International Journal on Artificial Intelligence Tools

An Efficient Approach for Tackling Large Real World Qualitative Spatial Networks

Michael Sioutis and Jean-François Condotta
Université Lille-Nord de France, Université d'Artois
CRIL-CNRS UMR 8188, Lens, France
{sioutis,condotta}@cril.fr

Manolis Koubarakis
National and Kapodistrian University of Athens
Department of Informatics and Telecommunications, Athens, Greece
koubarak@di.uoa.gr

We improve the state-of-the-art method for checking the consistency of large qualitative spatial networks that appear in the Web of Data by exploiting the scale-free-like structure observed in their constraint graphs. We propose an implementation scheme that triangulates the constraint graphs of the input networks and uses a hash table based adjacency list to efficiently represent and reason with them. We generate random scale-free-like qualitative spatial networks using the Barabási-Albert (BA) model with a preferential attachment mechanism. We test our approach on the already existing random datasets that have been extensively used in the literature for evaluating the performance of qualitative spatial reasoners, our own generated random scale-free-like spatial networks, and real spatial datasets that have been made available as Linked Data. The analysis and experimental evaluation of our method presents significant improvements over the state-of-the-art approach, and establishes our implementation as the only possible solution to date to reason with large scale-free-like qualitative spatial networks efficiently.

1. Introduction

Spatial reasoning is a major field of study in Artificial Intelligence, and in Knowledge Representation in particular. This field has gained a lot of attention during the last few years as it extends to a plethora of areas and domains that include, but are not limited to, ambient intelligence, dynamic GIS, cognitive robotics, spatiotemporal design, planning, and reasoning and querying with semantic geospatial query languages [1–3]. In this context, an emphasis has been made on qualitative spatial reasoning which relies on qualitative abstractions of spatial aspects of the commonsense background knowledge, on which our human perspective of physical reality is based. The concise expressiveness of the qualitative approach provides a promising framework that further boosts research and applications in the aforementioned areas and domains.

The Region Connection Calculus (RCC) is the dominant Artificial Intelligence approach for representing and reasoning about topological relations [4]. RCC can be

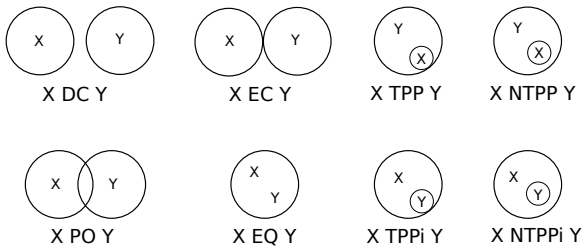


Fig. 1. Two dimensional examples for the eight base relations of RCC-8

used to describe regions that are non-empty regular closed subsets of some topological space by stating their topological relations to each other. RCC-8 is the constraint language formed by the following 8 binary topological relations of RCC: disconnected (DC), externally connected (EC), equal (EQ), partially overlapping (PO), tangential proper part (TPP), tangential proper part inverse ($TPPi$), non-tangential proper part ($NTPP$), and non-tangential proper part inverse ($NTPPi$). These eight relations are depicted in Figure 1.

In the literature of qualitative spatial reasoning there has been a severe lack of datasets for experimental evaluation of the reasoners involved. In most cases, datasets consist of randomly generated regular networks that have a uniform node degree distribution [5] and scale up to a few hundred nodes in experimental evaluations [6, 7]. These networks are often very hard to solve instances that do not correspond to real case scenarios [8] and are mainly used to test the efficiency of different algorithms and heuristic implementations. There has been hardly any practical exploitation of the stuctural properties of the networks' underlying constraint graphs. (However, a nice theoretical contribution exists in [9].) In the case where the datasets are real, they are mainly small and for proof of concept purposes, such as the one used in [10], with the exception of a real large scale and *successfully* used dataset employed in [11], viz., the `admingeo` dataset [12]. In fact, we will make use of this dataset again in this paper, along with an even bigger one that scales up to nearly a million of topological relations.

It has come to our attention that the real case scenario datasets we are particularly interested in correspond to graphs with a scale-free-like structure, i.e., the degree distribution of the graphs follows a power law. Scale-free graphs seem to match real world applications well and are widely observed in natural and human-made systems, including the Internet, the World Wide Web, and the Semantic Web [13–16]. We argue that the case of scale-free graphs applies also to qualitative spatial networks and we stress on the importance of being able to efficiently reason with such scale-free-like networks for the following two main reasons:

- The natural approach for describing topological relations inevitably leads to the creation of graphs that exhibit hubs for particular objects which are cited more than others due to various reasons, such as size, significance, and importance.

These hubs are in fact the most notable characteristic in scale-free graphs [13,14]. For example, if we were to describe the topological relations in Greece, Greece would be our major hub that would relate topologically to all of its regions and cities, followed by smaller hubs that would capture topological relations within the premises of a city or a neighborhood. It would not really make sense to specify that the porch of a house is located inside Greece, when it is already encoded that the house is located inside a city of Greece. Such natural and human-made systems are most often described by scale-free graphs [13–16].

- Real qualitative spatial datasets that are known today, such as `admingeo` [12] and `gadm-rdf` (`http://gadm.geovocab.org/`), come from the Semantic Web, also called the Web of Data, which is argued to be scale-free [15]. Further, more real datasets are to be offered by the Semantic Web community since RCC-8 has already been adopted by GeoSPARQL [3], and there is an ever increasing interest in coupling qualititave spatial reasoning techniques with linked geospatial data that are constantly being made available [17, 18]. Thus, there is a real need for scalable implementations of constraint network algorithms for qualitative and quantitative spatial constraints as RDF stores supporting linked geospatial data are expected to scale to billions of triples [17, 18].

In this paper, we concentrate on the consistency checking problem of large scale-free-like qualitative spatial networks and make the following contributions: (*i*) we explore and take advantage of the stuctural properties of the considered networks and propose an implementation scheme that triangulates their constraint graphs to retain their sparseness and uses a hash table based adjacency list to efficiently represent and reason with them, (*ii*) we investigate the pruning capacity of our partial local consistency approach that operates on a triangulated graph of a given network when compared with the local consistency approach that runs on the complete graph of that network, (*iii*) we make the case for a new series of random datasets, viz., *large* random scale-free-like RCC-8 networks, that can be of great use and value to the qualitative reasoning community and identify the phase transition for those datasets, (*iv*) we experimentally evaluate our approach against the state-of-the-art reasoners GQR [7], Renz's solver [5], and two of our own implementations which we briefly present here, viz., Phalanx and Phalanx$\triangledown$, and show that it significantly advances the state-of-the-art approach, and finally (*v*) we investigate and experiment with a state-of-the-art SAT-based implementation for RCC-8, and show that we outperform it by a very large scale.

The organization of this paper is as follows. Section 2 formally introduces the RCC-8 constraint language, chordal graphs along with the triangulation procedure, and scale-free graphs and the model we follow to create them. In Section 3 we present algorithms for reasoning with chordal RCC-8 networks. In Section 4 we overview the state-of-the-art techniques and present our approach. In Section 5 we experimentally evaluate our approach against the state-of-the-art reasoners. In Section 6 we discuss related work, and in Section 7 we conclude and give directions for future work.

Table 1. Composition of RCC-8 relations

⋄	**DC**	**EC**	**PO**	**TPP**	**NTPP**	**TPPi**	**NTPPi**	**EQ**
DC	B	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,TPPi NTPPi	DC,EC PO,TPP TPPi,EQ	DC,EC PO,TPP NTPP	EC,PO TPP NTPP	PO,TPP NTPP	DC,EC	DC	EC
PO	DC,EC PO,TPPi NTPPi	DC,EC PO,TPPi NTPPi	B	PO,TPP NTPP	PO,TPP NTPP	DC,EC PO,TPPi NTPPi	DC,EC PO,TPPi NTPPi	PO
TPP	DC	DC,EC	DC,EC PO,TPP NTPP	TPP NTPP	NTPP	DC,EC PO,TPP TPPi,EQ	DC,EC PO,TPPi NTPPi	TPP
NTPP	DC	DC	DC,EC PO,TPP NTPP	NTPP	NTPP	DC,EC PO,TPP NTPP	B	NTPP
TPPi	DC,EC PO,TPPi NTPPi	EC,PO TPPi NTPPi	PO,TPPi NTPPi	PO,EQ TPP TPPi	PO,TPP NTPP	TPPi NTPPi	NTPPi	TPPi
NTPPi	DC,EC PO TPPi NTPPi	PO,TPPi NTPPi	PO,TPPi NTPPi	PO,TPPi NTPPi	PO,TPP NTPP NTPPi TPPi,EQ	NTPPi	NTPPi	NTPPi
EQ	DC	EC	PO	TPP	NTPP	TPPi	NTPPi	EQ

2. Preliminaries

In this section we formally introduce the RCC-8 constraint language, chordal graphs along with the triangulation procedure, and scale-free graphs together with the Barabási-Albert (BA) model.

2.1. *The* RCC-*8 constraint language*

A (binary) qualitative temporal or spatial constraint language [19] is based on a finite set B of *jointly exhaustive and pairwise disjoint* (JEPD) relations defined on a domain D, called the set of base relations. The base relations of set B of a particular qualitative constraint language can be used to represent definite knowledge between any two entities with respect to the given level of granularity. B contains the identity relation Id, and is closed under the converse operation ($^{-1}$). Indefinite knowledge can be specified by unions of possible base relations, and is represented by the set containing them. Hence, 2^{B} represents the total set of relations. 2^{B} is equipped with the usual set-theoretic operations (union and intersection), the converse operation, and the weak composition operation denoted by $\diamond$ [19]. In the case of the qualitative spatial constraint language RCC-8 [4], as already mentioned in Section 1, the set of base relations is the set $\{DC,EC,PO,TPP,NTPP,TPPi,NTPPi,EQ\}$, with EQ being the identity relation (Figure 1).

Definition 2.1. A RCC-8 network comprises a pair (V, C) where V is a non empty

finite set of variables and C is a mapping that associates a relation $C(v, v') \in 2^{\mathsf{B}}$ to each pair (v, v') of $V \times V$. C is such that $C(v, v) \subseteq \{\text{EQ}\}$ and $C(v, v') = (C(v', v))^{-1}$ for every $v, v' \in V$.

The constraint graph of a RCC-8 network $\mathcal{N} = (V, C)$ is the graph (V, E), denoted by $G(\mathcal{N})$, for which we have that $(v, v') \in E$ iff $C(v, v') \neq \mathsf{B}$. The RCC-8 network $\mathcal{N} = (V, C)$ will be said to be trivially inconsistent if $\exists v, v' \in V$ with $C(v, v') = \emptyset$. Given two graphs G_1 and G_2, $G_1 \subseteq G_2$ will denote the fact that G_1 is a subgraph of G_2. In what follows, $C(v_i, v_j)$ will be also denoted by C_{ij}, and all RCC-8 networks will be not trivially inconsistent.

Let $\mathcal{T}$ be some topological space [20], and $\mathcal{R}(\mathcal{T})$ denote the set of all non-empty regular closed subsets in $\mathcal{T}$. The domain D of RCC-8 is the set $\mathcal{R}(\mathcal{T})$, which is infinite. As an example, a topological space can be defined using the set of real numbers $\mathbb{R}$, yielding the *one-dimensional Euclidean space*. A model for a RCC-8 network $\mathcal{N} = (V, C)$ is a structure of the form $\mathcal{M} = (\mathcal{R}(\mathcal{T}), \alpha)$, where α is a mapping that associates elements of $\mathcal{R}(\mathcal{T})$ to elements of V. The intepretation of the set of base relations B of RCC-8 in a topological space $\mathcal{T}$ is then as follows.

Definition 2.2. Let $\mathcal{N} = (V, C)$ be a RCC-8 network and $\mathcal{M} = (\mathcal{R}(\mathcal{T}), \alpha)$ a model. Then, $\forall v, v' \in V$ we have the following rules:
$EQ(v, v')$ iff $(\alpha(v) = \alpha(v'))$,
$TPP(v, v')$ iff $(\alpha(v) \subset \alpha(v') \wedge Fr(\alpha(v))^{\text{a}} \cap Fr(\alpha(v')) \neq \emptyset)$,
$TPPi(v, v')$ iff $TPP(v', v)$,
$NTPP(v, v')$ iff $(\alpha(v) \subset \alpha(v') \wedge Fr(\alpha(v)) \cap Fr(\alpha(v')) = \emptyset)$,
$NTPPi(v, v')$ iff $NTPP(v', v)$,
$DC(v, v')$ iff $(\alpha(v) \cap \alpha(v') = \emptyset)$,
$EC(v, v')$ iff $(\overset{\circ}{\alpha}(v) \cap \overset{\circ}{\alpha}(v') = \emptyset \wedge Fr(\alpha(v)) \cap Fr(\alpha(v')) \neq \emptyset)$,
$PO(v, v')$ iff $(\overset{\circ}{\alpha}(v) \cap \overset{\circ}{\alpha}(v') \neq \emptyset \wedge (\exists x \in \overset{\circ}{\alpha}(v), x \notin \alpha(v')) \wedge (\exists x \in \overset{\circ}{\alpha}(v'), x \notin \alpha(v)))$.

We say that model $\mathcal{M}$ satisfies network $\mathcal{N}$ if $\forall v, v' \in V$ the relation that holds between $\alpha(v)$ and $\alpha(v')$ in $\mathcal{T}$ belongs to $C(v, v')$.

Definition 2.3. A RCC-8 network $\mathcal{N}$ is consistent if there exists a model that satisfies it.

Checking the consistency of a RCC-8 network is $\mathcal{NP}$-complete in general [21,22]. Note that in order to represent arbitrary spatial regions, it is necessary to have a canonical model of RCC-8, i.e., a structure that allows to model any (syntactically) consistent RCC-8 network [23]. As we have defined the domain of RCC-8 based on the notion of topological space and topological space defines of course a canonical model, syntactic consistency is also semantically truthful for all random datasets considered throughout the paper. However, the real datasets considered in this paper

[a] $Fr(r) = (\bar{r} \setminus \overset{\circ}{r})$ where $\bar{r}$ and $\overset{\circ}{r}$ denote the adherence and the interior of region r in $\mathcal{T}$.

have their own domain interpretation that involves determined entities (constants or landmarks), and consistency checking with the algorithms presented in this paper might not yield semantic truth, as clearly pointed out in [24]. Nevertheless, we employ the real datasets because they present an interesting underlying structure and we can always assume a canonical model to be able to construct spatial configurations out of them. Further, a syntactic consistency check is still important as it complements the approach presented in [24] and should be carried out efficiently. Despite of $\mathcal{NP}$-completeness in the general case, reasoning can be made much more efficient with the use of large maximal tractable subclasses of RCC-8. These maximal tractable subclasses are the classes $\hat{\mathcal{H}}_8, \mathcal{C}_8$, and $\mathcal{Q}_8$ [25]. Consistency checking is then realised by a *path consistency*[b] algorithm that given a RCC-8 network $\mathcal{N} = (V, C)$ iteratively performs the following operation until a fixed point is reached: $\forall i, j, k \in V$, $C_{ij} \leftarrow C_{ij} \cap (C_{ik} \diamond C_{kj})$, where variables i, k, j form triangles that belong either to a completion [5] or a chordal completion (triangulation) [11] of the constraint graph of the input network. In the latter case, the algorithm is said to perform *partial path consistency* [26] as it does not consider a complete graph. Within the operation, weak composition of relations is aided by the weak composition table of RCC-8 [27], that is presented in Table 1. If $C_{ij} = \emptyset$ for a pair (i, j) then $\mathcal{N}$ is inconsistent, otherwise the (possibly) modified network $\mathcal{N}$ is *path consistent*. If the relations of the input RCC-8 network belong to some maximal tractable subclass of relations, path consistency implies consistency, otherwise a backtracking algorithm decomposes the initial relations into subrelations belonging to some maximal tractable subclass of relations spawning a branching search tree [28].

As mentioned earlier, when path consistency is enforced on the constraint graph of an input RCC-8 network, we refer to it as *partial path consistency*. In our case, and throughout this paper, we enforce path consistency on the *chordal* constraint graph of a given RCC-8 network, thus, whenever we use the term partial path consistency we implicitly consider chordal constraint graphs. Partial path consistency was originally introduced for finite domain CSPs in [26] and it was most recently used in the case of IA and RCC-8 networks by Chmeiss et al. [29] and Sioutis et al. [11] respectively.

2.2. *Chordal graphs and Triangulation*

We begin by introducing the definition of a chordal graph. The interested reader may find more results regarding chordal graphs, and graph theory in general, in [30].

Definition 2.4. **([30])** Let $G = (V, E)$ be an undirected graph. G is *chordal* or *triangulated* if every cycle of length greater than 3 has a chord, which is an edge connecting two non-adjacent nodes of the cycle.

[b]The literature suggests the term *algebraic closure* [19] instead, which is functionally equivalent to a path consistency algorithm where the weak composition operator $\diamond$ is used [19], so we will use this more traditional term throughout the paper.

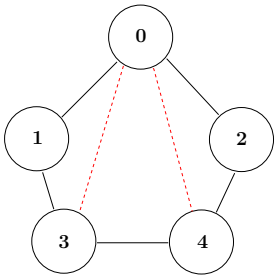


Fig. 2. Example of a chordal graph

The graph shown in Figure 2 consists of a cycle which is formed by five solid edges and two dashed edges that correspond to its chords. As for this part, the graph is chordal. However, removing one dashed edge would result in a non-chordal graph. Indeed, the other dashed edge with three solid edges would form a cycle of length four with no chords. Chordality checking can be done in $O(|V| + |E|)$ time for a given graph $G = (V, E)$ with the *maximum cardinality search* (MCS) algorithm which also constructs an elimination ordering α as a byproduct [31, 32]. In particular, MCS visits the vertices of a graph in an order such that at any point, a vertex is visited that has the largest number of visited neighbours. If a graph is not chordal, it can be made so by the addition of a set of new edges, called *fill edges*. This process is called the *elimination game* and can run as fast as in $O(|V| + (|E \bigcup F(\alpha)|))$ time for a given graph $G = (V, E)$, where $F(\alpha)$ is the set of fill edges that result by following the elimination ordering α, eliminating the nodes one by one, and connecting all nodes in the neighborhood of each eliminated node, thus, making it simplicial in the elimination graph [33]. The resulting filled graph $G' = (V, E \bigcup F(\alpha))$ is a *triangulation* of G [34]. A triangulation algorithm that makes use of the aforementioned methods is presented in Algorithm 1, while the methods themselves are well presented in [32].

Algorithm 1: Triangulation(G)

in : A graph $G = (V, E)$.
out : A triangulated graph $G' = (V, E')$ with $E \subseteq E'$.

```
1 begin
2     α ← MaximumCardinalitySearch(G);
3     F(α) ← EliminationGame(G, α);
4     G' = (V, E ⋃ F(α));
5     return G';
6 end
```

If the graph is already chordal, following an elimination ordering α produced by the maximum cardinality search algorithm means that no fill edges are added, i.e.,

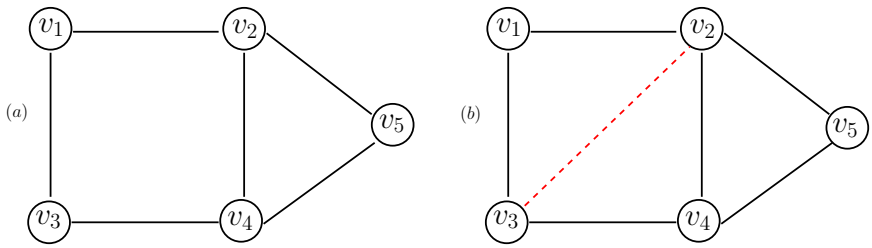


Fig. 3. Triangulation of the underlying constraint graph of a RCC-8 network

α is actually a *perfect elimination ordering* [31]. For example, a perfect elimination ordering for the chordal graph shown in Figure 2 would be the ordering $1 \rightarrow 3 \rightarrow 4 \rightarrow 2 \rightarrow 0$ of its set of nodes. In general, it is desirable to achieve chordality with as few fill edges as possible. However, obtaining an optimum graph triangulation is known to be $\mathcal{NP}$-hard [35]. In a RCC-8 network fill edges correspond to the universal relation B, i.e., the non-restrictive relation[c] that contains all base relations. As such, the *chordal constraint graph* of a given RCC-8 network is exactly its constraint graph augmented with constraints corresponding to relation B to make it chordal.

Chordal graphs become relevant in the context of qualitative spatial reasoning due to the following result obtained in [11], and later reformulated and used in [36], that states that path consistency enforced on the chordal constraint graph of an input network can yield consistency of the input network:

Proposition 2.1. *([11, 36]) For a given* RCC-*8 network $N = (V, C)$ with relations from one of the maximal tractable subclasses $\hat{\mathcal{H}}_8, \mathcal{C}_8$, and $\mathcal{Q}_8$ and for $G = (V, E)$ a chordal graph such that $G(\mathcal{N}) \subseteq G$, if $\forall (i, j), (i, k), (j, k) \in E$ we have that $C_{ij} \subseteq C_{ik} \diamond C_{kj}$, then N is consistent.*

An example of Proposition 2.1 is shown in Figure 3. The constraint graph of the initial network $\mathcal{N}$ is shown in (a). We triangulate this graph by adding edge (v_2, v_3) that is depicted as a dashed line in (b). A perfect elimination ordering for the obtained chordal graph would be the ordering $v_1 \rightarrow v_3 \rightarrow v_2 \rightarrow v_4 \rightarrow v_5$ of its set of nodes. By enforcing path consistency on the chordal constraint graph we can check the consistency of network $\mathcal{N}$ considering only 3 triangles of relations, viz., (v_1, v_2, v_3), (v_2, v_3, v_4), and (v_2, v_4, v_5). If we had opted to complete the constraint graph and obtain the corresponding complete graph of order 5, enforcing path consistency would result in considering a total of 10 triangles of relations. In general, the number of cycles of length 3, viz. triangles, in a complete graph G of order n, denoted by $c_3(G)$, is given by the following mathematical expression: $c_3(G) = n!/(6(n-3)!)$.

Triangulations work particularly well on sparse graphs with clustering properties, such as scale-free graphs. We are about to experimentally verify this in Sec-

[c]The result of the weak composition of any relation with relation B is relation B.

tion 5.

2.3. *Scale-free graphs*

We provide the following simple definition of a scale-free graph and elaborate on the details:

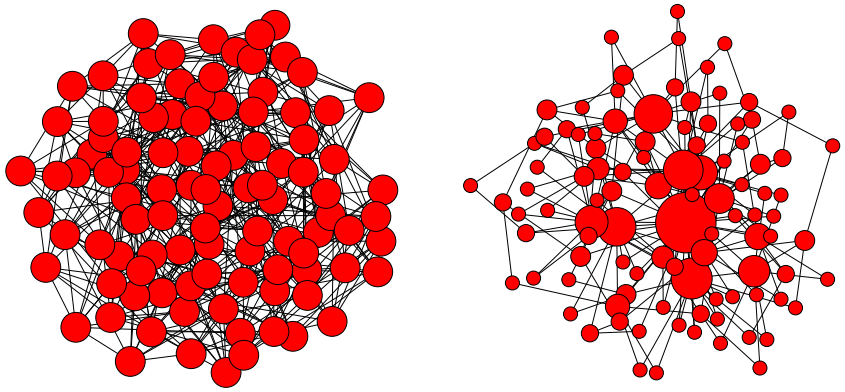
Definition 2.5. **([16])** Graphs with a power law tail in their node degree distribution are called *scale-free graphs*.

The degree of a node in a graph is the number of connections it has to other nodes (or the number of links adjacent to it) and the degree distribution $P(k)$ is the probability distribution of these degrees over the whole graph, i.e, $P(k)$ is defined to be the fraction of nodes in the network with degree k. Thus, if there are n number of nodes in total in a graph and n_k of them have degree k, we have that $P(k) = n_k/n$. For scale-free graphs the degree distribution $P(k)$ follows a power law which can be expressed mathematically as $P(k) \sim k^{-\gamma}$, where $2 < \gamma < 3$, although γ can lie marginally outside these bounds [37].

Algorithm 2: BA-Graph(n, m)

```
   in      : The number of nodes n, and the number of edges to attach from a new
             node to existing nodes m.
   out     : A scale-free graph G.
 1 begin
 2     if m < 1 or m ≥ n then
 3         raise RuntimeError, Invalid operation;
 4     V ← {0, ..., m − 1}; E ← ∅;
 5     targets ← V;
 6     repeatedNodes ← list([]);
 7     source ← m;
 8     while source < n do
 9         V ← V ⋃{source}; E ← E ⋃{(source, i) | i ∈ targets };
10         foreach i ∈ targets do
11             repeatedNodes.append(i);
12             repeatedNodes.append(source);
13         targets ← ∅;
14         while |targets| < m do
15             x ← random.choice(repeatedNodes);
16             targets.add(x);
17         source ← source + 1;
18     return G = (V, E);
19 end
```

There are several models to create random scale-free graphs that rely on *growth* and *preferential attachment* [38]. Growth denotes the increase in the number of



(a) regular graph (b) scale-free graph

Fig. 4. Structures of a random regular graph with an average degree $k = 9$ and a scale-free graph with a preferential attachment $m = 2$, both having 100 nodes

nodes in the graph over time. Preferential attachment refers to the fact that new nodes tend to connect to existing nodes of large degree which means that the more connected a node is, the more likely it is to receive new links. In real case scenarios, nodes with a higher degree have stronger ability to grab links added to the network. In a topological perspective, if we consider the example of Greece that we described in Section 1, Greece would be the higher degree node that would relate topologically to new regions (e.g., Imbros) in a deterministic and natural manner. In mathematical terms, preferential attachment means that the probability that an existing node i with degree k_i acquires a link with a new node is $p(k_i) \sim \frac{k_i}{\sum_j k_j}$. Among the different models to create random scale-free graphs, the Barabási-Albert (BA) model is the most well-studied and widely known one [16, 39]. The BA model considers growth and preferential attachment as follows. Regarding growth, it starts with an initial number m_0 of connected nodes and at each following step it adds a new node with $m \leq m_0$ edges that link the new node to m different existing nodes in the graph. When choosing the m different existing nodes to which the new node is linked, the BA model assumes that the probability p that the new node will be connected to node i depends on the degree k_i of node i with a value given by the expression $p = \frac{k_i}{\sum_j k_j}$, which is the preferential attachment that we mentioned earlier. The degree distribution resulting from the BA model is a power law of the form $P(k) \sim k^{-3}$, thus, it is able to create a subset of all the scale-free graphs that are characterised by a value γ such that $2 < \gamma < 3$. The scaling exponent is independent of m, the only parameter in the model (other than the order of the graph one would like to obtain of course). An algorithm that constructs a scale-free graph according to the BA model is presented in Algorithm 2.

Scale-free graphs are particularly attractive for our approach because they have the following characteristics:

- scale-free graphs are very sparse [15, 40], and
- scale-free graphs have a clustering coefficient distribution that also follows a power law, which implies that many low-degree nodes are clustered together forming very dense subgraphs that are connected to each other through major hubs [41].

Due to the aforementioned characteristics, scale-free graphs present themselves as excellent candidates for triangulation, as sparseness keeps time complexity for triangulation low and chordal graphs also exhibit a clustering structure [30], thus, they are able to fit scale-free graphs quite effectively. As an illustration of scale-free graphs, Figure 4 depicts a random regular graph, such as the ones used for experimental evaluation in the field of qualitative spatial reasoning, and a random scale-free graph generated using the BA model. Notice that the bigger the node is, the higher its degree is (these are the hubs).

3. Algorithms for reasoning with RCC-8 networks

In this section, we present algorithms to decide the consistency problem of a RCC-8 network by performing consistency checking restricted to triangles of a triangulation of the constraint graph corresponding to the input network. By Definition 2.1, a RCC-8 network is always a complete network. In what follows we will often use the term *chordal* RCC-*8 network* to refer to the network that would result by keeping only the constraints of the initial RCC-8 network that correspond to the edges of a triangulation of its constraint graph. This term will be particularly useful when visualizing RCC-8 networks.

3.1. *Partial Path Consistency*

From Proposition 2.1 in Section 2.2, we can have the following partial path consistency algorithm that applies path consistency on the chordal constraint graph of an input RCC-8 network. We name this algorithm PPC, presented in Algorithm 3.

Function PPC takes as parameters a RCC-8 network $\mathcal{N} = (V, C)$ and a chordal constraint graph $G = (V, E)$, which is obtained by triangulating the constraint graph corresponding to $\mathcal{N}$, namely, $G(\mathcal{N})$. The objective of PPC is to enforce path consistency on all triangles of relations in G. From Proposition 2.1 it follows that PPC decides the consistency of RCC-8 network $\mathcal{N}$, when $\mathcal{N}$ contains relations from one of the maximal tractable subclasses $\hat{\mathcal{H}}_8, \mathcal{C}_8$, and $\mathcal{Q}_8$ of RCC-8. The output of PPC is False if network $\mathcal{N}$ results in a trivial inconsistency (contains the empty relation), and True if the modified network $\mathcal{N}$ is partially path consistent and not trivially inconsistent.

If δ denotes the maximum degree of a vertex of G, we have that for each arc (i, j) selected at line 6, there are at most δ vertices of G corresponding to index k such

Algorithm 3: PPC($\mathcal{N}$, G)

in : A RCC-8 network $\mathcal{N} = (V, C)$ and a chordal graph $G = (V, E)$ such that $G(\mathcal{N}) \subseteq G$.
out : False if network $\mathcal{N}$ results in a trivial inconsistency (contains the empty relation), True if the (possibly) modified network $\mathcal{N}$ is partially path consistent.

```
1  begin
2      if ∃(i, j) ∈ E(G(N)) with C_ij = ∅ then
3          return False;
4      Q ← {(i, j) | (i, j) ∈ E};
5      while Q ≠ ∅ do
6          (i, j) ← Q.pop();
7          foreach k such that (i, k), (k, j) ∈ E do
8              t ← C_ik ∩ (C_ij ⋄ C_jk);
9              if t ≠ C_ik then
10                 if t = ∅ then
11                     return False;
12                 C_ik ← t; C_ki ← t^-1;
13                 Q ← Q ∪ {(i, k)};
14             t ← C_kj ∩ (C_ki ⋄ C_ij);
15             if t ≠ C_kj then
16                 if t = ∅ then
17                     return False;
18                 C_kj ← t; C_jk ← t^-1;
19                 Q ← Q ∪ {(k, j)};
20     return True;
21 end
```

that vertices i, j, k form a triangle. Additionaly, there exist $|E|$ arcs in the network and one can remove at most $|\mathsf{B}|$ values from any relation that corresponds to an arc, where B refers to the set of base relations of RCC-8 (the universal relation). As a result, the time and space complexity of PPC is $O(\delta \cdot |E| \cdot |\mathsf{B}|)$ and $O(|E|)$ respectively.

3.2. *Consistency*

For the general case of RCC-8 networks, i.e., when all RCC-8 relations are considered and not only the ones belonging to maximal tractable subclasses $\hat{\mathcal{H}}_8, \mathcal{C}_8$, and $\mathcal{Q}_8$ of RCC-8, we have the following backtracking algorithm, which we call PartialConsistency, presented in Algorithm 4.

PartialConsistency receives as input a RCC-8 network $\mathcal{N} = (V, C)$, a chordal constraint graph $G = (V, E)$, which is obtained by triangulating the constraint graph corresponding to $\mathcal{N}$, namely, $G(\mathcal{N})$, and a maximal tractable subclass of relations $\mathcal{S}$, which can be any of the classes $\hat{\mathcal{H}}_8, \mathcal{C}_8$, and $\mathcal{Q}_8$ [25]. Then, PartialConsistency

Algorithm 4: PartialConsistency($\mathcal{N}$, G, $\mathcal{S}$)

in : A RCC-8 network $\mathcal{N} = (V, C)$, a chordal graph $G = (V, E)$ such that $G(\mathcal{N}) \subseteq G$, and a maximal tractable subclass $\mathcal{S}$.
out : Null if network $\mathcal{N}$ is inconsistent, a (possibly) refined constraint network $\mathcal{N}$ if $\mathcal{N}$ is consistent.

```
1  begin
2      if ! PPC(N, G) then
3          return Null;
4      if ∀(i, j) ∈ E we have that C_ij ∈ S then
5          return N;
6      choose constraint C_ij such that C_ij ∉ S;
7      split C_ij into r_1, ..., r_k ∈ S: r_1 ∪ ... ∪ r_k = C_ij;
8      foreach r ∈ {r_l | 1 ≤ l ≤ k} do
9          replace C_ij with r in N;
10         result ← PartialConsistency(N, G, S);
11         if result ≠ Null then
12             return result;
13     return Null;
14 end
```

splits a relation C_{ij} into relations that belong to the maximal tractable subclass of relations $\mathcal{S}$ (line 7). After that, each of these relations is instantiated accordingly to the constraint network $\mathcal{N}$ (line 9) and the PPC algorithm is reapplied. Notice, however, that except for the first step, the PPC algorithm only has to be run for the paths that are possibly affected by each prior instantiation, as explained in [5, 7], which takes $O(|E| \cdot |\mathsf{B}|)$ intersections and compositions. This detail is not included in the PartialConsistency algorithm.

We also present an additional algorithm, which is the iterative counterpart of the recursive chronological backtracking algorithm. The recursive and iterative algorithms are functionally equivalent. However, the iterative algorithm does not suffer from a recursion depth or a recursion stack limit, which is the case for recursion in many languages, such as C, C++, Python, and Java. Further, its structure, although a little more complex, makes it easier to fine-tune it, performance-wise. We call the iterative algorithm IterativePartialConsistency, presented in Algorithm 5.

The else branch in line 17 belongs to the while branch in line 13 and is only executed if the while loop does not break. In this particular case, if the break command in line 16 is not executed, it means that we were not able to find any more values to assign to a variable and we exhausted our stack. The backtracking algorithms are presented in a very general, but, nevertheless, compact manner. Heuristics regarding variable and value selection can be easily implemented without perturbing the structure of our algorithms. We then have the following theorem:

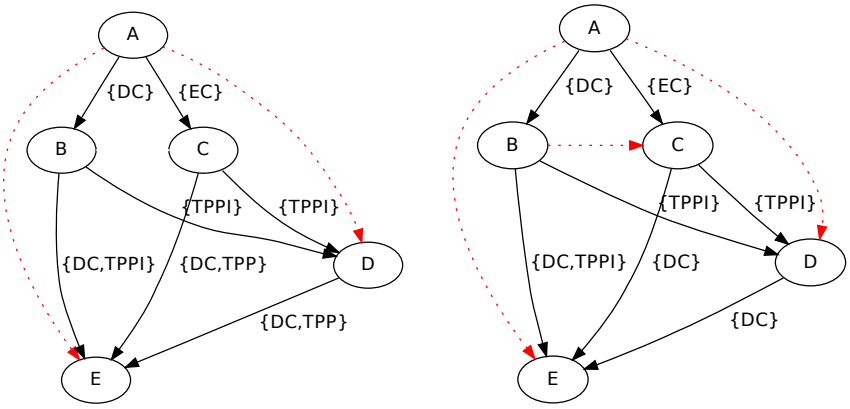
Theorem 3.1. ([11, 36]) For a given RCC-8 network $\mathcal{N} = (V, C)$, $G = (V, E)$

Algorithm 5: IterativePartialConsistency($\mathcal{N}$, G, $\mathcal{S}$)

```
in  : A RCC-8 network N = (V, C), a chordal graph G = (V, E) such that
      G(N) ⊆ G, and a maximal tractable subclass S.
out : Null if network N is inconsistent, a (possibly) refined constraint network
      N if N is consistent.
1  begin
2      Stack ← [ ] // Initialize stack;
3      if ! PPC(N, G) then
4          return Null;
5      while True do
6          if ∀(i, j) ∈ E we have that C_ij ∈ S then
7              return N;
8          choose constraint C_ij such that C_ij ∉ S;
9          split C_ij into r_1, ..., r_k ∈ S: r_1 ∪ ... ∪ r_k = C_ij;
10         r_values ← {r_l | 1 ≤ l ≤ k};
11         while True do
12             if ! r_values then
13                 while Stack do
14                     N, r_values ← Stack.pop();
15                     if r_values then
16                         break;
17                 else
18                     return Null;
19             r ← r_values.pop();
20             N' ← N;
21             replace C_ij with r in N;
22             if PPC(N, G) then
23                 break;
24             N ← N';
25         Stack.append(N', r_values);
26     raise RuntimeError, Cannot happen;
27 end
```

a chordal graph such that $G(\mathcal{N}) \subseteq G$, and $\mathcal{S}$ one of the maximal tractable subclasses $\hat{\mathcal{H}}_8, \mathcal{C}_8$, and $\mathcal{Q}_8$ of RCC-8, functions PartialConsistency($\mathcal{N}$, G, $\mathcal{S}$) and IterativePartialConsistency($\mathcal{N}$, G, $\mathcal{S}$) decide the consistency of network $\mathcal{N}$.

Given a network $\mathcal{N}$, its chordal constraint graph $G = (V, E)$, and a subclass $\mathcal{S}$, the search space for both PartialConsistency and IterativePartialConsistency is $O(\omega^{|E|})$, where ω is the branching factor provided by subclass $\mathcal{S}$ (e.g., $\omega = 1.4375$ for subclass $\hat{\mathcal{H}}_8$ of RCC-8 [5]). Again, we note that PartialConsistency and IterativePartialConsistency are functionally equivalent. The advantage of IterativePartialConsistency over PartialConsistency relies on the fact that it does not suffer from a recursion depth or a recursion stack limit and it avoids the overhead of



(a) chordal network (b) complete network

Fig. 5. Pruning capacity of PC on a chordal constraint network and its completion

costly recursive calls and call stack management altogether. In particular, we have experimentally verified that the performance of IterativePartialConsistency is in general 5% to 10% better than the performance of PartialConsistency for the datasets considered in this paper.

3.3. *Pruning capacity of* PPC

Given Proposition 2.1 presented in Section 2.2 and the partial path consistency algorithm presented in this section, the question arises whether more pruning can be obtained by performing path consistency on the complete graph of an input network.

Proposition 3.1. *For a given* RCC-*8 network* $\mathcal{N}$ *comprising relations only from one of the maximal tractable subclasses* $\hat{\mathcal{H}}_8, \mathcal{C}_8$, *and* $\mathcal{Q}_8$ *of* RCC-*8, we have that path consistency enforced on the complete graph of that network is able to prune more relations than path consistency enforced on a triangulation of the constraint graph of that network, when considering the common edges between the complete and the chordal graph.*

Proof. In Figure 5 we present an example of the pruning capacity of enforcing path consistency on a chordal constraint network and the pruning capacity of enforcing path consistency on the complete constraint network. For our example we use relations that are contained in all three maximal tractable subclasses $\hat{\mathcal{H}}_8, \mathcal{C}_8$, and $\mathcal{Q}_8$ of RCC-8. The chordal constraint network in Figure 5a is path consistent with respect to its corresponding chordal graph. However, the addition of edge (B, C) that completes the network, shown in Figure 5b, results in the pruning of base relation TPP on edges (C, E) and (D, E). □

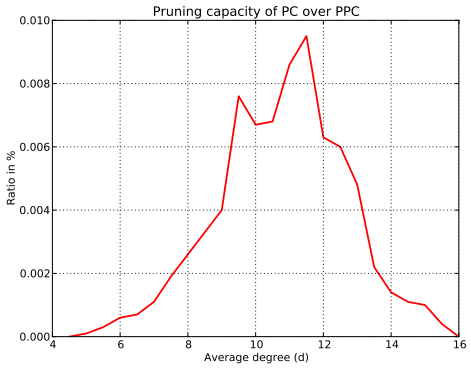


Fig. 6. Pruning capacity of path consistency over partial path consistency

Thus, in the case of maximal tractable subclasses $\hat{\mathcal{H}}_8, \mathcal{C}_8$, and $\mathcal{Q}_8$ of RCC-8 we cannot have a result similar to the one by Bliek and Sam-Haroud that consider convex CSPs and show that the pruning capacity of path consistency on a chordal constraint graph of an input network is equivalent to the pruning capacity of path consistency on the complete graph of that network when considering the common edges [26].

We tried to find out approximately how much more pruning we can obtain in the common edges when opting to apply path consistency on the complete graph rather than on a chordal constraint graph of a given network. Therefore, we considered networks of 100 nodes based on model $\mathsf{A}(\mathsf{n},\mathsf{d},\mathsf{l})$ [5], with $l = 4$ $(= |\mathsf{B}|/2)$ relations per edge, and an average degree d between 4.5 and 16.0 with a 0.5 step. For each degree we created 10000 networks. Regarding model $\mathsf{A}(\mathsf{n},\mathsf{d},\mathsf{l})$, network instances with an average degree d between values 8.0 and 11.0 lie within the *phase transition* region, where it is equally possible for networks to be consistent or inconsistent, and, thus, are harder to solve [5]. We expect more pruning to occur in that particular region for path consistency instead of partial path consistency, as more reasoning is required to decide the path consistency of a network. For each network we count the number of base relations that correspond to the common edges between a triangulation and the completion of its constraint graph, after path consistency has been applied on each respective graph. Then, we calculate a ratio ρ, which indicates how many more base relations were pruned by path consistency in comparison with partial path consistency.

The results of our experiment are shown in Figure 6. Indeed, we were right on our expectation, as one can clearly see that the biggest ratios lie within the phase transition region, with the small exception of degree 11.5, that lies marginally outside 8.0 and 11.0.[d] In a sense, we obtain similar experimental results with those

[d]Some small variations in the graph are normal and expected artifacts of random instance generation.

of Bliek and Sam-Haroud in [26], for a different kind of CSPs, viz., qualitative constraint networks over infinite domains. Path consistency is able to prune more relations than partial path consistency, but only in a very small percentage. In the best case, the ratio is $\sim$ %0.009, which translates to one more base relation being pruned in every 10000.

4. Overview of our practical approach

In this section we describe our own implementations of generic qualitative reasoners that build on state-of-the-art techniques, and consequently our practical approach of choice for tackling large scale-free-like RCC-8 networks.

Phalanx. We have implemented Phalanx in Python that is the generalized and code refactored version of PyRCC8 [11]. Phalanx supports small arbitrary binary constraint calculi developed for spatial and temporal reasoning that are relation algebras sharing common algebraic properties with RCC-8 (e.g., involution of converse implied by strong converse [42]), such as RCC-5 [4] and Allen's interval algebra (IA) [43], in a way similar to GQR [7]. Further, Phalanx presents significant improvements over PyRCC8 regarding scalability and speed. In particular, the new reasoner handles the constraint matrix that represents a qualitative network more efficiently during backtracking search, i.e., it does not create a copy of the matrix at each forward step of the backtracking algorithm (as is the case with Renz's solver [5] or PyRCC8), but it only keeps track of the values that are altered at each forward step to be able to reconstruct the matrix in the case of backtracking. A similar mechanism is also used to keep track of unassigned variables (i.e., relations that do not belong to maximal tractable subclasses of relations and are decomposed to subrelations at each forward step of the backtracking algorithm) that may dynamically change in number due to the appliance of path consistency at each forward step of the backtracking algorithm. For example, given a non-tractable RCC-8 network and a maximal tractable subclass $\mathcal{S}$, the path consistency algorithm can prune a relation that belongs to subclass $\mathcal{S}$ into an untractable relation, and vice versa.[e] This allows us to apply the heuristics that deal with the selection of the next unassigned variable faster, as we keep our set of unassigned variables minimal and we do not have to go over the whole set of variables each time to find the ones that are unassigned. The path consistency algorithm implementation has been also modified to better handle the cases where weak composition of relations leads to the universal relation. In these cases we can continue the iterative operation of the path consistency algorithm since the intersection of the universal relation with any other relation leaves the latter relation intact. Finally, there is also a weight over learned weights dynamic heuristic for variable selection, as in the latest version of GQR,

[e]Note that in the case where the RCC-8 network comprises relations only from the maximal tractable subclass $\mathcal{S}$ (i.e., it is tractable), this would not be possible as $\mathcal{S}$ is closed under the path consistency operations.

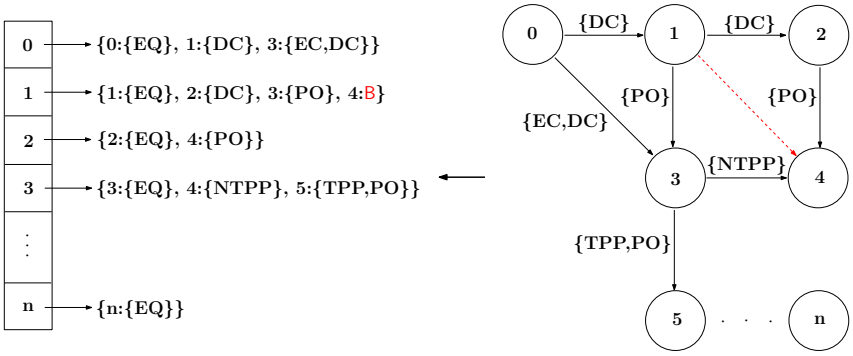


Fig. 7. Hash table based adjacency list for representing a chordal RCC-8 network

under release 1500[f].

Phalanx$\triangledown$. We have implemented Phalanx$\triangledown$ in Python that is the generalized and code refactored version of PyRCC8$\triangledown$ [11]. Phalanx$\triangledown$ is essentially Phalanx where the path consistency algorithm is replaced with a partial path consistency algorithm, as described in Section 3. Phalanx$\triangledown$ supports small arbitrary binary constraint calculi developed for spatial and temporal reasoning that are relation algebras sharing common algebraic properties with RCC-8 (e.g., involution of converse implied by strong converse [42]), and for which Proposition 2.1 holds, such as RCC-5 [4] and Allen's interval algebra (IA) [43]. Regarding triangulation, Phalanx$\triangledown$ is coupled with the implementation of the maximum cardinality search algorithm and a fast fill in procedure (as presented in Algorithm 1 in Section 2.2), as opposed to the heuristic based, but rather naive, triangulation procedure implemented in [11]. Though the maximum cardinality search algorithm does not yield minimal triangulations if the constraint graph of the input network is not chordal, it does guarantee that no fill edges are inserted if the graph is indeed chordal, as explained in Section 2.2. In addition, even for the non-chordal cases we obtain good results with this approach and have a fine trade-off between time efficiency and good triangulations. As with the case of Phalanx and PyRCC8, Phalanx$\triangledown$ presents significant improvements over PyRCC8$\triangledown$ regarding scalability and speed, by incorporating all the state-of-the-art techniques that apply to Phalanx and were mentioned earlier.

Sarissa. We improve the state-of-the-art techniques for tackling large scale-free-like RCC-8 networks by opting for a hash table based adjacency list to represent and reason with the chordal completion of the constraint graph of an input network. The variables of the input network (or the nodes) are represented by index numbers of a list, and each variable (or node) is associated with a hash table that stores

[f] `http://sfbtr8.informatik.uni-freiburg.de/R4LogoSpace/downloads/gqr-1500.tar.bz2`

Table 2. Triangulation time based on different methods

triangulation method	**BA-Graph(10000, 2)**
MCS	2.26s
MD	112.004s
GFI	–

key-value pairs of variables and relations. Figure 7 shows how an example chordal RCC-8 network is represented by our hash table based adjacency list approach. Self-loops (of the identity relation EQ) have been omitted from the network. The dashed edge $(1, 4)$ corresponds to a fill edge that results after triangulating the initial non-chordal graph consisting of solid edges. This fill edge is stored in the hash table based adjacency list as the universal relation B. For a given RCC-8 network $N = (V, C)$ and for $G = (V, E)$ its chordal constraint graph, our approach requires $O(|V| + |E| \cdot b)$ memory, where b is the size needed to represent a relation from the set of relations 2^{B} of RCC-8. Having a constraint matrix to represent the input RCC-8 network (that is typically used by the state-of-the-art reasoners Phalanx and Phalanx$_{\bigtriangledown}$ presented earlier), results in a $O(|V|^2 \cdot b)$ memory requirement, even if chordal graphs are used leaving a big part of the matrix empty, as is the case with Phalanx$_{\bigtriangledown}$, or Sparrow for IA [29]. Further, we still retain an $O(1)$ average access and update time complexity which becomes $O(\delta)$ in the amortized worst case, where δ is the maximum vertex degree of the chordal constraint graph that corresponds to the input network. Given that we target large scale-free-like, and, thus, sparse networks, this only incures a small penalty for the related experiments performed. The partial path consistency implementation also benefits from this approach as the queue data structure on which it is based has to use only $O(|E|)$ of memory to store the relations compared to the $O(|V|^2)$ memory requirement of Phalanx, GQR, and Renz's solver. The triangulation algorithm is based on Algorithm 1, as with Phalanx$_{\bigtriangledown}$. An alternative would be to use some special greedy heuristic rather than the maximum cardinality search (MCS) algorithm to obtain an elimination ordering, the simplest and fastest of which being the *minimum degree* (MD) heuristic [44]. As noted in [44], this heuristic has a time complexity of $O(|V| \cdot |E|)$ for a given graph $G = (V, E)$, which is an overkill for the large networks that are of our interest in this paper. Another choice would be the (minimum) greedy fill-in (GFI) heuristic (used in [29, 36]) with a time complexity of $O(|V|^3)$ [45, 46], which again marks it as prohibitive in our case. Nevertheless, we present in Table 2 the time needed on average to calculate a triangulation based on the aforementioned methods per graph of a small dataset of 10 networks created by the BA-Graph(10000, 2) generation model (Algorithm 2). The experiment was carried out on the same machine that we perform our experimental evaluation in Section 5 and shows that even for the simplest MD heuristic, the triangulation operation alone for a given graph, takes more time than triangulating the same type of graph with MCS and solving the network corresponding to that graph with a reasoner, as observed in Fig-

Table 3. Tabular overview of our reasoners

reasoner	data structure	core algorithm
Phalanx	adjacency matrix	path consistency
Phalanx$\bigtriangledown$	adjacency matrix	partial path consistency
Sarissa	hash table based adjacency list	partial path consistency

ure 9a for random network instances with constraint graphs of the same size and in Figure 13 for real network instances with constraint graphs of much bigger size.[g] Note that GFI was not able to compute a triangulation within the 5-minute timeout it was given. Closing our parenthesis on the different triangulations we explored, the techniques that we mentioned prior to our small comparison are implemented under the hood of our new reasoner which is called Sarissa. Sarissa, as with our other tools presented here, is a generic and open source qualitative reasoner written in Python[h], a general-purpose, interpreted high-level programming language which enables rapid application development. In the experimens to follow, Sarissa will be shown to greatly outperform state-of-the-art reasoners, such as GQR, for large RCC-8 networks of scale-free-like structure. However, in defense of GQR which was found to perform poorly in [11] under release 1418, we state that the latest version of GQR has undergone massive scalability improvements and is currently the most complete and fastest reasoner for handling reasonably scaled random *regular* qualitative networks. At this point, we can also claim that Renz's solver [5] has been fairly outdated, as it will become apparent in the experiments that we employ it for. Finally, our hash table based adjacency list allows extending a network with new nodes and edges in constant time. For the case of tractable RCC-8 networks and due to the notion of *canonical solutions* introduced in [47], we have already presented a vertex incremental variant of the partial path consistency algorithm for building a consistent RCC-8 enriched database incrementally [48]. As future work we plan to also have this functionality for non-tractable RCC-8 networks.

A tabular overview of the reasoners presented in this section is given in Table 3. All tools used in this paper are open-source and under a free license, and comprise a new set of qualitative reasoners that build on state-of-the-art Artificial Intelligence techniques. They can be acquired along with the datasets used in this paper upon request from the authors and also found online in the following address: `http://www.cril.fr/~sioutis/work.php`.

5. Experimental evaluation

In this section we compare the performance of Sarissa with that of Renz's solver [5], GQR (release 1500) [7], Phalanx, and Phalanx$\bigtriangledown$, with their best performing heuristics enabled.

[g] We used the LibTW library for MD and GFI (`http://treewidth.com/`).
[h] http://python.org/

Algorithm 6: $\mathsf{BA}(n, m, \mathcal{S} = \emptyset)$

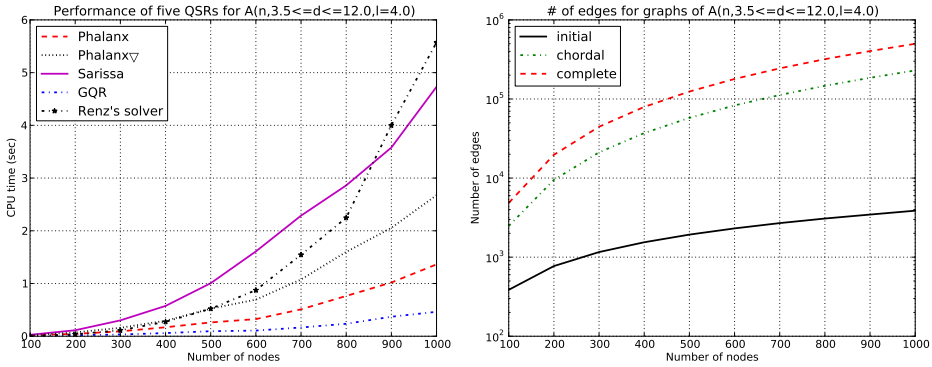
in : The number of nodes n, the number of edges to attach from a new node to existing nodes m, and an optional subclass $\mathcal{S}$.
out : A scale-free network $\mathcal{N} = (V, C)$.

```
1 begin
2     G ← BA-Graph(n, m);
3     V ← V(G);
4     C ← map({((v, v′) : (B if v ≠ v′ else {EQ})) | v, v′ ∈ V(G)});
5     foreach (v, v′) ∈ E(G) do
6         C(v, v′) ← random.choice(2^B if S = ∅ else S); C(v′, v) ← (C(v, v′))^−1;
7     return N = (V, C);
8 end
```

We considered both random and real datasets. Random datasets consist of RCC-8 networks generated by the usual $\mathsf{A}(\mathsf{n}, \mathsf{d}, \mathsf{l})$ model [5] and *large* RCC-8 networks generated by the $\mathsf{BA}(\mathsf{n}, \mathsf{m})$ model which we first introduce for benchmarking qualitative spatial reasoners in this paper. In short, model $\mathsf{A}(\mathsf{n}, \mathsf{d}, \mathsf{l})$ creates random regular networks (like the one depicted in Figure 4a) of size n, degree d, and an average number l of RCC-8 relations per edge, whereas model $\mathsf{BA}(\mathsf{n}, \mathsf{m})$ creates random scale-free-like networks (like the one depicted in Figure 4b) of size n and a preferential attachment value m. The algorithm that implements this model and generates scale-free-like RCC-8 networks is shown in Algorithm 6. For model $\mathsf{BA}(\mathsf{n}, \mathsf{m})$ the average number of base RCC-8 relations per edge defaults to $|\mathsf{B}|/2$, where B is the set of base relations of RCC-8. Real datasets consist of `admingeo` [12] and `gadm-rdf`[i] that comprise 11761/77907 nodes/edges and 276728/590865 nodes/edges respectively. In short, `admingeo` describes the administrative geography of Great Britain using RCC-8 relations, and `gadm-rdf` the world's administrative areas likewise. The experiments were carried out on a computer with an Intel Core 2 Quad Q9400 processor with a CPU frequency of 2.66 GHz, 8 GB RAM, and the Precise Pangolin x86_64 OS (Ubuntu Linux). Renz's solver and GQR were compiled with gcc/g++ 4.6.3. Sarissa, Phalanx, and Phalanx$\triangledown$ were run with PyPy[j] 1.9, which implements Python 2. Only one of the CPU cores was used for the experiments.

Random datasets. For model $\mathsf{A}(\mathsf{n}, \mathsf{d}, \mathsf{l})$ we considered network sizes between 100 and 1000 nodes with a 100 step and $l = 4$ $(= |\mathsf{B}|/2)$ relations per edge. For each size series we created 270 networks that span over a degree d between 3.5 and 12.0 with a 0.5 step, i.e., 15 network instances were generated for each degree. Regarding model $\mathsf{A}(\mathsf{n}, \mathsf{d}, \mathsf{l})$, network instances with an average degree d between values 8.0 and 11.0 lie within the *phase transition* region, where it is equally possible for networks to be consistent or inconsistent, thus, they are harder to solve [5]. The results are shown

[i] `http://gadm.geovocab.org/`
[j] `http://pypy.org/`



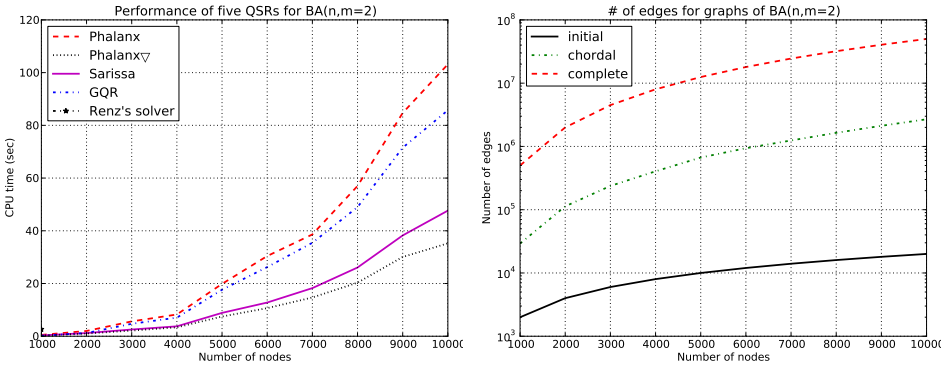
(a) performance for regular networks

(b) edges considered for different graph configurations of regular networks

Fig. 8. Experiment with random regular networks

in Figure 8a. GQR clearly outperforms all other reasoners with Phalanx coming close 2^{nd} and Renz's solver last. In the final step, when networks of 1000 nodes are considered, Renz's solver decides the networks in an average time of 5.571 sec, GQR in 0.463 sec, Phalanx in 1.363 sec, Phalanx$\triangledown$ in 2.675 sec, and Sarissa in 4.731 sec. In the particular case of Sarissa and Phalanx$\triangledown$ that use chordal graphs we note that they pay an extra cost for calculating the triangles of relations for each appliance of path consistency as these are not precomputated and stored in advance for memory efficiency [11]. Sarissa also pays an additional cost for not being able to access or update relational values in constant worst case time as it does not use a matrix. It is a fact that random regular networks are not triangulated very efficiently with our approach, which results in dense chordal graphs in most of the cases. This fact is depicted in Figure 8b where one can clearly see that the total number of edges in a triangulated constraint graph of an initial random regular network, is very close to the total number of edges in the completion of the constraint graph of that network. To be even more precise, for the random regular network instances considered, their respective chordal constraint graphs contained on average only 53% less edges than the completions of those graphs.

For model BA(n, m) we considered 30 networks for each size between 1000 and 10000 nodes with a 1000 step and a preferential attachment value of $m = 2$. Regarding model BA(n, m), and for the network sizes considered, we found that the *phase transition* region is defined for the specific value of $m = 2$. A value of $m = 1$ yields networks that do not have any triangles of relations, and, therefore, are always consistent, and a value of $m = 3$ yields almost exclusively inconsistent networks as the network size increases, that are very easy to decide. In particular, we experimented with 50 network instances of 10000 nodes and a preferential attachment value of $m = 3$, and none of them was consistent. The results for the network instances with a preferential attachment value of $m = 2$ are shown in Figure 9a. Sarissa and



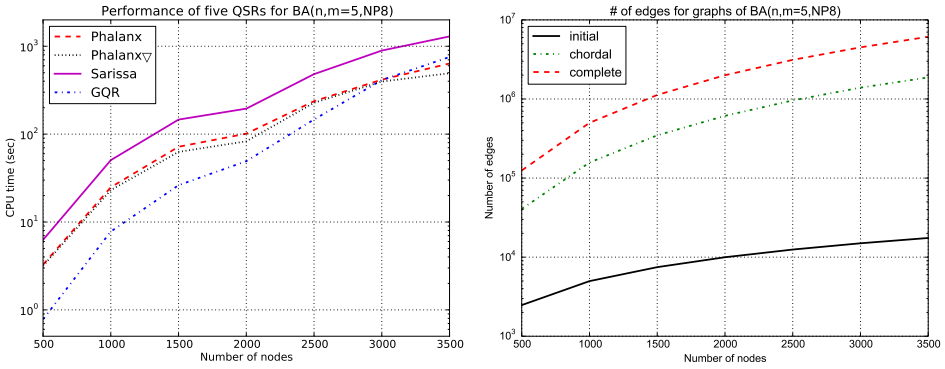
(a) performance for scale-free-like networks (b) edges considered for different graph configurations of scale-free-like networks

Fig. 9. Experiment with random scale-free-like networks

Phalanx▽ outperform all other reasoners by a large scale, and Renz's solver was able to solve only the networks of 1000 nodes (slower than all others) as it quickly hit the memory limit in our computer due to many recursive calls (leading to storing many copies of its constraint matrix). In the final step, when networks of 10000 nodes are considered, GQR decides the networks in an average time of 85.727 sec, Phalanx in 103.159 sec, Phalanx▽ in 35.241 sec, and Sarissa in 47.653 sec. Figure 9b depicts the number of edges that we consider when using a chordal completion of the constraint graph corresponding to a random scale-free-like network, rather than the (full) completion of that graph. In particular, for the random scale-free-like networks instances considered, their respective chordal constraint graphs contained on average around 95% less edges than the completions of those graphs. This is a huge improvement when compared with the 53% in the case of random regular networks. We note that Sarissa is still burdened with the additional cost of not being able to access or update relational values in constant worst case time.

We tried to push our implementations even further, and extended model BA(n, m) to generate random scale-free-like instances using the $\mathcal{NP}_8$ set of RCC-8 relations [5]. This set contains relations that do not belong to any maximal tractable subclass of RCC-8, thus, significantly increasing the search space of the problem. In particular, when using all RCC-8 relations and the maximal tractable subclass $\hat{\mathcal{H}}_8$ as a split set, the average branching factor is 1.4375. For a network instance of n nodes this translates to a $O(1.4375^{(n^2-n)/2})$ search space[k]. On the other hand, when using only $\mathcal{NP}_8$ relations, and again $\hat{\mathcal{H}}_8$ as a split set, the average branching factor

[k]In qualitative spatial reasoning, for a branching factor ω and a RCC-8 network of n nodes, the search space is $O(\omega^{(n^2-n)/2})$, where $O((n^2-n)/2)$ is the number of constraints that have to be split for the given n nodes. However, this is an overestimation, as using the path consistency algorithm as a preprocessing and forward checking step in a backtracking algorithm significantly reduces search space [49].

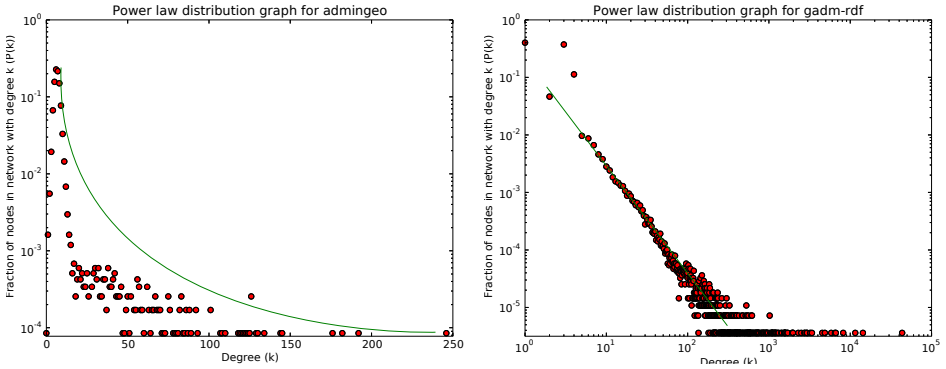


(a) performance for hard scale-free-like networks (b) edges considered for different graph configurations of hard scale-free-like networks

Fig. 10. Experiment with random hard scale-free-like networks

is 2.053, which given a network instance of n nodes translates to a $O(2.05^{(n^2-n)/2})$ search space. As such, the search space in the latter case is $O((2.053/1.434)^{(n^2-n)/2})$ bigger than in the first case, which has a serious impact on evaluation. We call our new extended model $\mathsf{BA}(\mathsf{n},\mathsf{m},\mathcal{NP}_8)$.

For model $\mathsf{BA}(\mathsf{n},\mathsf{m},\mathcal{NP}_8)$ we considered 10 networks for each size between 500 and 3500 with a 500 step and a preferential attachment value of $m = 5$. Regarding model $\mathsf{BA}(\mathsf{n},\mathsf{m},\mathcal{NP}_8)$, and for the network sizes considered, we found that the *phase transition* region is defined for the specific value of $m = 5$. In fact, for that value of m, checking the consistency of the network instances is very time consuming, and, thus, we did not push the network sizes to an even larger number. In any case, our dataset was sufficient to obtain a trend for the reasoners involved, and networks of 3500 nodes can be qualified as large, especially when $\mathcal{NP}_8$ relations are considered. The results for the hard network instances with a preferential attachment value of $m = 5$ are shown in Figure 10a. Initially we considered five reasoners, the four that are mentioned in the legend and Renz's solver. However, Renz's solver lacks the weight over learned weights dynamic heuristic for variable selection and could not keep up with the rest of the reasoners even for the smallest networks of 500 nodes. Further, it hit the memory limit. Sarissa has the worst performance of all reasoners, deciding the networks of 3500 nodes in an average time of 1294.27 sec. This was expected, since the increased search space for the hard instances leads to massively more table lookups, wearing down the hash table based adjacency list. However, the performance of Sarissa converges to the performance of GQR, which was the biggest surprise in this experiment. Due to its more advanced heuristic mechanics (restarts and nogood recording), we expected GQR to set the bar for this experiment. Judging by the experiment with random normal scale-free-like networks, network sizes of 3500 nodes were not really that of a challenge for GQR (Figure 9a). In particular, GQR starts off very well, but its performance deteriotates very fast and



(a) degree distribution for `admingeo` (b) degree distribution for `gadm-rdf`

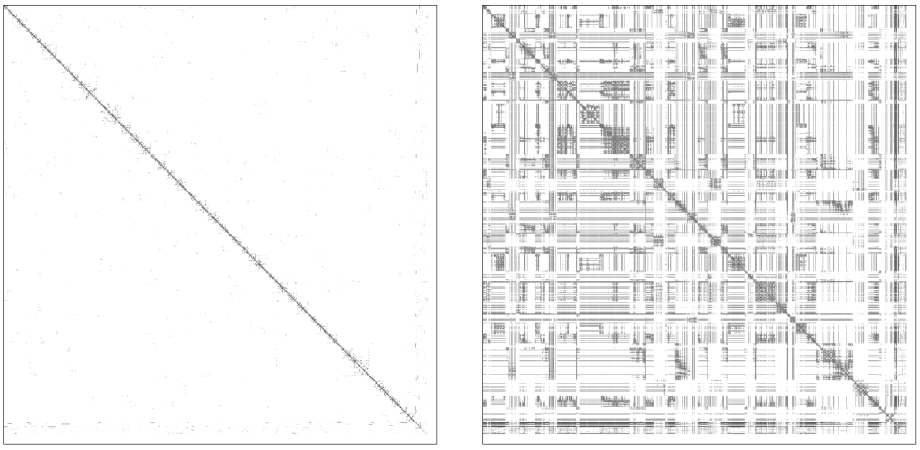
Fig. 11. Evidence of the power law node degree distribution of the real datasets considered

is outrunned by both Phalanx$_\bigtriangledown$ and Phalanx by the time when networks of 3000 nodes are considered. GQR should at least perform better than Phalanx as it did for random normal scale-free-like networks (Figure 9a), since both implementations use a matrix and GQR has more advanced heuristics.

In the final step, when networks of 3500 nodes are considered, GQR decides the networks in an average time of 754.98 sec, Phalanx in 636.32 sec, Phalanx$_\bigtriangledown$ in 493.37 sec, and as already mentioned Sarissa in 1294.27 sec. Figure 10b depicts the number of edges that we consider when using a chordal completion of the constraint graph corresponding to a random hard scale-free-like network, rather than the completion of that graph. In particular, for the random hard scale-free-like networks instances considered, their respective chordal constraint graphs contained on average around 71% less edges than the completions of those graphs.

To the best of our knowledge, the random datasets used in this paper are the biggest ones to date of all others that exist in literature.

Real datasets. For experimenting with our real datasets, viz., `admingeo` and `gadm-rdf`, we created constraint networks of different size, by taking into account a small number of relations from the initial dataset and increasing it at every next step. These real datasets comprise tractable networks. Further, they contain only up to 2 base relations per edge. Of course, for both datasets, the whole dataset was used as a final step. To backup our argument about the scale-free-like structure of real RCC-8 networks, we present Figure 11 that displays the power law degree distribution of our real networks. As `gadm-rdf` is a very large network, we display its degree distribution in log-log scale where the power law function is seen as a straight line [16]. We remind the reader that `admingeo` and `gadm-rdf` comprise 11761/77907 nodes/edges and 276728/590865 nodes/edges respectively. Triangulating the constraint graph of `admingeo` results in a total of 1961820 edges, which is 97% less than completing the graph that results in $(11761^2 - 11761)/2 =$



(a) initial constraint graph of `admingeo` (b) chordal constraint graph of `admingeo`

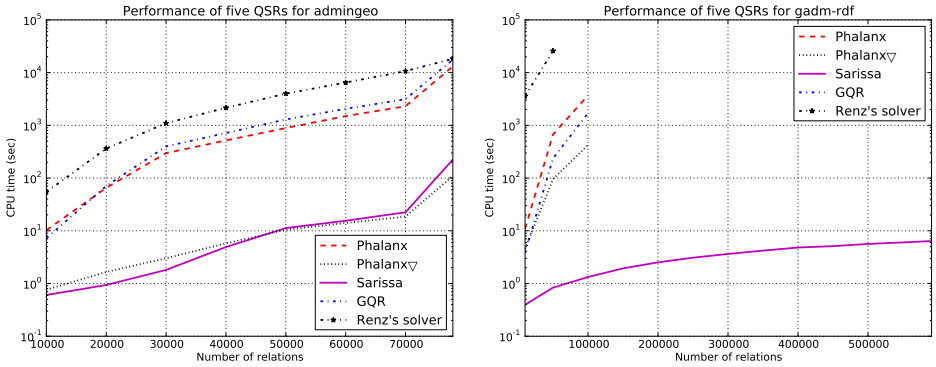
Fig. 12. Matrix representations of different graph configurations of `admingeo`

69154680 edges. Triangulating the constraint graph of `gadm-rdf` results in a total of 596574 edges, which is $\sim 99.99\%$ less than completing the graph that results in $(276728^2 - 276728)/2 = 38289054628$ edges. In fact, `admingeo` consists of some big cycles that result in the addition of plenty fill edges in its initial graph, as opposed to `gadm-rdf`.

To obtain a better understanding on the effect of triangulation, Figure 12 depicts the adjacency matrix of the initial constraint graph of `admingeo` (Figure 12a) and the matrix of the chordal constraint graph of `admingeo` (Figure 12b).[1] It is worth noting that `admingeo` exhibits a strong locality of reference property [50], i.e., adjacent nodes are close to each other with respect to their labelling and, thus, form a dense part around the main diagonal of the graph's adjacency matrix.

We proceed with the experimental evaluation. The results for `admingeo` are shown in Figure 13a. Sarissa and Phalanx$_\bigtriangledown$ have approximately equal performance and significantly outperform all other reasoners. In the final step, both reasoners run in ~ 150 sec, when the 3^{rd} best reasoner for this experiment, viz., Phalanx, runs in ~ 4 hours. Up to this point we have considered networks that fit the size of the matrix that accompanies all state-of-the-art reasoners. We proceed with `gadm-rdf`, a dataset almost 30 times bigger than `admingeo`. The results for `gadm-rdf` are

[1]We used gnuplot to create matrices of 800×800 pixels. For a given network of 8000 nodes for example, each pixel corresponds to a submatrix of $(8000/800) \times (8000/800)$ size which results by partitioning the initial matrix of $(8000) \times (8000)$ size with a $(8000/800)$ step for each row and each collumn. For a particular pixel, color white indicates the complete absense of an edge in the $(8000/800) \times (8000/800)$ submatrix, and color black a full of edges submatrix. In-between shades, such as light and dark grey, indicate, more or less, how many edges the corresponding submatrix has. For a complete network, its adjacency matrix would be completely black.



(a) performance for `admingeo` (b) performance for `gadm-rdf`

Fig. 13. Performance of five reasoners for real datasets

shown in Figure 13b. Sarissa is the only implementation that was able to reason with the whole dataset. Sarissa completes the final step of the 590865 relations in under ~ 7 sec, when the 2^{nd} best reasoner for this experiment, viz., Phalanx$\bigtriangledown$, can reason only up to 100000 relations in ~ 7 min. GQR reasons up to 100000 relations in ~ 28 min, Phalanx in double the time of GQR, and, finally, Renz's solver reasons up to 50000 relations in ~ 7 hours. (In practice, Renz's solver was able to fit the 100000 relations of the next step, but judging by its overall performance it would require several days to reason with them.) `Gadm-rdf` is the biggest real dataset to date to have been succesfully employed in an experimental evaluation of qualitative spatial reasoners. Surprisingly, Sarissa runs the `gadm-rdf` experiment faster than the `admingeo` one, but this is due to more relations being inferred in the latter case as a result of dataset particularities that affect the triangulation procedure and the overall reasoning process.

5.1. *A compact* SAT *encoding for* RCC-*8*

In [51] an implementation based on a compact SAT encoding is proposed for RCC-8. The authors introduce the notion of a *dtree* (decomposition tree), which, in fact, can be seen as a chordal graph, since a chordal graph yields a natural decomposition tree of its cliques [30, 52]. Further, the authors use a dedicated graph partitioning software with heuristics to partition the network, viz., hMETIS[m], and do not rely in a linear time triangulation of its underlying graph. The implementation is significantly outrunned by GQR under the older release 1418 by the time networks of 150 nodes are considered (see Figure 13c in [51, chapt. 6.4]). In this paper, we considered hard network instances of 3500 nodes, and were able to perform better than GQR under its newest release, 1500. The compact SAT encoding presented in [51] becomes too large when network sizes increase beyond a few hundred nodes [51, chapt. 6.4].

[m] `http://glaros.dtc.umn.edu/gkhome/metis/hmetis/overview`

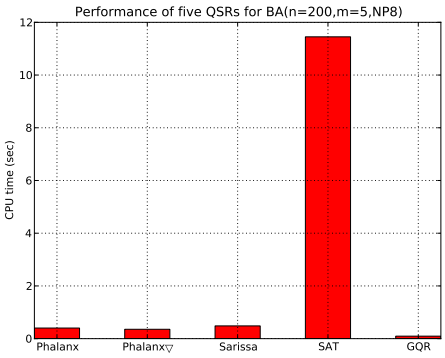


Fig. 14. Experiment with random hard scale-free-like networks and a SAT-based implementation

Nevertheless, we decided to run a small experiment with the SAT-based implementation of [51], the GQR reasoner under its newest release 1500, and our own reasoners that we presented here, viz., Phalanx, Phalanx$_{\triangledown}$, and Sarissa. The benchmark used in [51] consisted of RCC-8 networks generated by the $\mathsf{H}(\mathsf{n},\mathsf{d},\mathsf{l})$ model [5]. Here, we considered RCC-8 networks generated by the $\mathsf{BA}(\mathsf{n},\mathsf{m},\mathcal{NP}_8)$ model.[n] In particular, we created 10 networks with a size of 200 nodes and a preferential attachment value of $m = 5$. The size of 200 nodes might seem too small, but some of the instances consumed over 2 GB when processed by the SAT-based implementation. In any case, the size of 200 nodes is sufficient to draw a conclusion. The results are shown in Figure 14. For a given hard RCC-8 network, Phalanx, Phalanx$_{\triangledown}$, and Sarissa all decide its consistency in approximately 0.4 sec, and GQR in around 0.1 sec. In fact, most of the time consumed by our own reasoners was for initializing data structures and allowing the JIT compiler of PyPy to kick in. On the other hand, the SAT-based implementation needed nearly 12 sec to decide the consistency of a given RCC-8 network, on average. It follows that state-of-the-art SAT encodings are not able to tackle large scale-free-like RCC-8 networks.

At this point we conclude our experimental evaluation. We have presented a complete set of experiments with several network generation models, and several graphs displaying the amount of edges considered in each approach and the effect of the triangulations. Most importantly, we have introduced a new model for generating RCC-8 networks, viz., $\mathsf{BA}(\mathsf{n},\mathsf{m}[,\mathcal{NP}_8])$, and have identified the phase transition region for both normal and hard network instances of that model. We must also state that the techniques implemented in Sarissa can be easily adopted by any rea-

[n]We used this particular model because a SAT-based implementation deals quite well with hard instances when compared with native reasoners [6]. We verified this by conducting some experiments with networks generated by the $\mathsf{BA}(\mathsf{n},\mathsf{m})$ model for which the SAT-based implementation considered here had indeed even poorer performance when compared with the rest of the reasoners.

soner, such as GQR. The outcome of our evaluation should be interpreted more as a comparison between different techniques, and not just as a comparison between differently coded reasoners per se.

6. Related Work

The work presented in this paper is based on, and extends, the work of [53] and the work of [11]. In particular, experimentation in [11] was moderate, in the sense that no hard network instances (viz., instances comprising relations only from the $\mathcal{NP}_8$ set of RCC-8 relations [5]) were used for evaluation, while the comparison between partial path consistency and path consistency was limited to a single random generation model (viz., the $\mathsf{A}(\mathsf{n},\mathsf{d},\mathsf{l})$ model [5]) of networks that scaled up to no more than 100 nodes. In this paper, we explored and took advantage of the stuctural properties of real world networks and proposed an implementation scheme that triangulates their constraint graphs to retain their sparseness and uses a hash table based adjacency list to efficiently represent and reason with them, we investigated and compared the pruning capacity of partial path consistency and path consistency when considering the common edges between a triangulation and the completion respectively of the constraint graph of an input network, we made the case for a new series of random datasets, viz., *large* random scale-free-like RCC-8 networks of model $\mathsf{BA}(\mathsf{n},\mathsf{m}[,\mathcal{NP}_8])$, and argued that they can be of great use and value to the qualitative reasoning community, we experimentally identified the phase transition region for datasets of model $\mathsf{BA}(\mathsf{n},\mathsf{m}[,\mathcal{NP}_8])$, we devised a novel tool, viz., Sarissa, and evaluated it against the state-of-the-art reasoners GQR [7], Renz's solver [5], and two of our own implementations which we presented here, viz., Phalanx and Phalanx$\bigtriangledown$, and showed that it significantly advances the state-of-the-art approach for tackling large real world qualitative spatial networks, and finally we investigated and experimented with a state-of-the-art SAT-based implementation for RCC-8 and showed that we outperform it by a very large scale. All aforementioned contribution points are not considered in [11], while a fair amount of them is also new with respect to [53]. In particular, the presentation of the algorithms, the investigation of the pruning capacity of partial path consistency with respect to path consistency, the experimentation with hard network instances, the utilization of a state-of-the-art SAT-based implementation for RCC-8 in our evaluation, the analysis on the effect of the triangulations, and the experimentation with different triangulation methods, are not present in [53].

In [54] a method was suggested that claims to provide a more general and effective solution than the one presented here. However, the method was shown to be faulty in [52]. In particular, the work of [54] presents an approach that tackles the same type of RCC-8 networks that is of our interest here, while it also claims that it goes beyond chordal graphs by generalizing to a larger family of graphs. Recent literature suggests that the work presented in [54] (i) falls very short with respect to the state-of-the-art for tackling large RCC-8 networks of real world structure [55],

and also (*ii*) lacks soundness [52]. Regarding point (*i*), the reader is kindly referred to the experimental evaluation that took place in [55], where Sarissa was used to tackle real networks scaling up to millions of nodes and was shown to significantly outperform all other reasoners. The results of [55] complement nicely the results of this work and strengthen the practicality of Sarissa. Regarding point (*ii*), a full analysis of the issues that lead to a lack of soundness for the approach presented in [54] takes places in [52]. In [52], it is also explained that had the approach of [54] been sound, it would perform even worse in the evaluation in [55], as it benefits from wrongfully reduced complexity bounds (e.g., some constraints erroneously get pruned off). In [52] we also propose fixes so that the approach of [54] yields soundness. In short, it is necessary that the work of [54] uses chordal graphs to achieve soundness, but not yet sufficient, as the partitioning algorithm employed in [54] is not proper.

7. Conclusion and Future Work

In this paper we have presented an approach that employs chordal graphs and a hash table based adjacency list implementation to tackle large scale-free-like qualitative spatial networks, and goes well beyond the state-of-the-art qualitative spatial reasoners which were found to fall short of the task. It should be noted that we settled with a hash table based adjacency list after trying several approaches with list/dictionary combinations as it gave the best results in our experiments. Identifying the best encoding is not the aim of this work though. Clearly, there is no universally good solution to represent sparse networks.

One could argue that even though being able to tackle a real dataset of nearly a million relations fairly easily, our approach is still far from the billion relations goal set in [17, 18]. However, for the case of tractable RCC-8 networks and due to the notion of *canonical solutions* introduced in [47], we have already presented a vertex incremental variant of the partial path consistency algorithm for building a consistent RCC-8 enriched database incrementally [48]. As future work we plan to also have this functionality for non-tractable RCC-8 networks. This can be achieved by initially reasoning with a small part of the dataset, and then for every new piece of data considering only relevant existing RCC-8 relations (if any), reasoning with the resulted fused piece of information, and continuing the process, while maintaining chordality [56]. Our hash table based adjacency list allows extending a network with new nodes and edges in constant time. In a likewise manner, for quering or updating data one would only have to consider relations relevant to his regular or update query. Other future work consists of further exploring and optimizing on the hierarchical structure that real datasets present, as argued in [18]. In particular, it would be interesting to explore which relations are used more than others in real datasets and whether this could be of some use or not. We would also like to investigate if stuctural or hierarchical properties are observed in real IA networks which we are currently in the process of obtaining from temporally

enriched datasets.

Finally, we feel that it is very important to motivate the qualitative reasoning community to get involved with the structure that real datasets present, and to this direction large random scale-free-like networks can be of great use and value to further improve existing reasoners and present (possibly mixed) solutions that can scale up to billions of relations.

Acknowledgments

This work was funded by a PhD grant from Université d'Artois and region Nord-Pas-de-Calais. We would like to thank Assist. Prof. Konstantinos Blekas for selecting the work of [53] for possible publication in a special issue of the *International Journal of Artificial Intelligence Tools*, but at the time of contacting us regarding this matter we had already submitted our work in the same journal. We would also like to thank the reviewers for their valuable comments and their overall feedback throughout the review process.

References

1. S. Hazarika, *Qualitative Spatio-Temporal Representation and Reasoning: Trends and Future Directions* (Igi Global, 2012).
2. M. Koubarakis and K. Kyzirakos, Modeling and Querying Metadata in the Semantic Sensor Web: The Model stRDF and the Query Language stSPARQL, in *ESWC* (2010).
3. Open Geospatial Consortium, OGC GeoSPARQL - A geographic query language for RDF data, OGC® Implementation Standard (2012).
4. D. A. Randell, Z. Cui and A. Cohn, A Spatial Logic Based on Regions and Connection, in *KR* (1992)
5. J. Renz and B. Nebel, Efficient Methods for Qualitative Spatial Reasoning, *JAIR* **15** (2001) 289–318.
6. M. Westphal and S. Wölfl, Qualitative CSP, Finite CSP, and SAT: Comparing Methods for Qualitative Constraint-based Reasoning, in *IJCAI* (2009).
7. Z. Gantner, M. Westphal and S. Wölfl, GQR-A Fast Reasoner for Binary Qualitative Constraint Calculi, in *AAAI Workshop on Spatial and Temporal Reasoning* (2008).
8. T. Walsh, Search on High Degree Graphs, in *IJCAI* (2001).
9. M. Bodirsky and S. Wölfl, RCC8 is polynomial on networks of bounded treewidth, in *IJCAI* (2011).
10. G. Christodoulou, E. G. M. Petrakis and S. Batsakis, Qualitative Spatial Reasoning Using Topological and Directional Information in OWL, in *ICTAI* (2012).
11. M. Sioutis and M. Koubarakis, Consistency of Chordal RCC-8 Networks, in *ICTAI* (2012).
12. J. Goodwin, C. Dolbear and G. Hart, Geographical Linked Data: The Administrative Geography of Great Britain on the Semantic Web, *TGIS* **12** (2008) 19–30.
13. A.-L. Barabasi and E. Bonabeau, Scale-Free Networks, *Scientific American* (2003) 50–59.
14. O. Hein, M. Schwind and W. Knig, Scale-Free Networks - The Impact of Fat Tailed Degree Distribution on Diffusion and Communication Processes, *Wirtschaftsinformatik* **47** (2006) 21–28.
15. M. Steyvers and J. B. Tenenbaum, The Large-Scale Structure of Semantic Networks:

Statistical Analyses and a Model of Semantic Growth, *Cognitive Science* **29** (2005) 41–78.
16. A. L. Barabasi and R. Albert, Emergence of scaling in random networks, *Science (New York, N.Y.)* **286** (1999) 509–512.
17. M. Koubarakis, K. Kyzirakos, M. Karpathiotakis, C. Nikolaou, M. Sioutis, S. Vassos, D. Michail, T. Herekakis, C. Kontoes and I. Papoutsis, Challenges for Qualitative Spatial Reasoning in Linked Geospatial Data, in *BASR* (2011).
18. C. Nikolaou and M. Koubarakis, Querying Incomplete Geospatial Information in RDF, in *SSTD* (2013).
19. J. Renz and G. Ligozat, Weak Composition for Qualitative Spatial and Temporal Reasoning, in *CP* (2005).
20. J. Munkres, *Topology* (Prentice Hall, Incorporated, 2000).
21. J. Renz and B. Nebel, On the Complexity of Qualitative Spatial Reasoning: A Maximal Tractable Fragment of the Region Connection Calculus, *AIJ* **108** (1999) 69–123.
22. J. Renz and B. Nebel, Spatial Reasoning with Topological Information, in *Spatial Cognition* (1998).
23. J. Renz, A Canonical Model of the Region Connection Calculus, in *KR* (1998).
24. S. Li, W. Liu and S. Wang, Qualitative constraint satisfaction problems: An extended framework with landmarks, *AIJ* **201** (2013) 32–58.
25. J. Renz, Maximal Tractable Fragments of the Region Connection Calculus: A Complete Analysis, in *IJCAI* (1999).
26. C. Bliek and D. Sam-Haroud, Path consistency on triangulated constraint graphs, in *IJCAI* (1999).
27. S. Li and M. Ying, Region connection calculus: Its models and composition table, *AIJ* **145** (2003) 121–146.
28. J. Renz and B. Nebel, Qualitative Spatial Reasoning Using Constraint Calculi, in *Handbook of Spatial Logics* (2007), pp. 161–215.
29. A. Chmeiss and J.-F. Condotta, Consistency of Triangulated Temporal Qualitative Constraint Networks, in *ICTAI* (2011).
30. M. C. Golumbic, *Algorithmic Graph Theory and Perfect Graphs*, 2nd edn. (Elsevier Science, 2004).
31. R. E. Tarjan and M. Yannakakis, Simple Linear-Time Algorithms to Test Chordality of Graphs, Test Acyclicity of Hypergraphs, and Selectively Reduce Acyclic Hypergraphs, *SIAM J. Comput.* **13** (1984) 566–579.
32. A. Berry, J. R. S. Blair and P. Heggernes, Maximum Cardinality Search for Computing Minimal Triangulations, in *WG* (2002).
33. S. Parter, The use of linear graphs in Gauss elimination, *SIAM review* **3** (1961) 119–130.
34. D. R. Fulkerson and O. A. Gross, Incidence matrices and interval graphs., *Pacific J. Math.* **15** (1965) 835–855.
35. M. Yannakakis, Computing the Minimum Fill-In is NP-Complete, *SIAM J. on Algebraic Discrete Methods* **2** (1981) 77–79.
36. N. Amaneddine, J.-F. Condotta and M. Sioutis, Efficient Approach to Solve the Minimal Labeling Problem of Temporal and Spatial Qualitative Constraints, in *IJCAI* (2013).
37. K. Choromański, M. Matuszak and J. Miekisz, Scale-free graph with preferential attachment and evolving internal vertex structure., *J. Stat. Phys.* **151** (2013) 1175–1183.
38. B. Bollobás, Mathematical results on scale-free random graphs, in *Handbook of Graphs and Networks* (Wiley, 2003), pp. 1–37.
39. R. Albert and A.-L. Barabási, Statistical mechanics of complex networks, *Rev. Mod.*

Phys. **74** (2002) 47–97.
40. C. I. Del Genio, T. Gross and K. E. Bassler, All Scale-Free Networks Are Sparse, *Phys. Rev. Lett.* **107** (2011) 178701.
41. S. N. Dorogovtsev, A. V. Goltsev and J. F. F. Mendes, Pseudofractal scale-free web, *Physical Review E* **65** (2002) 066122+.
42. F. Dylla, T. Mossakowski, T. Schneider and D. Wolter, Algebraic Properties of Qualitative Spatio-temporal Calculi, in *COSIT* (2013).
43. J. F. Allen, Maintaining knowledge about temporal intervals, *CACM* **26** (1983) 832–843.
44. P. Heggermes, A. Pothen, G. Kumfert and S. C. Eisenstat, The computational complexity of the minimum degree algorithm, Tech. Rep. NASA/CR-2001-211421-ICASE report 2001-42, NASA (Hampton, VA US) (2001).
45. D. J. Rose, A graph-theoretic study of the numerical solution of sparse positive definite systems of linear equations, in *Graph theory and computing* (Academic Press, New York, 1973), pp. 183–217.
46. P. Jégou and C. Terrioux, Tree-Decompositions with Connected Clusters for Solving Constraint Networks, in *CP* (2014).
47. J. Huang, Compactness and its implications for qualitative spatial and temporal reasoning, in *KR* (2012).
48. M. Sioutis and J. Condotta, Incrementally Building Partially Path Consistent Qualitative Constraint Networks, in *AIMSA* (2014).
49. P. B. Ladkin and A. Reinefeld, Fast Algebraic Methods for Interval Constraint Problems, *AMAI* **19** (1997) 383–411.
50. P. Liakos, K. Papakonstantinopoulou and M. Sioutis, On the Effect of Locality in Compressing Social Networks, in *ECIR* (2014).
51. J. Huang, J. J. Li and J. Renz, Decomposition and tractability in qualitative spatial and temporal reasoning, *AIJ* **195** (2013) 140–164.
52. M. Sioutis, Y. Salhi and J. Condotta, On the Use and Effect of Graph Decomposition in Qualitative Spatial and Temporal Reasoning, in *SAC* (2015).
53. M. Sioutis and J. Condotta, Tackling Large Qualitative Spatial Networks of Scale-Free-Like Structure, in *SETN* (2014).
54. C. Nikolaou and M. Koubarakis, Fast Consistency Checking of Very Large Real-World RCC-8 Constraint Networks Using Graph Partitioning, in *AAAI* (2014).
55. M. Sioutis, Triangulation versus Graph Partitioning for Tackling Large Real World Qualitative Spatial Networks, in *ICTAI* (2014).
56. A. Berry, P. Heggernes and Y. Villanger, A vertex incremental approach for maintaining chordality, *Discrete Mathematics* **306** (2006).