

SAT representation of randomly deployed Wireless Sensor Networks*

**Csaba Biró, Gábor Kusper, Tibor Radványi,
Sándor Király, Péter Szigetváry, Péter Takács**

Eszterhazy Karoly University of Applied Sciences
{birocs, gkusper, dream, ksanyi, szigipet, takip}@aries.ektf.hu

Abstract

Wireless sensor networks consist of numerous - mainly low-cost - sensors. These systems main purpose is to collect environmental data and transmit those measurements. Each sensor can communicate through broadcasting to other sensors in their broadcasting range. There are several requirements that a wireless sensor network should satisfy. One of the most important question is whether communication among all the sensor nodes is possible. In this paper we introduce two logical models that are suitable to represent wireless sensor networks. We generate SAT problems from the network communication model and the restrictions defining expected behavior, which are then analyzed with SAT solvers. The SAT solvers are used for model inspection. Whereby if a requirement is fulfilled, the SAT solver cannot find a substitution value for the function, otherwise the substitution value produced by the SAT solver appoints a counterexample. Current modern SAT solvers are capable to examine the satisfiability of SAT problems having thousands of variables, thus being able to decide even in huge sensor networks that the model fulfills the defined requirements (the degree of fault tolerance or power consumption, etc.)

Keywords: WSN, RFID, Logic model, SAT representation

MSC: 68R05, 68R10, 68W01

1. Introduction

Due to the rapid development of low-cost, small-sized, low-powered micro-electronic and electro-mechanic devices (Micro-Electric-Mechanical-Systems – MEMS) nowadays we can build cost-efficiently and easily wireless sensor networks consist of

*Supported by: FutureRFID Development possibilities in the RFID / NFC technology TÁMOP-4.2.2.C-11/1/KONV-2012-0014.

thousands of sensors even on hard to reach terrains. The uses of such networks can even include among others the detection of seismic activity, or monitoring audible or radar sound waves. The sensors - that can be considered the nodes of network - are capable of processing a limited amount of information and furthermore wireless communication. Nowadays sensor networks are an integral part of any C4ISR system (Command, Control, Communication, Computing, Intelligence, Surveillance, Reconnaissance, Targeting)[1, 2].

The placement of the sensors can be either deterministic or random, but the final location depends largely on the attributes of the area or building, or simply on accessibility. While deterministic deployment takes into consideration the environmental parameters and the field of application to place the sensors strategically, the random method of deployment (e.g. spreading from an airplane) makes the sensors final position unpredictable. Another important trait of such networks is about the type of nodes it contains. We can differentiate homogenous and heterogeneous nodes that made up a network [3].

The most critical aspect of a wireless sensor network is its fault tolerance. Many challenges are to be faced to ensure the appropriate level of fault tolerance, like problems with the power supply, hardware failures, communication errors/disruptions, malicious attacks. Nowadays the most intensively researched field deals with the optimization of the sensor deployment locations to ensure more effective power-consumption and communication, and an acceptable level of fault tolerance.

2. Preliminaries

Given a Boolean variable x , there exist two *literals*, the positive literal x and the negative literal \bar{x} . A *clause* is a disjunction of literals, a *cube* is a conjunction of literals. Either a clause or a cube can be considered as a finite set of literals.

A *truth assignment* for a (finite) clause set or cube set F is a function ϕ that maps literals in F to $\{0, 1\}$, such that if $\phi(x) = v$, then $\phi(\bar{x}) = 1 - v$. A clause resp. cube C is satisfied by ϕ if $\phi(l) = 1$ for some resp. every $l \in C$. A clause set resp. cube set F is satisfied by ϕ if ϕ satisfies C for every resp. some $C \in F$.

For representing the input clause set for a SAT solver the DIMACS CNF¹ format is commonly used, which references a Boolean variable by its (1-based) index. A negative literal is referenced by the negated reference to its variable. A clause is represented by a sequence of the references to its literals, terminated by a "0". The iCNF format extends the CNF format with a cube set.² A cube, called an assumption, is represented by a leading character "a" followed by the references to its literals and a terminating "0".

¹www.satlib.org/Benchmarks/SAT/satformat.ps

²<http://users.ics.tkk.fi/swiering/icnf/>

3. SAT problem

Propositional satisfiability is the problem of determining, for a formula of the propositional logic, if there is an assignment of truth values to its variables for which that formula evaluates to true. SAT is considered the problem of propositional satisfiability for formulas in conjunctive normal form (CNF). SAT is one of the most-researched NP-complete problems [6] in several fields of computer science, including theoretical computer science, artificial intelligence, hardware design, and formal verification [12]. Modern sequential SAT solvers are based on the Davis-Putnam-Logemann-Loveland (DPLL) [7] algorithm. This algorithm performs Boolean constraint propagation (BCP) and backtrack search, i.e., at each node of the search tree it selects a decision variable, assigns a truth value to it, and steps back when conflict occurs.

Conflict-driven clause learning (CDCL) [12] is based on the idea that conflicts can be exploited to reduce the search space. If the method finds a conflict, then it analyzes this situation, determines a sufficient condition for this conflict to occur, in form of a learned clause, which is then added to the formula, and thus avoids that the same conflict occurs again. This form of clause learning was first introduced in the SAT solver **GRASP** [11] in 1996. Besides clause learning, lazy data structures are one of the key techniques for the success of CDCL SAT solvers, such as “watched literals” as pioneered in 2001, by the CDCL solver **Chaff** [9, 10]. Another important technique is the use of the VSIDS heuristics and the first-UIP backtracking scheme. In the state-of-the-art CDCL solvers, like **PrecoSAT** and **Lingeling** [4, 5], several other improvements are applied.

Besides enhanced preprocessing techniques like e.g. failed literal detection, variable elimination, and blocked clause elimination, clause deletion strategies and restart policies have a great impact on the performance of the CDCL solver.

4. Definitions

Let α_0 be a language of propositional logic and Φ be a *WFF* of propositional logic. Denote $\mathcal{I} : \alpha_0 \rightarrow \{\mathcal{T}, \mathcal{F}\}$ a boolean interpretation of Φ .

Let $\mathcal{S} = \{x : \mathcal{I}(x)\}$ be a set, where $x_1, x_2, \dots, x_n (n \in \mathbb{N})$ variables the representations of the sensors.

By communication graph we mean is a directed graph $\mathcal{D}_{\mathcal{S}} = (\mathcal{S}; \mathcal{E}_{\mathcal{C}})$, where $\mathcal{E}_{\mathcal{C}} \subseteq \mathcal{S} \times \mathcal{S}$. An edge $(x_i, x_j) \in \mathcal{E}_{\mathcal{C}}$ represents the possibility of messaging from node x_i to x_j in $\mathcal{D}_{\mathcal{S}}$. The x_i can send message to x_j iff x_j in x_i transmission range.

In the case of homogeneous sensors the $\mathcal{D}_{\mathcal{S}}$ graph is symmetric accordingly $\mathcal{D}_{\mathcal{S}} = (\mathcal{S}; \mathcal{E}_{\mathcal{C}})$ is equivalent to a simple undirected graph $\mathcal{G}_{\mathcal{S}} = (\mathcal{S}; \mathcal{E}_{\mathcal{C}})$.

A **clique** in our communication graph is an undirected graph $\mathcal{G}_{\mathcal{S}} = (\mathcal{S}; \mathcal{E}_{\mathcal{C}})$ is a subset of the vertex set $\mathcal{Cl} \subseteq \mathcal{S}$, such that for every two vertices in \mathcal{Cl} , there exists an edge connecting the two.

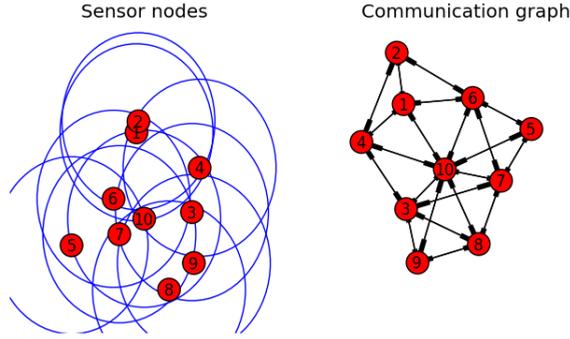


Figure 1: Homogeneous sensor nodes with their communication graph

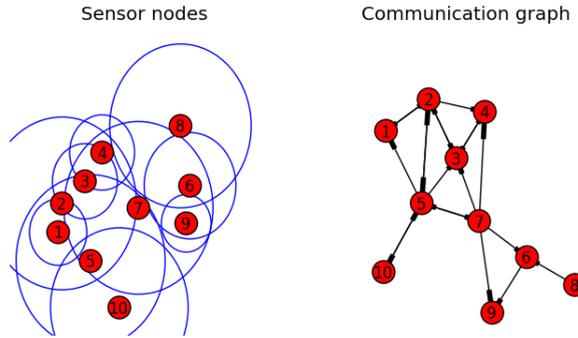


Figure 2: Heterogeneous sensor nodes with their communication graph

A **maximal clique** Cl_{max} is a clique that cannot be extended by including one more adjacent vertex, that is, a clique which does not exist exclusively within the vertex set of a larger clique.

A **n -clique** is a clique of order n .

5. Results

In our case the sensors are deployed randomly and homogeneous typed, i.e., some of them might be very near to each other, others might be far from the rest of the sensors.

5.1. 1st examination

Model

In this model we define the communication by logical implication. If sensor x_1 can send message to both sensor x_2 and x_3 , then the model is:

$$(x_1 \supset x_2) \wedge (x_1 \supset x_3). \tag{5.1}$$

We examine which sensors can communicate with which sensors, thus producing a logical model and graph of the communication, and thereafter we define the requirements for the communication.

In this approach it can be checked in a given state of a randomly distributed sensor network (assuming all sensors are awoken) whether it can be ensured that each sensor can communicate with every other one. The graph theory that interprets the communication graph of the problem answers whether it is a connected graph or not. The first examination is appropriate to examine both global or local properties.

Example 5.1. For example let's examine the steps to build a model of a randomly distributed, homogeneous sensor network having 10 nodes and the answers it gives to the communication restrictions.

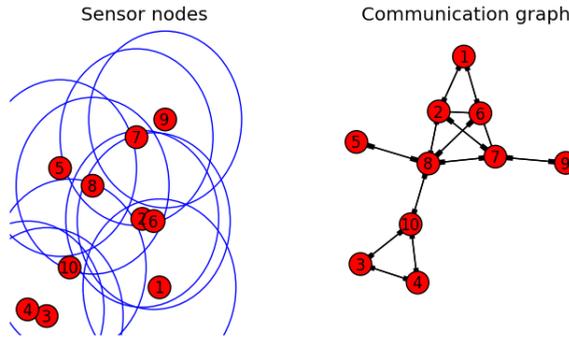


Figure 3: Homogeneous sensor nodes with their communication graph

The description of the communication of x_1 sensor:

$$(x_1 \supset x_2) \wedge (x_1 \supset x_6) \wedge (x_2 \supset x_1) \wedge (x_6 \supset x_1) \tag{5.2}$$

$$(\neg x_1 \vee x_2) \wedge (\neg x_1 \vee x_6) \wedge (\neg x_2 \vee x_1) \wedge (\neg x_6 \vee x_1) \tag{5.3}$$

DIMACS CNF format is the following:

p cnf 10

```
-1 7 0
-1 8 0
-7 1 0
-8 1 0
```

We attain the full model when every sensor's communication is added as a clause.

Defining requirements

The restrictions that allow us to check whether communication is available between any two sensors can be defined as follows:

```
1 2 3 4 5 6 7 8 9 10 0
-1 -2 -3 -4 -5 -6 -7 -8 -9 -10 0
```

Results

The produced DIMACS CNF formatted file is examined with `MiniSat 2.2.0` – a complete SAT solver, which returns unsatisfiable (UNSAT), thus the model fulfills the requirements, namely communication is ensured between any two nodes.

If the starting problem is UNSAT, thus communication is ensured between any two sensors, we can individually investigate the role of each sensor taking part in the communication.

We take the sensors out of the model one by one, and with the remaining we test again the communication problem between any two nodes. If the solver returns satisfiable (SAT) for the reduced model, then the currently removed node is extremely important to the sensor network, as its removal breaks the requirement of unhindered communication between any two sensor. The significance of this in graph theory is that the removal of this node makes the graph not connected anymore.

In our example the removal of nodes 7, 8, and 10 would return with SAT, thus the malfunction of those sensors means the communication in that sensor network is inadequate.

5.2. 2nd examination

Model

The second model concentrates mainly on the inner structure of the communication graph. This model represents whether a sensor is in a state of either asleep or awakened, rather than describing their communication.

After constructing the communication graph, we search for all the biggest sized cliques, namely every subgraph that is a complete graph. According to this, in

such cliques, each sensor can communicate with every other ones.

The clique consisting of the nodes 1, 2, 3, 4 is represented with the string: (1,2,3,4).

Representing restrictions inside a clique

Let a clique consisting of n sensors be given (n -clique), the communication restrictions of an n -clique on the k degree means the combination without repetition of the elements of the n -clique on the k degree.

The strongest restriction that can be formulated for a clique is when k equals 1, meaning every sensor in that clique is awake.

In the following we will examine the problem when k equals 2.

Let's look at a clique consisting of 4 sensors (1,2,3,4) having a communication restriction on the 2nd degree:

- 1 2 0
- 1 3 0
- 1 4 0
- 2 3 0
- 2 4 0
- 3 4 0

Example 5.2. For example let's examine the steps of building a model of a randomly distributed, homogeneous sensor network having 10 nodes and the answers it gives to the communication constrains.

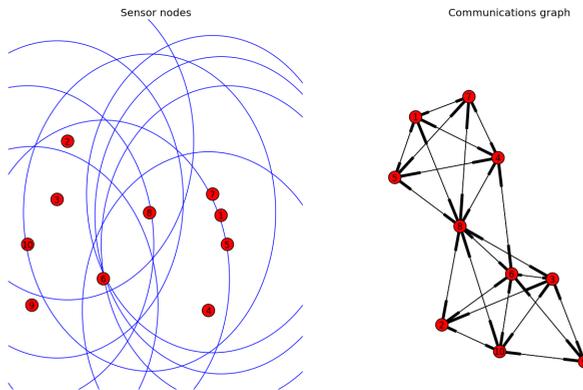


Figure 4: Homogeneous sensor nodes with their communication graph

The cliques that can be generated from the sensor network on the figure 4: (8, 2, 10, 3, 6)

(8, 4, 1, 5, 7)

(8, 4, 6)

(9, 10, 3, 6)

The first clique (8, 2, 10, 3, 6) having a communication restriction on the 2nd degree.

8 2 0

8 10 0

8 3 0

8 6 0

2 10 0

2 3 0

2 6 0

10 3 0

10 6 0

3 6 0

When examining a certain sensor we extend the model with a singleton clause containing a negative literal (e.g. -1 0) representing that particular sensor.

Defining requirements

The requirements for this model are defined as assumptions. The set of assumptions contains (as introduced in the previous model how to define) all the clauses representing all the possible communications.

Example:

a -1 2 0

a -1 3 0

a -1 4 0

...

The number of assumptions

$$\binom{m}{2} - n^2, \quad (5.4)$$

where n is the number of sensors, $m = 2n$ minding that a sensor is represented by a negative and a positive literal couple.

As in this situation only the states describing communication is interesting to us, we remove the uninterpretable cases regarding this model. This number is exactly the number of sensors squared.

Thus a model of a sensor network that has 10 nodes this number is:

$$\frac{20!}{2!18!} - 10^2 = 90. \quad (5.5)$$

Result

The resulting icnf file is then examined with iMiniSat, which can check the assumptions separately. For us, the UNSAT cases are interesting, because the number of those specify how many sensors the examined one can communicate with directly.

Let x be the number of UNSAT cases, this can be calculated with the following formula

$$K(1) = \frac{x - (n - 1)}{n - 2}, \quad (5.6)$$

where n is the number of sensors.

The previous example, checking for the first sensor, there are 41 UNSAT cases, thus the first sensor can communicate with 4 sensors directly, according to

$$K(1) = \frac{41 - (10 - 1)}{10 - 2}. \quad (5.7)$$

5.3. 3rd examination

Model

Extending the second model with assigning a negative literal to cliques, we can define the number of sensors in various cliques a certain sensor can communicate with. With this we can actually enable or disable certain cliques for a given sensor.

The cliques – defined previously – extended with the identifiers assigned to them.

(8, 2, 10, 3, 6, -11)
 (8, 4, 1, 5, 7, -12)
 (8, 4, 6, -13)
 (9, 10, 3, 6, -14)

The first clique (8, 2, 10, 3, 6, -11) that has a communication restriction on the 2nd degree, makes:

8 2 -11 0
 8 10 -11 0
 8 3 -11 0
 8 6 -11 0
 2 10 -11 0
 2 3 -11 0
 2 6 -11 0
 10 3 -11 0
 10 6 -11 0
 3 6 -11 0

Result

Checking sensor 8, with only adding its representational clause $(-8 \ 0)$ defined previously to the model, the number of UNSAT cases is 73. It doesn't change the result, if we add the identifying singleton clause $(14 \ 0)$ for the fourth clique to the model, because the sensor 8 is not a member of the fourth clique, thus the sensors in there do not participate in its communication directly. That means that it does not pose a restriction for it if we switch off that clique.

Extending the model with the singleton identifying clause $(12 \ 0)$ for the second clique:

```
-8 0
14 0
12 0
```

The number of UNSAT cases is 41, which means that can communicate directly with 4 sensors in the second clique.

Extending the model with the singleton identifying clause $(13 \ 0)$ for the second clique:

```
-8 0
14 0
12 0
13 0
```

The number of UNSAT cases is 49, which means that it can communicate directly with 5 sensors in the second and third cliques. We can observe that the sensor 4 can be found in both cliques but regarding communication it is taken into consideration only once.

6. Conclusions and future work

In this paper we introduced two SAT representation that can be used for modeling sensor networks. The power of the first model is that it can provide a fast answer whether it can be ensured that communication between any two sensors is available in sensor networks having several tens of thousands of nodes, and furthermore which sensors must remain in awakened state in case of an alert. The second model is primarily useful for the analysis of sensor networks inner structure. Both model are suitable to examine either global or local properties. In the later we will examine for what opportunities these models might provide in optimizing sensor location and distribution, in the efficiency of energy consumption, and in the assurance of communication.

References

- [1] I.F. AKYILDIZ, W. SU, Y. SANKARASUBRAMANIAM, E. CAYIRCI , Wireless sensor networks: a survey *Computer Networks* 38 (2002) 393 – 422
- [2] BOLIC, MIODRAG, SIMPLOT-RYL, DAVID STOJMENOVIC, IVAN (EDS.) *RFID Systems - Research Trends and Challenges* John Wiley & Sons 576 Pages, 2010.
- [3] A. BHARATHIDASAN, V. A. S. PONDURU, *Sensor Networks: An Overview*
- [4] A. BIÈRE, Lingeling, Plingeling, PicoSAT and PrecoSAT at SAT Race 2010. Technical Report 10/1, FMV Reports Series, JKU, 2010.
- [5] A. BIÈRE, Lingeling and Friends Entering the SAT Challenge 2012. *Haiifa Verification Conference, Department of Computer Science Series of Publications B*, vol. B-2012-2, pp. 33–34, 2012.
- [6] S. A. COOK, The Complexity of Theorem-Proving Procedures. *Proc. of STOC'71*, pp. 151–158, 1971.
- [7] M. DAVIS, G. LOGEMANN, D. LOVELAND, A Machine Program for Theorem Proving. *Commun. ACM*, vol. 5, no. 7, pp. 394–397, 1962.
- [8] C.M. LI AND ANBULAGAN, Look-Ahead versus Look-Back for Satisfiability Problems. *Lecture Notes in Computer Science*, vol. 1330, pp. 342–356, 1997.
- [9] M. W. MOSKEWICZ, C. F. MADIGAN, Y. ZHAO, L. ZHANG, S. MALIK, Chaff: Engineering an Efficient SAT Solver. *Proc. of DAC'01*, pp. 530–535, 2001.
- [10] Y. S. MAHAJAN, Z. FU, S. MALIK, Zchaff2004: An Efficient SAT Solver. *Lecture Notes in Computer Science: Theory and Applications of Satisfiability Testing*, vol. 3542, pp. 360–375, 2005.
- [11] J. P. MARQUES-SILVA, K. A. SAKALLAH, GRASP: A New Search Algorithm for Satisfiability. *Proc. of ICCAD'96*, pp. 220-227, 1996.
- [12] A. BIÈRE, M. HEULE, H. VAN MAAREN, T. WALSH, *Handbook of Satisfiability*. IOS Press, Amsterdam, 2009.