ECAI 2008
M. Ghallab et al. (Eds.)
IOS Press, 2008

doi:10.3233/978-1-58603-891-5-673

Two Stage Knowledge Discovery for Spatio-temporal Radio-emission Data

Matthias Haringer[1], Lothar Hotz[1] and Vera Kamp

Abstract. In this paper, we introduce a method to examine and interpret spatio-temporal radio emission datasets. The goal is to find communication patterns in the data in respect to spatial, temporal, and frequency based attributes. The chosen approach is a combination of two different AI-methods. First a *clustering algorithm* groups spatially close data points to potential emitters. In a second step a *model-based constraint solving technique* is applied to find relationships between the identified emitters. The used models describe rules of the communications that are to be found. This guarantees a flexible search for different kinds of communication.

1 Introduction

The approach to be introduced arose from the need to find communications in radio emission data. The used data is recorded via three or more antennas with distances of several hundred kilometers. The receiver systems match frequency and timing of the received data snippets of all included antennas and attach location information acquired from run time differences of the matching received snippets. Several thousands of matching snippets are collected every second and stored in large databases. This data is the basis of the proposed technique. The interpretation of this data, like finding communication patterns, has mostly been done manually by domain experts with the help of frequency catalogs and database-based approaches. In this paper, we introduce a communication recognition method where less user interaction and expert knowledge is needed.

The radio emission data consists of single data atoms that are called *emissions* in this context. An emission is the smallest unit of received data where the location could be determined. An emission represents a direct or modulated electro-magnetic transmission, which can be a connection to a single partner or a broadcast for multiple receivers. Emissions can be radio station broadcasts (lasting the whole day, always the same frequency), two way transceiver communication, or automated data communication. Each emission has at least following attributes: A start-time, a duration, a frequency range, and a location (latitude and longitude). As only the sender and not the receiver of an emission is directly known, a communication can only be detected when two stations subsequently ”talk” to each other with two or more subsequent emissions (simplex communication), or when they use a ongoing connection on different frequencies (duplex communication). A communication in this context is, therefore, the combination of two or more emissions from two or more different emitters. The simplest communication consists of two emissions. One emission is the initial message and the other is the response.

One way to find out if two emissions are part of a communication is their temporal behavior, as the sequences for natural speech and radio communication protocols follow specific rules.

Another hint of a communication is the location of the emissions. If subsequent emissions come from the same two locations, it can be assumed that two sending units (in the following called emitters) are communicating with each other. An emitter is in this context a stationary unit capable of sending and receiving emissions. A problem of the spatial information (longitude, latitude) is the varying precision (see Section 3).

A third important indicator of some types of communication is frequency equality. Most emitters use the same frequencies for a communication. However, it would be too restrictive to limit the system to that, because some emitters change their frequency with each emission of a communication or after defined time periods.

A large challenge in the given application is that there are many forms of radio communication like simplex, duplex, shallow and deep emitter hierarchies with specific timings, different modulation types, and different data transmission types. Examples of communications are mobile telephone transmissions, radio stations, airplane communication and all kinds of wireless communication. The search for patterns is, therefore, very difficult and a general search for all communications with one set of criteria is not possible. This led to the idea of a model-based approach, where specific communication types can be formulated: Sometimes almost nothing is known about the communications to be found. This would require a general model. However, in most cases some more restricting parameters like the approximate duration, the gaps between emissions of a communication, one or more frequencies, the number of involved emitters, or the location of some emitters are known. These tasks are different use cases which have to be addressed with our prototype.

A problem are arbitrarily fitting emissions which result from the large amount of simultaneous emissions. The system finds only possible communications, which have to be reviewed by experts. But the expert time compared to analysis with raw data can be reduced significantly with this method.

2 General Approach

We identified two subproblems for finding communications with time and location based emission data: *Emitter identification* which allows to determine an emitter for each emission and *communication identification* between emissions of multiple emitters. In a first step we investigated in different data-mining approaches and examined their suitability to our problem. Clustering was found suitable for emitter identification and sequence analysis for combining emissions to communications. Clustering has the capability to find similar

[1] HITeC c/o Department Informatik, University of Hamburg, Germany, email: {haringer,hotz}@informatik.uni-hamburg.de

[2] Plath GmbH, Hamburg, Germany, email: kamp@plath.de

[3] The whole work was performed in cooperation with Plath GmbH.

data objects with the help of a distance measure. Clustering is applicable, if spatial, temporal or other 1 to n dimensional data groups have to be identified. Taking into account the spatial relationship of the emission data, clustering is helpful to find emitters, frequency groups, and temporal accordance. Sequence analysis allows to find temporally connected relationships and with that the recognition of compound events. Sequence analysis can be used in the given scenario for finding communications and communication sequences. As there are several different types of communication and the parameters should be easily changeable a model-based approach has been chosen. Therefore, following general process of analysis and interpretation is proposed:

1. Data preparation: exclusion of non relevant attributes, plausibility checks, data reduction, dimension reduction, and data partitioning.
2. Emitter identification: Group emissions into potential emitters using clustering techniques.
3. Application of the main interpretation method: model-based recognition of temporally repeating structures between emitters.

Data reduction is very important with large data sets to reduce the computation time of the whole system. In the following emitter identification (Section 3), and finding communication patterns between emitters (Section 4) are explained in more detail. The complete approach has been implemented in a prototypical system (Section 5) and has been validated with several experiments (Section 6).

3 Emitter identification

In this phase emissions which originate from the same emitter are to be grouped together. One advantage of this approach is to reduce the complexity for the later model-based steps. For finding a *start* of a communication with one emission of an emitter e, for instance, all other emissions except in e are candidates for a communication. To detect *ongoing* communications only the participating emitters have to be taken into account. Clustering algorithms have been identified as most important method to emission emitter estimation. Each emission has longitude and latitude as spatial information. Longitude and latitude have different uncertainties depending on the used receivers and the distance from the emitter. If for example 3 receivers in a 1000km equilateral triangle are used, the best emission location results are in and near this triangle As the distance increases, different ellipses with increasing areas have to be considered as locations instead of points. As most of the clustering algorithms work only for point data, the geometrical centers of the uncertainty areas are used for clustering. Latitude and longitude are transformed to two dimensional points on the earth surface. As a distance measure for clustering, euclidian distance is sufficient, as only spatially very close emissions are grouped together. A better solution would be to use the arc distance on the sphere surface, but several clustering methods do not easily support custom distance functions and we wanted to compare different algorithms first.

Most clustering methods are directly applicable to the large dataset. The time complexity for relocation and hierarchical algorithms can be controlled by appropriate termination criteria. Most clustering algorithms are apart from the initialization and the definition of the distance measure free from interaction.

In the following different clustering methods are compared for the suitability for the targeted application:

- Exclusive and overlapping relocational clustering algorithms can use random generated and evenly distributed locations as start centroids. The number of centroids and clusters has to be known beforehand. The ordinality of emitters can be roughly approximated from the number of emissions.
- Density-based methods are due to the circular or elliptic forms of emitter clusters not relevant. Density-based clustering is especially interesting for concave cluster forms.
- Probabilistic methods assume a few defined probability distributions in the dataset. Whether this applies to the emission data, would have to be examined first.
- For hierarchical clustering only the link method and the terminate criterion have to be predefined. Complete hierarchical clustering has a complexity of $O(n^2)$ which is too time consuming for large datasets. In our application the number of expected data objects per cluster is small compared to the total number of objects. Because of that the algorithms can be terminated after several iterations, as soon as a maximal distance criterion is met.

Modern heterogeneous clustering algorithms unite the advantages of several cluster methods. Because of the large datasets it was important that the algorithm was not only main memory-based. The targeted complexity should be O(n log(n)) or better O(n) to be applicable.

Several clustering methods have been tested: Different algorithms from the CLUTOclustering toolkit and the CURE clustering algorithm. The CLUTO clustering toolkit is a freely accessible easy to use library for several clustering methods described in [4]. CLUTO allows to parameterize and adapt partitioning and hierarchical clustering algorithms. The algorithms in CLUTO are optimized for large datasets and a high dimensionality. This is especially true for the algorithms based on partitioning. Drawbacks of CLUTO are that all algorithms in CLUTO need a previously defined number of clusters. Another problem is that no user defined distance functions can be introduced for more complex clustering tasks. The solution for avoiding the predefined cluster number problem, is to examine the density and distribution of the data to estimate the cluster number. It has been found that 1 to 10 clusters per 1000 emissions give the best results.

The CURE (Clustering Using Representatives) algorithm [2] is an agglomerative, hierarchical cluster algorithm. The algorithm usually stops at a predefined number of clusters c. It was changed to terminate when a maximal distance between two clusters has been reached.

The algorithms have been applied to different emission datasets and examined for performance, memory usage, and quality of the results (see Section 5).

4 Model-based communication recognition

The clustered emissions from the emitter identification build the input for model-based recognition of communication structures. Manually edited communication models are a further input. Figure 1 illustrates such a model of a communication structure (here a simplex communication between two partners). The communication model consists of several *emission models*, which have a start-time and end-time and which have to fulfill certain conditions. Each emission model represents a generic description of real emissions. Instead of absolute time relative relations between time points are specified. Communication models may also be composed. The communication model in Figure 1 consists for example of two sub-models: One that describes a start of a communication (*start-connection*) and one that describes the follow-up communications (*alternating-communications*).

For automatic computation of models a textual, LISP-based model definition language has been defined. This language and its use is discussed in Section 4.1. In the further steps a communication model is seen as a specification of a constraint problem. Thus, such models are an abstraction of typically hard to formulate constraint problems. For processing, the models are automatically mapped to a constraint problem (see Section 4.2).

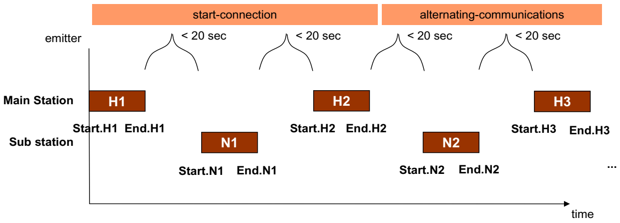

Figure 1. Graphical notation of a simplex communication model.

4.1 Representing Communication Models

For representing communication models we developed the new declarative language `ModoCom` (*Modeling of Communications*). This language provides following modeling facilities:

Primitive communication models specify a combination of emissions. Each emission belongs to a communication partner (i.e. an emitter). Combinations of emissions can be restricted by conditions.

Each emission of a partner can be described by parameters, like start, end, delay, longitude, latitude etc.

Restrictions specify conditions on the emissions of the communication partners. Conditions can be time related (e.g. partner `a` before partner `b` with delay `d`), related to the frequency of the emission (e.g. equal frequency for simplex communications) or spatially related (e.g. different positions in space).

Compositional communication models combine primitive communication models or other compositional communication models. Thus, a hierarchical structure of models is established.

Primitive communication models can be seen as templates that are used to identify communication structures in the emitter clusters. The result of evaluating primitive communication models are combinations of emissions that fulfill the restrictions of the primitive communication model. In Figure 2 an example for a primitive communication model is given. Two partners (emitters) are being specified - a main station (`?m`) and a sub station (`?s`). The emissions of these emitters are referenced by variables (`?m1, ?m2, ?s1`). Those are restricted by the specified constraint relations to have maximally 20 seconds *delay*. These restrictions relate parameters like `emission-endtime` or `emission-cluster` by the shown constraint relations (`less`, `adder`).

Primitive communication models can be combined to *compositional communication models*. The computation of such models consists of two steps: First communications that fulfill the sub-models are generated (`:generate`). This step leads to combinations of emissions. In the second step these combinations are again composed according to further restrictions. In Figure 3 a communication model is composed from two sub-models `simplex-2-partners-start-connection` and `simplex-2-partners-alternating`. Their combination results are again referenced by variables (`?mir`, `?mar`). The domains of these variables are combinations of emissions (`a-cmb`). These are generated and further restricted by combining them with the specific constraint relation `combine-models`.

```
(define-communication-model
  :name simplex-2-partners-start-connection
  :parameters ((delay 20.0))
  :partners
   (;; main station
    (?m :type cluster
        :events ((?m1 :type emission)
                 (?m2 :type emission)))
    ;; sub station
    (?s :type cluster
        :events ((?s1 :type emission))))
  :restrictions
   ((related-to (?m1 ?m2 ?s1)
      (;; similar frequency different clusters
       (frequency-overlap-for-three ?m1 ?m2 ?s1)
       (unequal (emission-cluster ?m2)
                (emission-cluster ?s1))
       ;; different emissions
       (unequal ?m2 ?s1) (unequal ?m1 ?s1)
       (unequal ?m1 ?m2)))
    (related-to (?m2 ?s1 delay)
      (;; m1.end + 20 < s1.start
       (less (emission-endtime ?m1)
             (emission-starttime ?s1))
       (adder (emission-endtime ?m1) tmp
              (emission-starttime ?s1))
       (less tmp delay)))
    (related-to (?m2 ?s1 delay)
      (;; s1.end + 20 < m2.start
       (less (emission-endtime ?s1)
             (emission-starttime ?m2))
       (adder (emission-endtime ?s2) tmp
              (emission-starttime ?m2))
       (less tmp delay)))))
```

Figure 2. Declarative specification of a primitive communication model. The delay between `?m1` and `?s1` as well as between `?s1` and `?m2` has to be less than the delay. `adder` is a constraint relation that ensures: $a + b = c$. Thus, the second argument (here `?tmp`) is the distance between the first and the third argument.

4.2 Using constraints for recognizing communications

Each communication model is mapped to a constraint problem. A constraint problem is specified by constraint variables having a domain and n-ary constraints. A constraint restricts the domains of the variables (also called *pins*) that are connected to it. Because one variable may be connected to several constraints, a *constraint network* is formed. A *solution of a constraint network* given by constraints and variables is a tuple of values for each variable that satisfies the constraints. Given a constraint network the *constraint satisfaction problem* is the problem of computing one, several or all solutions of the network (see e.g. [5]). Several constraint systems exist which implement this kind of computation.

A solution of a constraint network is given by restricting the given variable domains. In principle, every combination of values for the variables has to be checked, whether it is consistent with the constraints. Therefore, a constraint system solves the combinatorical challenge of identifying combinations of values that are consistent with the constraints.

This facility is used for computing communications consisting of emissions (e.g. modeled by variables). Restrictions given by communications, like duration, structure, and spatial location, are mapped to constraints. In principle, every emission can be combined to form a communication, however, only those combinations that are consistent with the restrictions of the communication are of interest. Because of that, the challenge of communication identification can be mapped to a constraint problem.

The mapping of non-aggregated, primitive models like the simplex model is as follows: each partner (or emitter cluster) of the communication model are modeled as a constraint variable. The set of emissions that are gathered in one cluster is the domain of a constraint variable. The restrictions on the emissions that are given in the model are mapped to constraints. Each expression is one constraint, the variables of the expression are the pins of the constraint.

By solving the constraint problem for a primitive communication model a combination of emissions is computed that fulfills the re-

```
(define-communication-model
  :name simplex-2-partners-aggregate
  :sub-models
   ((?ma :type simplex-2-partners-start-connection
         :solutions (?mar :type a-cmb))
    (?mi :type simplex-2-partners-alternating
         :solutions (?mir :type a-cmb)))
  :clusters ((?a :type spatialcluster)
              (?b :type spatialcluster))
  :generate
    ((related-to (?ma ?a ?b)
       ((unequal ?a ?b)
        (compute-submodel ?ma ?a ?b)))
     (related-to (?mi ?a ?b)
       ((unequal ?a ?b)
        (compute-submodel ?mi ?b ?a))))
  :restrictions
    ((related-to (?mir ?mar ?a ?b)
       ((unequal ?a ?b)
        (combine-models ?mir ?mar)))))
```

Figure 3. Declarative specification of a compositional communication model. Two sub-models are referenced and emissions that fit to this sub-models are generated. The result of this generation are combinations of emissions. Those are combined with the constraint relation `combine-models`.

strictions specified in the communication model. This is achieved by computing all solutions of the constraint network by global propagation.

Compositional communication models are mapped as follows: Results of sub-models are combinations of emissions. These combinations are taken as domains for the variables of the compositional communication model (e.g. of `?mir`). Thus, while constraints of primitive communication models handle primitive parameter values like time points or spatial values, constraints of compositional communication models handle emissions belonging to combinations of emissions. Such constraints are newly defined as functions (here also called *constraint relations*) for the constraint system (e.g. `compute-submodel`). These functions implement specific algorithms which take the time-line of emissions or equivalence classes built by frequencies into account. This approach of using results of a constraint problem as variable domains of a further constraint problem is new (at least to our knowledge) and here called *cascading constraints*. Through the compositional structure of the communication models cascading constraints are implicitly modeled, i.e. the results of one communication model (yielded by solving a constraint problem) are automatically transferred to the next higher aggregation level and there used as input for the next constraint problem. In typical applications of constraints only one level of variables are used.

4.3 Discussion of the Model-Based Approach

Through the clustering of emissions the search for communications is facilitated. For searching a communication between two partners, for example, only emissions of the two participating emitters have to be considered. By using such clusters it is not necessary to compare emissions of one cluster with each other, but only emissions of different clusters. Furthermore, if a start of a communication is identified between a set of clusters, only emissions of those clusters have to be considered for continuing the communication.

Due to the prescribed modeling language for representing communications, generic descriptions of typical communication structures can be specified which represent a set of communications. The communication models enable the modeling and identification of commonly known and frequently used communication structures (like *simplex communication of two partners*). Models can be composed to more complex ones. Such compositional models are solved by cascading constraints.

With communication models an enumeration of some finite communications is avoided. This leads, for example, to better maintenance properties of the resulting system. Furthermore, the models can be easily communicated to domain experts (here communication experts), because they concentrate on domain aspects (like delay between emissions).

By using a declarative language, a strict separation of the models (i.e. the knowledge about the domain) from the algorithms (i.e. constraints solving algorithms) is achieved. Thus, the constraint algorithms can be improved without changing the models as long as the language stays the same (e.g. improving variable ordering). The separation of models and algorithms also enables the concentration on the formulation of the model instead of algorithm development. Furthermore, an evolutionary modeling approach is supported by frequently testing specified models with the constraint solver.

By using a constraint solver the algorithms are clearly defined and have known properties (e.g. termination as long as variable domains are reduced). A specific algorithm developed for recognizing communications would be a black box for others than the developers, e.g. properties of such an algorithm would have to be newly identified. However, parts of specific algorithms can be incorporated by implementing new constraint relations.

5 Prototype implementation

For evaluating the approach described in this paper, we realized a prototype with a distributed architecture. The emissions and the results are stored in an Oracle database. The previously described clustering methods are implemented or integrated in a extendable C++ clustering module. The data is read from the database, passed to the clustering algorithm, and the resulting clusters are stored in the database.

The developed model-based constraint system (communication identification) is based on SCREAMER [3] and Common Lisp.This constraint system provides finite domains of numbers, symbols and objects as well as intervals of numbers (i.e. it is a heterogenous constraint system). Additionally, n-ary constraints and the definition of domain-specific constraint relations (like `compute-submodel`) are supported (see also *constraint operators* defined in [1]). Such constraint relations enable the implementation of cascading constraints which use solutions of a constraint problem as input for a further constraint problem (i.e. as domains of structured variables). Furthermore, we enhance SCREAMER by introducing *series as variable domains*. Because constraint variables have typically *sets* as domains, no support for series of values are given in constraint systems. We enhanced SCREAMER for using ordered sets of variable values, which reduces the computation time in our experiments from several hours to several minutes, because of pruning values within a variable domain.

The communication identification module reads the clusters and emissions from the database, performs the search with the selected communication model, and stores found communications in the database. For user-interaction a Java based Graphic User Interface (GUI) has been implemented. The GUI can select data stores, map regions etc. Each module may run on a separate computers and provides services (like `compute-clusters`, `compute-interpretations`, `call-visualization`), which are implemented with the remote-procedure protocol XML-RPC [2]. The framework has well defined interfaces and allows the integration of other modules and algorithms.

[2] www.xmlrpc.com

6 Experiment Results

We have performed several experiments with the prototype. For the first experiment we reduced the emission database to those emissions, containing spatial information and which belong to a six hours time period. This reduction results in a set of about 50000 emissions (dataset $DS1$). Three of the examined use cases have been finding radio stations, simplex-, and duplex communications in this data.

Algorithms	No.Clust.	No.Emiss.	Qual.	Time
CLUTO Graph	1000	50000	ok	273s
CLUTO Graph	50	500	ok	0.6s
CLUTO Agglo	50	500	ok	0.9s

Table 1. Comparing clustering algorithms.

The used clustering algorithms for the emitter identification are: CLUTO normal partitioning (*CLUTO RB*), CLUTO direct partitioning (*CLUTO Direct*), CLUTO graph-based partitioning (*CLUTO Graph*), CLUTO agglomerative hierarchical (*CLUTO Agglo*), and the *CURE* algorithm.

Model	Algorithm	No.	Com.	Time
Radio emitter	CLUTO Direct	1000	91	direct
Simplex model	CLUTO Direct	1000	62	16 h
Simplex model	CLUTO Graph	1000	304	3:45 h

Table 2. Results of recognizing communications within $DS1$ with 50000 emissions. Runtime without cluster computation.

$CLUTO\ RB$ and *CLUTO Direct* did produce unacceptable shapes of clusters. This is due to the inefficiency of these algorithms to low dimensional clustering. The *CURE* and the *CLUTO Agglo* algorithm lead to memory overflow (swapping), because they are purely main memory based. In Table 1 the result of the most promising algorithms is summarized. The agglomerative approach produced the best cluster shapes, but was not scalable above 1000 emissions due to memory issues. The graph algorithm performed reasonably well for cluster shapes and run time.

The used models for the communication identification specify start connection communications similar to those presented in Section 4.1 for simplex and duplex communications as well as models for identifying radio emitters (long emissions, one emitter). In Table 2 some results for recognizing communications in $DS1$ are shown. The performance and the result of the model processing depends heavily on the quality of the clusters. This examination supports our approach with two phases. The identified communications are seen as hints for real communications and thus, have to be further examined, e.g. by visualization techniques. In Table 3 one example of a communication start is shown which fulfills the restrictions of the simplex model.

Domain experts have evaluated the found communications and have identified them as well-known and as newly accounted frequencies, emitters, and communications. For this evaluation task several use cases have been developed like *change of frequencies*, *increasing appearances of communications*, *command structures*, *communications with multiple partners and multiple frequencies*. In Figure 4 a screenshot of the developed prototype is shown, where several identified communications are visualized.

Emis.	Start	End	Cl.	Freq.	Lon E	Lat N
19020	06:00:27	06:00:34	0	8766k	28 21	42 24
19482	06:00:37	06:00:44	299	8766k	26 38	43 14
19938	06:00:46	06:00:54	0	8766k	28 21	42 19
20880	06:01:06	06:01:13	0	8766k	28 21	42 19
21375	06:01:15	06:01:23	299	8766k	26 38	43 14
21846	06:01:17	06:01:30	299	8766k	26 38	43 14

Table 3. Several subsequent emissions of two clusters with the same frequency. The emissions are given by their id ($Emission$), starting and ending time ($Start$, End), their clusters, the emission's frequency, and the longitude and latitude coordinates (E = Eastern, N = Northern).

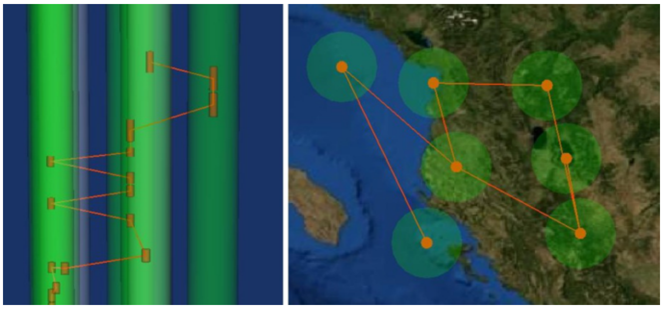

Figure 4. Visualization of several identified communications. Communications are visualized by connected points. Clusters are visualized as transparent shapes. Left: Side-view - vertically the time axis. Right: Top-view with geographical map.

7 Conclusion

We introduced a two step approach for finding communications in radio emission data with spatial information. The first step uses spatial clustering to identify emitters. Several data preparation and clustering algorithms were applied and graph-based partitioning was the most promising choice for our application. The second step uses a model-based interpretation approach which is based on constraint solving with newly introduced cascading constraints. Thus, the approach demonstrates the combination of two AI-methods, i.e. clustering algorithms and model-based interpretation based on constraints. With defining a relatively simple model the user is able to find specific or general communications. We presented results with a prototype implementation which proved the basic concept of our approach.

As a next step the single methods can be refined and optimized. A customized cluster algorithm has to be implemented which uses an optimized distance measure, integrates the location inaccuracies, and uses caching strategies. Emitter identification for moving emitters (tracking) could be introduced.

Another improvement would be a graphical model editor which allows to define the models graphically and generates models using the introduced modeling language as an output. Further models and use cases for different communication types have to be examined.

Other application areas could be the evaluation of mobile communication networks, bio-physiological analysis, or other large scale spatio-temporal datasets.

REFERENCES

[1] Frédéric Benhamou, 'Heterogeneous constraint solving', in *ALP '96: Proceedings of the 5th International Conference on Algebraic and Logic Programming*, pp. 62–76, London, UK, (1996). Springer-Verlag.

[2] Sudipto Guha, Rajeev Rastogi, and Kyuseok Shim, 'CURE: an efficient clustering algorithm for large databases', in *Proceedings of ACM SIGMOD International Conference on Management of Data*, pp. 73–84, (1998).

[3] Jeffrey Mark Siskind and David Allen McAllester, 'Screamer: A portable efficient implementation of nondeterministic common lisp', Technical Report IRCS-93-03, Institute for Research in Cognitive Science, Philadelphia, PA, (1993).

[4] M. Steinbach, G. Karypis, and V. Kumar, 'A comparison of document clustering techniques', in *KDD Workshop on Text Mining*, (2000).

[5] E. P. K. Tsang, *Foundations of Constraint Satisfaction.*, Academic Press, London, San Diego, New York, 1993.