end for**
22: 　　$r = r'$
23: **end while**
24: $\mathbf{s} = []$
25: **for all** $\ell \in L$ **do**
26: 　　$\mathbf{s}(\ell) = \bigoplus_{D_2}(\{v | (\ell', v, w) \in r[\ell]\})$
27: **end for**
28: **return** $\mathbf{s}$

---

parallel monitoring algorithm. Indeed, both the functions defined in Algorithm 1 and Algorithm 2 can be parallel executed for each location. The different monitoring instances must communicate to exchange the values computed at iteration $i$. Another possible improvement of this algorithm is based on an *online* computation of the monitoring. Following an approach similar to the one considered in [10], each location can identify its monitoring value by using only partial informations. Early termination of the monitor procedure is then possible when the satisfaction or violation of a property is found.

## 6 EXAMPLES

In this section we present some example of the expressibility and potentiality of STREL.

### 6.1 ZigBee protocol monitoring

Given a MANET with a ZigBee protocol (Example 4.2), we consider as spatial models both its proximity and connectivity graphs, computed with respect to the Cartesian coordinates. The Nodes have three kinds of roles: *coordinator*, *router* and *EndDevice*, as described in Example 4.2. Moreover, each device is also equipped with a sensor to monitor its battery level ($X_B$), the humidity ($X_H$) and the pollution ($X_H$) in its position. The semiring is the union between the *max/min* semiring $\mathbb{R}^\infty$ (for the proximity graph) and the *integer*

**Algorithm 2** Function escape

1: inputs: $(L, \mathbf{W}), f : B \times A \rightarrow B, d : B \rightarrow \{true, false\}, \mathbf{s}_1$
2: $\forall \ell \in L.e[\ell] = \{(\ell, \mathbf{s}_1(\ell), 0)\}$
3: $stable = false$
4: **while** $\neg stable$ **do**
5:     $stable = true$
6:     $e' = e$
7:     **for all** $\ell_1 \in L$ **do**
8:         **for all** $\ell_2 : \ell_1 \overset{w}{\mapsto} \ell_2$ **do**
9:             $N = \{(\ell, v \otimes_{D_2} \mathbf{s}_1(\ell_1), f(w', w)) | (\ell, v, w') \in r[\ell_2]\}$
10:         **for all** $(\ell, v, w) \in N$ **do**
11:             **if** $\exists (\ell, v', w') \in e'[\ell_1]$ **then**
12:                 $e'[\ell_1] = e'[\ell_1] - \{(\ell, v', w')\} \cup \{(\ell, v \oplus_{D_2} v', w \oplus_B w')\}$
13:             **else**
14:                 $e'[\ell] = e'[\ell] \cup \{(\ell, v, w)\}$
15:             **end if**
16:         **end for**
17:         **end for**
18:     **if** $e'[\ell] \neq e[\ell]$ **then**
19:         $stable = false$
20:     **end if**
21:     **end for**
22:     $e = e'$
23: **end while**
24: $\mathbf{s} = []$
25: **for all** $\ell \in L$ **do**
26:     $\mathbf{s}(\ell) = \bigoplus_{D_2}(\{v | (\ell', v, w) \in e[\ell] \wedge d(w)\})$
27: **end for**
28: **return s**

semiring $\mathbb{N}^\infty$ (for the connectivity graph). We will use also two types of distances: *hops* and the $\Delta$ distances described in Example 3.9. As in the Example 4.2, atomic propositions $\{coord, router, end\_dev\}$ describe the type of nodes. We also consider inequalities on the values that are read from sensors, plus special propositions $@_\ell$ which encode the address of a specific location, i.e. they are true only in the location $\ell$.

In the following, we describe several properties of these Zig-Bee MANET networks that are easily captured by STREL logic, to exemplify its expressive power.

A class of properties naturally encoded in STREL related to the connectivity of the network. First, we can be interested to know if a node is properly connected, meaning that it can reach the coordinator through a path of routers:

$$\phi_{connect} = end\_dev\,\mathcal{R}_{m \leq 1}^{hops}(router\,\mathcal{R}_{m < \infty}^{hops}coord) \qquad (1)$$

The meaning of this property is that an end node reaches in a step a node which is a router and that is connected to the coordinator via a path of routers.

We may also want to know if there is a path to the router which is reliable in terms of battery levels, for instance such that all routers

have a battery level above 30%:

$$\phi_{reliable\_router} = ((X_B > 30\%) \wedge router)\mathcal{R}_{m < \infty}^{hops}coord$$

$$\phi_{reliable\_connect} = end\_dev\,\mathcal{R}_{m \leq 1}^{hops}(\phi_{reliable\_router}) \qquad (2)$$

The properties focus on spatial connectivity at a fixed time. We can add also temporal requirements, for instance asking that a broken connection is restored within $h$ time units:

$$\phi_{connect\_restore} = G(\neg\phi_{connect} \rightarrow F_{[0,h]}\phi_{connect}) \qquad (3)$$

Another class of properties of interest is the acyclicity of transmissions. To this end, we need to force the connectivity graph to be direct, with edges pointing in the direction of the coordinator (i.e. transmission reduces the distance from the coordinator). With STREL, we can easily detect the absence of a cycle locally, i.e. for a fixed location $\ell$. This is captured by $\phi_{acyclic}^\ell = \neg\phi_{cycle}^\ell$, where

$$\phi_{acyclic}^\ell = @_\ell \mathcal{R}_{m \leq 1}^{hops}(\neg@_\ell \wedge \diamond^{hops}@_\ell) \qquad (4)$$

In order to characterize the whole network as acyclic, we need to take the conjunction of the previous formulae for all locations (or at least for routers, enforcing end devices to be connected only with routers). This is necessary as STREL is interpreted locally, on each location, and this forbids us to express properties of the whole network with location unaware formulae. This is a price for an efficient monitoring, as global properties of networks require more expressive and computationally expensive logics. However, we can use the parametrization of STREL and the property of a Voronoi diagram to specify the global connection or the acyclicity of the graph. Indeed, the proximity graph connects always all the locations of the system, then the property $\boxdot^\Delta\phi$, verified on the proximity graph, holds iff $\phi$ holds in all the location of the system.

Up to now we have presented qualitative properties, depending on the type of node. If we express properties of sensor measurements, we can also consider a quantitative semantics, returning a measure of robustness of (dis)satisfaction. As an example, we can monitor (5) if in each location an high value of pollution eventually implies, within $T$ time units, an high value of humidity, or (6) in which locations it is possible to find a 'safe' route, where both the humidity and the pollution are below a certain threshold. We can also check (7) if a location, which is not safe, is at distance at most 5 from a location which is safe. Finally (8), we can check if a target device (identified by $X_S = 1$) is reachable from all the locations in less than 10 hops.

$$\phi_{PH} = (X_P > 150) \Rightarrow F_{[0,T]}(X_H > 100) \qquad (5)$$

$$\phi_{Safe} = G_{[0,T]}\mathcal{E}_{m \geq k}^\Delta (X_H < 90) \wedge (X_P < 150) \qquad (6)$$

$$\phi_{some} = \diamond_{m \leq 5}^\Delta \phi_{Safe} \qquad (7)$$

$$\phi_{target} = \boxdot^{hops}\diamond_{m < 10}^{hops} (X_S = 1) \qquad (8)$$

## 6.2 Invariance properties of the Euclidean spatial model

The properties we consider with respect to the Euclidean spatial model are typically local and depend on the relative distance and position among nodes in the plane. As such, they should be invariant with respect to change of coordinates, i.e. with respect to isometric transformations of the plane. This class of transformations includes

translations, rotations, and reflections, and can be described by matrix multiplications of the form

$$\begin{bmatrix} x'_\ell \\ y'_\ell \\ 1 \end{bmatrix} = \begin{bmatrix} \beta \cos(\alpha) & -\beta \sin(\alpha) & \beta t_x \\ \gamma \sin(\alpha) & \gamma \cos(\alpha) & \gamma t_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_\ell \\ y_\ell \\ 1 \end{bmatrix}$$

Invariance of satisfaction of spatial properties holds in STREL logic, for the Euclidean space model of Definition 3.4. Consider more specifically an Euclidean space model $\mathcal{E}(L, \mu, R) = \langle L, \mathbf{W}^{\mu, R} \rangle$ and $\mathcal{E}(L, \mu', R) = \langle L, \mathbf{W}^{\mu', R} \rangle$, obtained by applying an isometric transformation $A$: $\mu'(\ell) = A(\mu(\ell))$. For invariance to hold, we need to further require that distance predicates, used in spatial operators, are invariant for isometric transformations. More specifically, for any isometry $A$, we require a distance predicate $d$ on the semiring $\mathbb{R}^\infty \times \mathbb{R}^\infty$ to satisfy $d((x, y)) = d(A((x, y)))$. This is the case for the norm-based predicates used in the examples, of the form $d((x, y)) = \|(x, y)\|_2 \leq r$.

Notice that, the path structure is preserved (the edges given by $R$ is the same), and the truth of isometry-invariant distance predicates along paths in $\mathcal{E}(L, \mu, R)$ and $\mathcal{E}(L, \mu', R)$ is also the same. This straightforwardly implies that the truth value of spatial operators will be unchanged by isometry.

PROPOSITION 6.1 (EQUISATISFIABILITY UNDER ISOMETRY). *Let $\mathcal{E}(L, \mu, R) = \langle L, \mathbf{W}^{\mu, R} \rangle$ be an euclidean spatial model and $\mathcal{E}(L, \mu', R) = \langle L, \mathbf{W}^{\mu', R} \rangle$ an isometric transformation of the former. Consider a spatial formula $\varphi_1 \mathcal{R}_d^f \varphi_2$ or $\mathcal{E}_d^f \varphi_1$, where $d$ is an isometry preserving predicate. Assume $\mathbf{m}(\lambda, \vec{x}, \varphi_j, t, \ell) = \mathbf{m}'(\lambda, \vec{x}, \varphi_j, t, \ell)$, $j = 1, 2$, where $\mathbf{m}$ and $\mathbf{m}'$ are the monitoring functions for the two spatial models. Then it holds that $\mathbf{m}(\lambda, \vec{x}, \varphi_1 \mathcal{R}_d^f \varphi_2, t, \ell) = \mathbf{m}'(\lambda, \vec{x}, \varphi_1 \mathcal{R}_d^f \varphi_2, t, \ell)$ and $\mathbf{m}(\lambda, \vec{x}, \mathcal{E}_d^f \varphi_1, t, \ell) = \mathbf{m}'(\lambda, \vec{x}, \mathcal{E}_d^f \varphi_1, t, \ell)$, for all $\ell$ and $t$.*

## 7 CONCLUSION AND FUTURE WORK

The rise of mobile and spatially distributed CPS demands for novel efficient and effective spatio-temporal formal frameworks to specify concisely spatio-temporal requirements and to enable the qualitative and quantitative spatio-temporal monitoring of such properties over spatially distributed CPS. STREL provides an intuitive formal framework that enable to express formally spatio-temporal requirements and to monitor them automatically over the execution of *mobile* and *spatially distributed* CPS. We have demonstrated the feasibility of our approach showing an application of STREL to monitor a simulated mobile ad hoc sensor network. While in this paper we define the logic and provide an offline monitoring algorithm, future research includes the design of distributed monitoring algorithms, a thorough investigation of the expressiveness, learning STREL requirements directly from data and synthesizing control policies to ensure a given requirement. A set of API that implements the algorithms considered in this paper is currently under development[8].

## ACKNOWLEDGMENT

## REFERENCES

[1] F. Aurenhammer. 1991. Voronoi Diagrams; a Survey of a Fundamental Geometric Data Structure. *ACM Comput. Surv.* 23, 3 (1991), 345–405. https://doi.org/10.1145/116873.116880

[2] A. Aydin-Gol, E. Bartocci, and C. Belta. 2014. A Formal Methods Approach to Pattern Synthesis in Reaction Diffusion Systems. In *Proc. of CDC: the 53rd IEEE Conference on Decision and Control.* IEEE, 108–113. https://doi.org/10.1109/CDC.2014.7039367

[3] E. Bartocci, E. Aydin-Gol, I. Haghighi, and C. Belta. 2016. A Formal Methods Approach to Pattern Recognition and Synthesis in Reaction Diffusion Networks. *IEEE Transactions on Control of Network Systems* PP, 99 (2016), 1–1. https://doi.org/10.1109/TCNS.2016.2609138

[4] B. Bennett, A. G. Cohn, F. Wolter, and M. Zakharyaschev. 2002. Multi-Dimensional Modal Logic As a Framework for Spatio-Temporal Reasoning. *Applied Intelligence* 17, 3 (Sept. 2002), 239–251. https://doi.org/10.1023/A:1020083231504

[5] S. Bistarelli, U. Montanari, and F. Rossi. 1997. Semiring-based constraint satisfaction and optimization. *J. ACM* 44, 2 (1997), 201–236. https://doi.org/10.1145/256303.256306

[6] D. Bresolin, P. Sala, D. Della Monica, A. Montanari, and G. Sciavicco. 2010. A Decidable Spatial Generalization of Metric Interval Temporal Logic. In *Proc. of TIME 2010: the 17th International Symposium on Temporal Representation and Reasoning.* 95–102. https://doi.org/10.1109/TIME.2010.22

[7] L. Caires and L. Cardelli. 2003. A spatial logic for concurrency (part I). *Information and Computation* 186, 2 (2003), 194–235. https://doi.org/10.1016/S0890-5401(03)00137-8

[8] V. Ciancia, D. Latella, M. Loreti, and M. Massink. 2016. Spatial Logic and Spatial Model Checking for Closure Spaces. In *SFM 2016: 16th Intern. School on Formal Methods for the Design of Computer, Communication, and Software Systems (LNCS),* Vol. 9700. Springer, 156–201. https://doi.org/10.1007/978-3-319-34096-8_6

[9] B. Delaunay. 1934. Sur la sphère vide. *Bulletin de l'Académie des Sciences de l'URSS, Classe des sciences mathématiques et naturelles* 6 (1934), 793–800.

[10] J. V. Deshmukh, A. Donzé, S. Ghosh, X. Jin, G. Juniwal, and S. A. Seshia. 2015. Robust Online Monitoring of Signal Temporal Logic. In *Proc. of RV 2015: the 6th International Conference on Runtime Verification (LNCS),* Vol. 9333. Springer, 55–70. https://doi.org/10.1007/978-3-319-23820-3_4

[11] A. Donzé, T. Ferrer, and O. Maler. 2013. Efficient Robust Monitoring for STL. In *Proc. of CAV 2013: the 25th International Conference on Computer Aided Verification (LNCS).* 264–279. https://doi.org/10.1007/978-3-642-39799-8_19

[12] R. Grosu, S. A. Smolka, F. Corradini, A. Wasilewska, E. Entcheva, and E. Bartocci. 2009. Learning and detecting emergent behavior in networks of cardiac myocytes. *Commun. ACM* 52, 3 (2009), 97–105. https://doi.org/10.1145/1467247.1467271

[13] I. Haghighi, A. Jones, J. Z. Kong, E. Bartocci, Grosu R., and C. Belta. 2015. SpaTeL: A Novel Spatial-Temporal Logic and Its Applications to Networked Systems. In *Proc. of HSCC 2015: the 18th International Conference on Hybrid Systems: Computation and Control.* ACM, 189–198. https://doi.org/10.1145/2728606.2728633

[14] A. Lluch-Lafuente and U. Montanari. 2005. Quantitative mu-calculus and CTL defined over constraint semirings. *Theor. Comput. Sci.* 346, 1 (2005), 135–160. https://doi.org/10.1016/j.tcs.2005.08.006

[15] O. Maler and D. Nickovic. 2013. Monitoring properties of analog and mixed-signal circuits. *STTT* 15, 3 (2013), 247–268. https://doi.org/10.1007/s10009-012-0247-9

[16] M. Marx and M. Reynolds. 1999. Undecidability of compass logic. *J Logic Computation* 9, 6 (1999), 897–914. https://doi.org/10.1093/logcom/9.6.897

[17] L. Mottola, M. Moretta, K. Whitehouse, and C. Ghezzi. 2014. Team-level programming of drone sensor networks. In *Proc. of the 12th ACM Conference on Embedded Network Sensor Systems, SenSys '14, Memphis, Tennessee, USA, November 3-6, 2014.* ACM, 177–190. https://doi.org/10.1145/2668332.2668353

[18] L. Nenzi, L. Bortolussi, V. Ciancia, M. Loreti, and M. Massink. 2015. Qualitative and Quantitative Monitoring of Spatio-Temporal Properties. In *Proc. of RV 2015: the 6th International Conference on Runtime Verification (LNCS),* Vol. 9333. Springer, 21–37. https://doi.org/10.1007/978-3-319-23820-3_2

[19] C. L. Talcott. 2008. Cyber-Physical Systems and Events. In *Software-Intensive Systems and New Computing Paradigms - Challenges and Visions.* LNCS, Vol. 5380. Springer, 101–115. https://doi.org/10.1007/978-3-540-89437-7_6

[20] Y. Tan, M. C. Vuran, and S. Goddard. 2009. Spatio-Temporal Event Model for Cyber-Physical Systems. In *2009 29th IEEE International Conference on Distributed Computing Systems Workshops.* IEEE, 44–50. https://doi.org/10.1109/ICDCSW.2009.82

---

[8]STREL API are public available at https://github.com/Quanticol/strel