Random generation of a locally consistent spatio-temporal graph

Aurelie Leborgne, Marija Kirandjiska, Florence Le Ber

Random generation of a locally consistent spatio-temporal graph

Aurélie Leborgne[1], Marija Kirandjiska[1], and Florence Le Ber[1]

Université de Strasbourg, CNRS, ENGEES, ICube UMR 7357, F-67000 Strasbourg
{aurelie.leborgne,florence.le-ber}@unistra.fr

Abstract. In this paper, an original approach, for the generation of locally coherent spatio-temporal graphs embeddding frequent inexact patterns, is presented. This approach is based on a work carried out previously, in which a configurable generator for spatio-temporal graphs has been implemented. These graphs contain spatial and spatio-temporal edges that are labelled with the RCC8 topological relations. The objective being to check the consistency of these relations during the construction of the graph, the path-consistency method, based on the composition of weak relation, has been implemented in the graph generator. The approach is precisely described and some experiences are detailed. Our final objective is to build a generator allowing to generate test bases that can be used to highlight the advantages and disadvantages of graph extraction methods.

Keywords: Spatio-temporal graph · RCC8 · consistency · graph generation.

1 Introduction

The regular improvement of data collection tools and techniques leads more and more often to model and analyze data that have a spatial but also a temporal dimension. A natural way to model such data is to use spatio-temporal graphs (ST graphs), which can be used to represent different phenomena such as events in videos [19], movement of dunes on the seaside [9], or brain activity [12], *etc.*.

In order to analyze the collected data, e.g. for extracting recurrent phenomena in time and/or space, it is necessary to develop algorithms to search for frequent patterns/subgraphs in ST graphs. However, to develop such algorithms, it is essential to have a test base of annotated ST graphs. Unfortunately, the annotation of such data is a very tedious task and even impossible to perform precisely given the amount of data we handle. The only solution to obtain a test base is therefore to develop a spatio-temporal graph generator in which we master the frequent patterns present in these ST graphs.

The generation of such graphs has been described in [11]. In this paper, we are interested in how to manage the coherence of graphs for the considered set of relations, namely, the qualitative spatial relations of RCC8 theory [15]. These relations make it possible to model the evolution of land use on a territory. Fig.

1, for example, represents the evolution of a set of neighboring plots (modeled by polygons) over time.

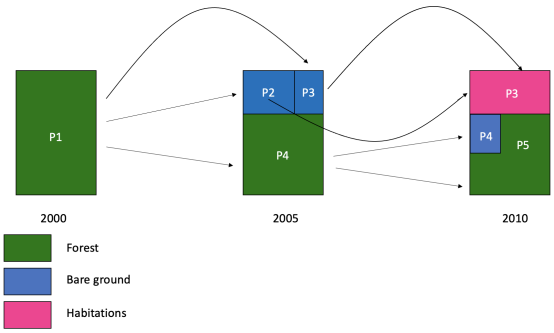


Fig. 1: Evolution of land use in a territory

This article is organized as follows. Section 2 describes the theoretical elements on which our approach is based, namely the spatio-temporal graph model, the RCC8 relations and then the qualitative constraint networks. Sections 3 and 4 present the generation algorithm and the experiments carried out. Finally, a conclusion closes the article.

2 Preliminaries

2.1 Graphs simulation

Graph generation is an important issue in many fields, to simulate real graphs, to test algorithms or to analyze, visualize or transform data [3]. In most cases, the goal is to generate realistic graphs. Many models have been presented in this sense for the generation of complex graphs adapted to the representation of natural or human systems (semantic web, social networks). Most approaches are based on statistical distributions of graph properties (number of vertices, number of edges, degree of vertices, *etc.*). Barabási-Albert model [5, 4] is one of the best known. In the Chung-Lu model [1] the probability of an edge is proportional to the product of the degrees of its two vertices. The Recursive MATrix method (R-MAT) [6] and Park and Kim model [13] are recursive models to generate synthetic graphs. The RDyn approach was recently designed [17] for generating temporal graphs representing the dynamics of network communities. In a context closer to ours, Kuramochi and Karypis [10] proposed an algorithm to generate random general graphs including known patterns to create synthetic datasets. Nevertheless, the consistency issue in the generated graph is not addressed, as edges have no semantics in these approaches.

In [11], we have developed an approach to generate, in a random way, semantic-temporal graphs, where the edges are endowed with semantics. We introduce be-

low the model of such a graph, inspired by [9]. This is a spatio-temporal graph, defined as the union of three sub-graphs:

- **The graph of spatial relations**, that spatially characterizes the interactions between entities at a given time. It is composed of nodes (disks), and edges (in green) on Fig. 2.
- **The graph of spatio-temporal relations**, which is based on the same characteristics as the graph of spatial relations, but considering entities at different times. It is composed of nodes (disks), and edges (in red) on Fig. 2.
- **The graph of filiation relations**, defines the concept of identity. It allows to characterize the transmission of the identity through time. It is composed of nodes (disks), and edges (in blue) on Fig. 2.

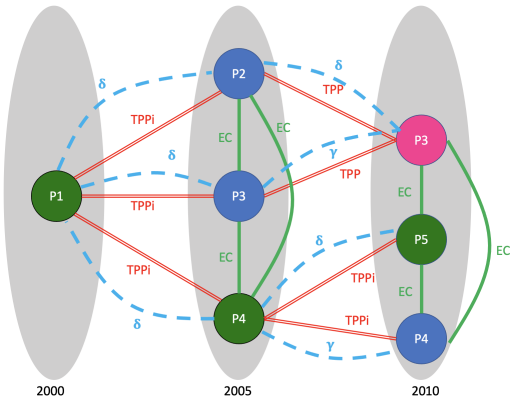


Fig. 2: ST graph modeling the evolution of the territory of Fig. 1

Formally, a ST graph is defined as follows. Let $\mathcal{T} = \{t_1, t_2, ...t_n\}$, a time domain, where t_i represents a time instance of a given granularity and $t_i < t_{i+1}$ for all $i \in [1, n]$. Let Δ a set of entities, $\{e_1, e_2, ..., e_m\}$. Let also Σ, a set of spatial relations, and Φ, a set of filiation relations.

A spatio-temporal graph $\mathcal{G}$ is a tuple (U, E_Σ, E_Φ), where U is the set of vertices $(e_l, t_i) \in \Delta \times \mathcal{T}$, E_Σ is the set of tuples $((e_i, t_i)T(e_j, t_j))$ where $(e_i, t_i), (e_j, t_j) \in U$, $t_i \leq t_j \leq t_{i+1}$, and $T \in \Sigma$, and E_Φ is the set of tuples $((e_i, t_i)\rho(e_j, t_{i+1}))$ where $(e_i, t_i), (e_j, t_{i+1}) \in U$, and $\rho \in \Phi$.

This graph model was initially introduced to represent the evolution of geographical entities [9]. Other filiation relations are studied in [8]. Besides, an algorithm has been proposed to check the consistency of this representation with regards to the specifications of a spatio-temporal database.

2.2 RCC8 relations

The spatial and spatio-temporal relations we use are the base relations **B** of the RCC8 theory [15] on the spatial domain Δ. These relations define the position

of two regions : $DC(x, y)$ regions x and y are disconnected ; $EC(x, y)$ they are externally connected ; $PO(x, y)$ they partially overlap ; $TPP(x, y)$ x is a tangential proper part of y; $TPP_i(x, y)$ y is a tangential proper part of x; $NTPP(x, y)$ x is a non-tangential proper part of y; $NTPP_i(x, y)$ y is non-tangential proper part of x ; $EQ(x, y)$ x and y are equal (see Figure 3).

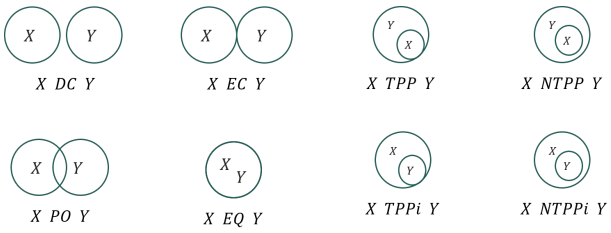


Fig. 3: The 8 base relations of RCC8 theory

Set $2^{\mathbf{B}}$ represents the set of relations constructed from the base relations. It is provided with the usual set operations, union and intersection, the inverse operation and the weak composition. A relation of $2^{\mathbf{B}}$ is therefore written as a union of basic relations, for example $R = \{DC, EC\}$ and is interpreted as a disjunction. The opposite (denoted $\smile$) of a relation is the union of the inverses of its base relations. The weak composition is noted $\diamond$ and defined as follows: Let R and S two relations of $2^{\mathbf{B}}$, $R \diamond S = \{b \in \mathbf{B} | b \cap (R \circ S) \neq \emptyset\}$ where $R \circ S = \{(x, z) \in \Delta^2 | \exists y \in \Delta, (x, y) \in R \, \mathtt{et}(y, z) \in S\}$. The weak composition of the basic relations is represented in a composition table [14], as shown in Fig. 7. For example, suppose that three regions x, y, z are such as $TPP(x, y)$ and $EC(y, z)$ holds, then $\{DC, EC\}(x, z)$ holds (see Fig. 4).

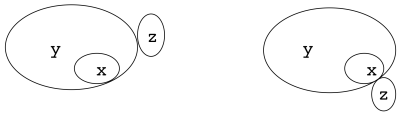


Fig. 4: Two possible configurations for x and z knowing $TPP(x, y)$ and $EC(y, z)$

2.3 Consistency

The notion of consistency of a set of spatial or temporal relations between regions has been studied in the framework of qualitative constraint networks [7]. A network of qualitative constraints is a couple $N = (V, C)$ where V is a set of variables on a continuous domain $\mathcal{D}$ and C an application that associates to each pair of variables (V_i, V_j) a set C_{ij} of base relations $\{r_1, ..., r_l\}$ taken from

an algebra of relations. This set represents the disjunction of possible relations between the two entities represented by the variables V_i and V_j. A consistent instantiation of N is one where each variable V_i takes a value $e_i \in \mathcal{D}$ such that for any pair (V_i, V_j) the atomic relation verified by the variables V_i and V_j belongs to C_{ij}.

Checking the consistency of a network is an NP-complete problem in the general case [20]. Local methods have been proposed to check weaker forms of consistency, including path-consistency: a qualitative constraint network N is said to be path-consistent if for all variables V_i, V_j, $V_k \in V$, $C_{ij} \subseteq C_{ik} \circ C_{kj}$ [7].

The path-consistency method consists in performing the triangulation operation : $C_{ij} = C_{ij} \cap (C_{ik} \circ C_{kj})$ for any triplet until a fixed point is obtained. The final network is path-consistent and equivalent to the initial network. In the framework of RCC8, where we can't use the composition, but the weak composition $\diamond$, we will speak about algebraic closure [16].

Various algorithms have been proposed to check the path-consistency. Their computation time is related to the number of accesses to the composition table and thus to the size of the relation set $2^{\mathbf{B}}$ [2]. The authors of [18] present an algorithm for incrementally checking the consistency of a qualitative constraint network, which grows by adding spatial or temporal entities. The algorithm exploits a triangulated graph: when an entity is added at step t, it is linked to all the entities present at step $t-1$ (triangulation). The path-consistency method is then applied to the graph.

Our approach is different as our aim is to generate a graph where each relation is atomic, and where the issue is to assign to the current edge a relation which is consistent with the existing ones, as we detail below.

3 Generation of locally consistent ST graphs

We present here an approach for generating locally consistent spatio-temporal graphs including frequent inexact patterns (subgraphs). We rely on the method presented in [11], which allows to generate such graphs, including patterns, but without dealing with the issue of consistency.

We briefly describe here the algorithm for generating spatio-temporal graphs and their embedded patterns, and then detail and justify the algorithm for checking the local consistency of the generated spatial and spatio-temporal relations. Let us note that these two types of relations are derived from the set $2^{\mathbf{B}}$ (see Sect. 2.2).

The algorithm described in [11] simulates spatio-temporal graphs containing patterns *drown* in a uniform stochastic generation of nodes and edges. It is a fully parameterizable algorithm, exploiting Poisson's law, in which it is possible to choose the size of the generated graph, the number of spatial, temporal and filiation relations per node, as well as the size of the source-patterns to be embedded and the number of their transformations. In this paper, we consider a modified algorithm to control the proportion of patterns, in number of nodes, in the total graph. The other parameters are listed in Table 1.

Parameter	Description
λ_n	expectation of the zero-truncated Poisson distribution for the total number of nodes in the graph
λ_r	expectation of the zero-truncated Poisson distribution for the number of nodes per time instant
Λ_e	Triplet of Poisson law expectations for the number of spatial/spatio-temporal/filiation relations per node
$labels_n$	List of available labels for nodes
$labels_e$	Size 3 table of available label lists for each type of relation
patterns	A list of tuples where each tuple is composed of a time instant number and the pattern to insert at this time instant

Table 1: Parameters for the generation of spatio-temporal graphs

This algorithm has three main steps. It first calculates the total number of nodes (parameter λ_n, Tab. 1) of the graph to be generated.

- **Step 1**: generation and transformation of source-patterns, according to specific parameters (see below); each pattern is assigned to a time instant of the graph, this information is stored in the parameter *patterns*.
- **Step 2**: random generation of nodes for each time instant in the graph (parameters λ_r and $labels_n$). The number of nodes is adjusted according to the number of nodes in the patterns assigned to the current time instant.
- **Step 3**: random generation of relations between nodes (parameters Λ_e and $labels_e$). The nodes of the current time instant are connected to each other and to the nodes of the previous time instant. The number of edges is adjusted for the nodes of the patterns (if they already have any). Each edge is labeled with an atomic relation.

In the first step, the pattern generation is based on the following parameters (Tab. 2): the number of inserted patterns depends on a proportion p (as a percentage of the number of nodes) that these patterns should represent in the total graph. The number of nodes in a source-pattern is randomly drawn in a range (*pnodes*). Parameters λ_r and λ_e have the same role as for the complete graph. Each source-pattern is repeated according to a value (support) randomly drawn in the interval *support*. Finally, each repetition gives rise to transformations (parameter λ_t) in order to introduce variations based on each source pattern.

The average theoretical complexity of this algorithm is $O(\lambda_n \times \lambda_r)$, each node of the graph being potentially linked to all nodes of the same time instant and of the preceeding time instant. In the worst case, when the number of time instants decreases, the complexity tends to $O(\lambda_n^2)$ [11].

The objective of the work presented here is to generate a locally consistent graph based on the consistency model presented in Sect. 2.3. More precisely, it consists in generating spatial and spatio-temporal relations while making sure that they form spatial or temporal consistent triangles with the existing edges in the graph.

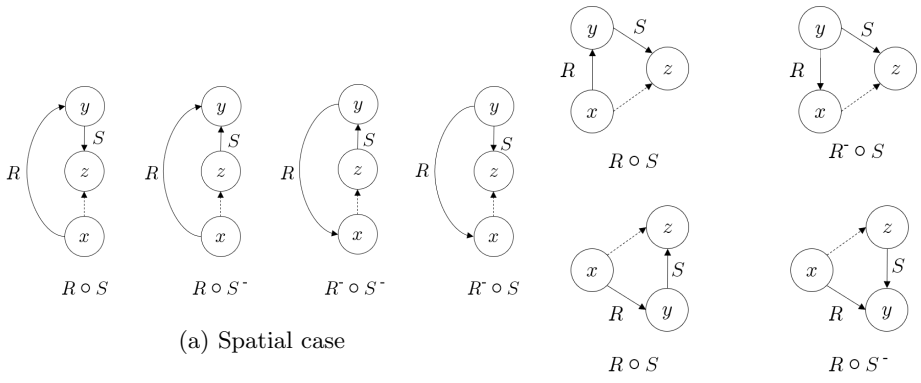
Parameter	Description
p	Proportion of nodes in the patterns / number in the graph
pnodes	Interval for the number of nodes in a source pattern
λ_r	expectation of the zero-truncated Poisson distribution for the number of nodes per time instant
Λ_e	Triplet of Poisson law expectations for the number of spatial/spatial-temporal/filiation relations per node
support	Interval for the number of repetitions of a source-pattern
λ_t	expectation of the Poisson law for the number of transformations to be performed on a source-pattern

Table 2: Parameters for pattern generation

Definition 1. *A **consistent triangle** is a clique of 3 vertices in which the three relations modeled by edges are consistent with each other, that is, e_i, e_j, e_k being the vertices of such a triangle, $R_{ik} \subseteq R_{ij} \diamond R_{jk}$ and $R_{ij} \subseteq R_{ik} \diamond R_{kj}$.*

Definition 2. *A subgraph consisting of three nodes that belong to the same time instant will be called a **spatial triangle**. The composition schemes of the 4 possible cases, depending on the direction of the edges xy and yz, are shown in Fig. 5a.*

Definition 3. *A subgraph consisting of three nodes that belong to two successive time instants will be called a **temporal triangle**. Composition schemes of the 4 possible cases, depending on the direction of the edges xy and yz, are presented in Fig. 5b.*



(a) Spatial case

(b) Spatio-temporal case

Fig. 5: Different configurations for the composition of edges connecting the nodes x and y on the one hand, and y and z on the other hand, in a triangle

Choosing to limit our approach to a local coherence (3-coherence) is linked to practical aspects: on the one hand, the embedded patterns are small (3-4 nodes per time instant, 2 or 3 time instants at most); on the other hand, the number of relations per node is generally low (even if experiments have been conducted with high densities, see Sect. 4). Finally, we seek to limit the complexity of ST graph generation.

Algorithm 1 describes this method. An initialization phase is necessary (l. 1). It allows to initialize a list L with all the relations of $\mathbf{B}$. To determine a relation between nodes x and z, we search all the nodes y, that have a relation with both x and z (l. 2). The different configurations (Fig. 5) are considered. For each node, the list is updated by keeping only the possible relations among those already present in L (l. 4, 6, 8, 10). Finally, the relation between x and z is randomly assigned among the possible relations (present in L) (l. 14). In case the list is empty, no relation is assigned.

The theoretical complexity of algorithm 1 is $O(\lambda_r)$ since, given a pair (x, z), it examines at most all nodes in the current and preceding time instants. In contrast, since all relations are atomic relations, only one access to the composition table is needed for processing a triangle.

Algorithm 1 Generation of a relation between two nodes with local consistency check

Input: nodes x, z
Output: relation between x and z

1: $L = \mathbf{B}$
2: **for each** node y such that (x,y) and (y,z) $\in E_\Sigma$ **do**
3: **if** $R(x, y)$ and $S(y, z)$ **then**
4: $L \leftarrow L \cap R \diamond S$
5: **else if** $R(x, y)$ and $S(z, y)$ **then**
6: $L \leftarrow L \cap R \diamond S^\smile$
7: **else if** $R(y, x)$ and $S(z, y)$ **then**
8: $L \leftarrow L \cap R^\smile \diamond S^\smile$
9: **else**
10: $L \leftarrow L \cap R^\smile \diamond S$
11: **end if**
12: **end for**
13: **if** $L \neq \emptyset$ **then**
14: **return** random relation in L
15: **else**
16: **return** no relation
17: **end if**

An example of an obtained graph is presented in Fig. 6: pattern nodes are designated by the letter P, while generic nodes have only a number. Nodes with the same number have the same label. On this example, at time t_4, P2 is connected to P1 by a spatial edge $NTPP$, P2 is connected to P3 (t_5) by a spatio-temporal

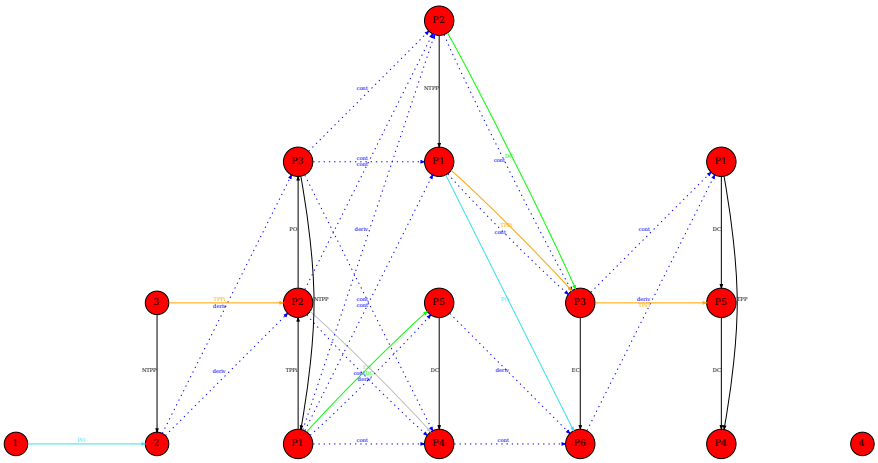


Fig. 6: A coherent spatio-temporal graph including patterns: spatial edges are represented in black, spatio-temporal edges in color, filiation edges in dashed lines

edge DC, P1 and P3 are connected to P6 (t_5) respectively by edges having labels PO and EC. To connect P1 and P3, the algorithm must successively examine triangles (P1,P2,P3) and (P1,P6,P3):

- For (P1,P2,P3), we have $NTPP^{\smile} \diamond DC = NTTP_i \diamond DC = \{DC, EC, PO, TPP_i, NTPP_i\}$, as shown in the blue box of the composition table (Fig. 7).
- For (P1,P6,P3), we have $PO \diamond EC^{\smile} = PO \diamond EC = \{DC, EC, PO, TPP_i, NTPP_i\}$, as shown in the yellow box of the composition table (Fig. 7).

Finally the relation between P1 and P3 must be chosen in the set $\{DC, EC, PO, TPP_i, NTPP_i\}$, here TPP_i has been selected.

4 Experimentations

In this experimental phase we studied the performance, in terms of computation time, of the generation algorithm by varying the different parameters (see Tab. 1 and 2).

This set of tests was performed on an Ubuntu machine 18.04.4 LTS, 32 Go of RAM and 32 cores. However our algorithm only used one of these cores.

In order to observe the influence of the different parameters on the complexity of the generation of consistent spatio-temporal graphs, we varied these parameters one by one.

$\diamond$	DC	EC	PO	TPP	$NTPP$	TPP_i	$NTPP_i$	EQ
DC	DC, EC, PO, TPP, $NTPP$, TPP_i, $NTPP_i$, EQ	DC, EC, PO, TPP, $NTPP$	DC, EC, PO, TPP, $NTPP$	DC, EC, PO, TPP, $NTPP$	DC, EC, PO, TPP, $NTPP$	DC	DC	DC
EC	DC, EC, PO, TPP_i, $NTPP_i$	DC, EC, PO, TPP, TPP_i, EQ	DC, EC, PO, TPP, $NTPP$	EC, PO, TPP, $NTPP$	PO, TPP, $NTPP$	DC, EC	DC	EC
PO	DC, EC, PO, TPP_i, $NTPP_i$	DC, EC, PO, TPP_i, $NTPP_i$	DC, EC, PO, TPP, $NTPP$, TPP_i, $NTPP_i$, EQ	PO, TPP, $NTPP$	PO, TPP, $NTPP$	DC, EC, PO, TPP_i, $NTPP_i$	DC, EC, PO, TPP_i, $NTPP_i$	PO
TPP	DC	DC, EC	DC, EC, PO, TPP, $NTPP$	TPP, $NTPP$	$NTPP$	DC, EC, PO, TPP, TPP_i, EQ	DC, EC, PO, TPP_i, $NTPP_i$	TPP
$NTPP$	DC	DC	DC, EC, PO, TPP, $NTPP$	$NTPP$	$NTPP$	DC, EC, PO, TPP, $NTPP$	DC, EC, PO, TPP, $NTPP$, TPP_i, $NTPP_i$, EQ	$NTPP$
TPP_i	DC, EC, PO, TPP_i, $NTPP_i$	EC, PO, TPP_i, $NTPP_i$	PO, TPP_i, $NTPP_i$	EQ, PO, TPP, TPP_i	PO, TPP, $NTPP$	TPP_i, $NTPP_i$	$NTPP_i$	TPP_i
$NTPP_i$	DC, EC, PO, TPP_i, $NTPP_i$	PO, TPP_i, $NTPP_i$	PO, TPP_i, $NTPP_i$	PO, TPP_i, $NTPP_i$	PO, TPP, $NTPP$, EQ, TPP_i, $NTPP_i$	$NTPP_i$	$NTPP_i$	$NTPP_i$
EQ	DC	EC	PO	PO	$NTPP$	TPP_i	$NTPP_i$	EQ

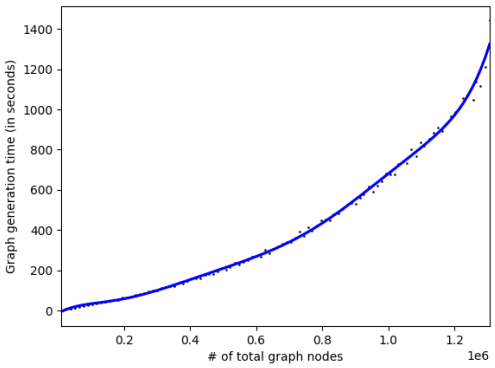
Fig. 7: Composition table for the base relations of RCC8 theory

4.1 Variation of the node number

In this first experiment, only the parameter λ_n varies, which determines the total number of nodes in the graph. Parameter λ_r is fixed so that the number of time instant does not change (the number of nodes per time instant varies proportionally to the total number of nodes, see Tab. 8 (a)). Figure 8 (b) shows that the generation time of the graphs grows exponentially with the total number of nodes and thus with the number of nodes per time instant. In fact, for each node created, the algorithm has to go through the nodes of the same time instance and of the previous one to establish the relations: as λ_r is the average number of nodes per time instant, so on average there are at most $2\lambda_r \times \lambda_n \approx \lambda_n^2$ operations.

Parameter	Value
graph generation	
λ_n	varies from 10000 to 10^6
λ_r	λ_n / 100
Λ_e	[5,5,2]
Patterns generation	
p	30
pnodes	[5,15]
λ_r	2
Λ_e	[5,5,2]
support	[10,20]
λ_t	average pattern size / 2

(a)



(b)

Fig. 8: Graph generation time as a function of the total number of nodes (a) Parameters (b) Resulting curve

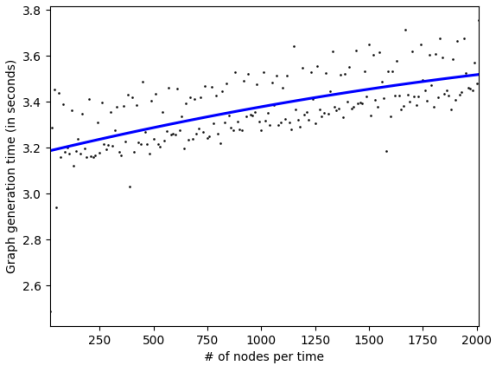
In a second step, in order to examine the influence of the number of nodes per time, we fixed the total number of nodes and varied the number of nodes per time instance. This experiment shows that there is a linear relation between the generation time of the graphs and the number of nodes per time (see Fig. 9 (b)), for a fixed total number of nodes. Everything being fixed, the only variation is due to the number of nodes to be visited to establish the relations in the current time instance and with the previous one. As above, the number of operations is on average $2\lambda_n \times \lambda_r$. Parameter λ_n being fixed, the calculation time is therefore proportional to λ_r.

4.2 Variation of the edge number

We are interested here in the influence of the generation of edges on the generation time of graphs. We varied therefore the number of relations per node. In this experiment, the total number of nodes and nodes per time instant are fixed. Figure 10 illustrates this experience. There is a linear relationship between the number of relations per node and the generation time of the complete graph until we reach a plateau with around 100 relations per node. The linear part can be explained in this way: for each node we have on average $\lambda_e = \Lambda_e[1]+\Lambda_e[2]+\Lambda_e[3]$ creation of edges and $\Lambda_e[1]+\Lambda_e[2]$ constraint checkings to be performed. For the whole graph, there are $\lambda_n \times \lambda_e$ operations, this value growing linearly with λ_e, λ_n being fixed. The constant part is due to the saturation of the graph, *i.e.* each node has reached its maximum number of edges. Indeed, in this experiment, a time instance contains an average of 100 nodes, each of which has at most one relation of each type with a node of the same or previous time instance. Note that the computational time associated with the consistency checking of rela-

Parameter	Value
Graph generation	
λ_n	10000
λ_r	varies from 100 to 2000
Λ_e	[5,5,2]
Patterns generation	
p	30
pnodes	[5,15]
λ_r	2
Λ_e	[5,5,2]
support	[10,20]
λ_t	average pattern size / 2

(a)



(b)

Fig. 9: Generation time of a graph as a function of the number of nodes per time instant (a) Parameters (b) Resulting curve

tions is only related to the number of triplets to examine, since each edge carries only an atomic relation.

4.3 Variation of the pattern number

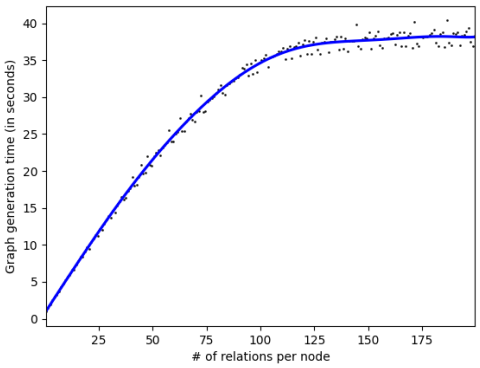
In this last experiment, we vary the number of patterns inserted in the graphs (or more exactly the proportion of nodes coming from patterns, set by the parameter p, Fig. 11 (a)), the graph size being constant. Figure 11 (b) shows that by increasing this proportion, the generation time of the graphs increases linearly. This can be explained as follows: the generation and transformation time of the source patterns is constant (parameters *pnodes*, λ_r, Λ_e, *support* and λ_t are fixed), the size of the patterns is constant (parameter *pnodes*), only the number of source-patterns to be generated to reach a given proportion of nodes wrt the total number of nodes in the graph varies.

5 Conclusion

This paper presents a method to generate spatio-temporal graphs which spatial and spatio-temporal edges are locally consistent. To do so, we rely on an existing algorithm for generating random spatio-temporal graphs [11]. This algorithm has been modified in the sense that at each addition of an edge (spatial or spatio-temporal) between two nodes, the consistency of this edge with the pre-existing edges connecting these two nodes to the same nodes is checked. The path-consistency method is used for this purpose, in the framework of the RCC8 theory.

Parameter	Value
Graph generation	
λ_n	10000
λ_r	100
Λ_e	varies from [0,0,0] to [200,200
Patterns generation	
p	30
pnodes	[5,15]
λ_r	2
Λ_e	[5,5,2]
support	[10, 20]
λ_t	average pattern size / 2

(a)



(b)

Fig. 10: Graph generation time as a function of the number of relations per node (a) Parameters (b) Resulting curve

The particularity of the algorithm is to insert in the generated graph frequent inexact patterns. These patterns are generated and transformed, also including the local consistency checking.

In addition, complexity tests were performed in order to highlight the most costly steps in the generation of spatio-temporal graphs. These tests have confirmed that the added consistency checking step does not change the general complexity of the original algorithm.

Our general aim is to generate 'realistic' graphs, with respect to an application domain, here the evolution of farming territories. Generated graphs are 'realistic' if they are similar to graphs representing real spatio-temporal data. Spatial consistency is a characteristic of these real graphs. Other characteristics (e.g., node degree for the various relation types) will also be studied.

In the future, a more extensive path-consistency could be developed, following the work of [18]. Furthermore, graphs generated by this approach will be used to test various frequent pattern mining methods in a spatio-temporal graph. Qualitative experiments will then be conducted to evaluate the quality of the frequent patterns that will be mined on the generated spatio-temporal graphs.

Acknowledgements

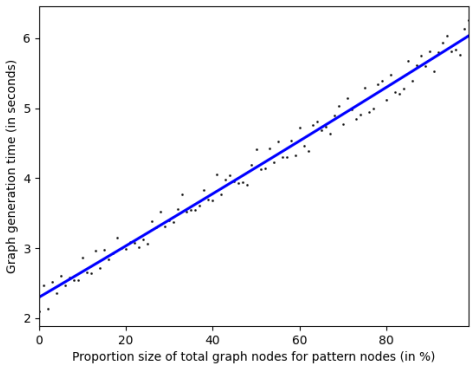
This work was carried out in the framework of the METEC-Graphe project, supported by the Idex Université de Strasbourg.

References

1. Aiello, W., Chung, F., Lu, L.: A random graph model for massive graphs. In: Proc. of the 32th ACM symposium on Theory of computing, STOC'00. pp. 171–

Parameter	Value
Graph generation	
λ_n	10000
λ_r	100
Λ_e	[5,5,2]
Patterns generation	
p	varies from 0 to 100%
pnodes	[5,15]
λ_r	2
Λ_e	[5,5,2]
support	[10,20]
λ_t	average pattern size / 2

(a)



(b)

Fig. 11: Graph generation time as a function of the proportion of inserted patterns (a) Parameters (b) Resulting curve

180 (2000)
2. Bessière, C.: A simple way to improve path consistency processing in interval algebra networks. In: Proceedings of the 13th National Conference on Artificial Intelligence (AAAI-96). pp. 375–380 (1996)
3. Bonifati, A., Holubová, I., Prat-Pérez, A., Sakr, S.: Graph generators: State of the art and open challenges. ACM Comput. Surv. **53**(2) (April 2020)
4. Campbell, C., Shea, K., Albert, R.: Comment on control profiles of complex networks. Science **346**(6209), 561–561 (2014)
5. Ferrer i Cancho, R., Solé, R.: Optimization in complex networks. In: Statistical mechanics of complex networks, pp. 114–126 (2003)
6. Chakrabarti, D., Zhan, Y., Faloutsos, C.: R-mat: A recursive model for graph mining. In: Proceedings of the 2004 SIAM Int Conference on Data Mining. pp. 442–446 (2004)
7. Condotta, J.F., Ligozat, G., Saade, M.: Empirical study of algorithms for qualitative temporal or spatial constraint networks. In: Proceedings of the 17th European Conference on Artificial Intelligence, Riva del Garda, Italy (2006)
8. Del Mondo, G., Rodríguez, M.A., Claramunt, C., Bravo, L., Thibaud, R.: Modeling consistency of spatio-temporal graphs. Data & Knowledge Engineering **84**, 59–80 (2013)
9. Del Mondo, G., Stell, J.G., Claramunt, C., Thibaud, R.: A graph model for spatio-temporal evolution. Journal of Universal Computer Science **16**, 1452–1477 (2010)
10. Kuramochi, M., Karypis, G.: Frequent subgraph discovery. In: Proceedings 2001 IEEE Int Conference on Data Mining. pp. 313–320 (2001)
11. Leborgne, A., Nuss, J., Le Ber, F., Marc-Zwecker, S.: An approach for generating random temporal semantic graphs with embedded patterns. In: Graph Embedding and Mining, ECML-PKDD 2020 Workshop Proc. (2020)
12. Leborgne, A., Le Ber, F., Degiorgis, L., Harsan, L., Marc-Zwecker, S., Noblet, V.: Analysis of brain functional connectivity by frequent pattern mining in graphs. application to the characterization of murine models. In: Proceedings of the International Symposium on Biomedical Imaging ISBI'2021. pp. 1–4 (2021)

13. Park, H., Kim, M.S.: Trilliong: A trillion-scale synthetic graph generator using a recursive vector model. In: Proceedings of the 2017 ACM Int Conference on Management of Data. pp. 913–928 (2017)
14. Randell, D.A., Cohn, A.G., Cui, Z.: Computing transitivity tables: A challenge for automated theorem provers. In: Kapur, D. (ed.) Automated Deduction—CADE-11. pp. 786–790. Springer Berlin Heidelberg (1992)
15. Randell, D.A., Cui, Z., Cohn, A.G.: A Spatial Logic based on Regions and Connection. In: Proceedings 3rd International Conference on Knowledge Representation and Reasoning. pp. 165–176. Morgan Kaufmann Publishers (1992)
16. Renz, J., Ligozat, G.: Weak composition for qualitative spatial and temporal reasoning. In: van Beek, P. (ed.) Principles and Practice of Constraint Programming - CP 2005. pp. 534–548. Springer (2005)
17. Rossetti, G.: RDYN: graph benchmark handling community dynamics. Journal of Complex Networks **5**(6), 893–912 (2017)
18. Sioutis, M., Condotta, J.F.: Incrementally building partially path consistent qualitative constraint networks. In: AIMSA 2014: Artificial Intelligence: Methodology, Systems, and Application. pp. 104–116 (09 2014)
19. Sridhar, M., Cohn, A.G., Hogg, D.C.: Relational graph mining for learning events from video. In: Stairs 2010: Proceedings of the Fifth Starting AI Researchers' Symposium. pp. 315–327 (2011)
20. Vilain, M., Kautz, H., Beek, P.V.: Constraint propagation algorithms for temporal reasoning: A revised report. In: Weld, D.S., De Kleer, J. (eds.) Readings on Qualitative Reasoning about Physical Systems, pp. 373–381. Morgan Kaufmann (1989)