Miskolc Mathematical Notes
Vol. 19 (2018), No. 2, pp. 755–768

HU e-ISSN 1787-2413
DOI: 10.18514/MMN.2018.2140

EQUIVALENCE OF STRONGLY CONNECTED GRAPHS AND BLACK-AND-WHITE 2-SAT PROBLEMS

CSABA BIRÓ AND GÁBOR KUSPER

Received 09 November, 2016

Abstract. Our goal is to create a propositional logic formula to model a directed graph and use a SAT solver to analyse it. This model is similar to the well-know one of Aspvall et al., but they create a directed graph from a 2-SAT problem, we generate a 2-SAT problem from a directed graph. In their paper if the 2-SAT problem is unsatisfiable, then the generated directed graph is strongly connected, in our case, if the directed graph is strongly connected, then the generated 2-SAT problem is a black-and-white 2-SAT problem, which has two solutions: where each variable is true (the white assignment), and where each variable is false (the black one). If we see a directed graph as a communication model of a network, then we can ask in our model whether a node can send a message to another one through the network. More specifically we can ask whether all nodes can send messages to all other ones, i.e., the graph is strongly connected or not.

2010 *Mathematics Subject Classification:* 94C15; 68R10; 03B05; 03B70

Keywords: black-and-white SAT problem, SAT representation, communication graph, connectivity test, strongly connected graph

1. INTRODUCTION

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. By SAT we mean the problem of propositional satisfiability for formulas in conjunctive normal form (CNF). SAT is one of the most-researched NP-complete problems [14] in several fields of computer science, including theoretical computer science, artificial intelligence, hardware design, and formal verification [1]. Modern sequential SAT solvers are based on the Davis-Putnam-Logemann-Loveland (DPLL) [11] algorithm. This algorithm performs Boolean constraint propagation (BCP) and variable branching, i.e., at each node of the search tree it selects a decision variable, assigns a truth value to it, and steps back when a conflict occurs.

The research was supported by the grant EFOP-3.6.1-16-2016-00001 "Complex improvement of research capacities and services at Eszterhazy Karoly University".

By k-SAT we mean the problem of propositional satisfiability for formulas in CNF, where each clause consists of at most k literals. While 2-SAT is solvable in linear time [2], 3-SAT is NP-complete [9]. These days one of the most promising branch of mathematics is the idea, that we try to unify different mathematical theories, like in case of Langlands program [10], which relates algebraic number theory to automorphic forms and representation theory. Another nice example is the modularity theorem [16] (formerly called the Taniyama–Shimura–Weil conjecture), which states that elliptic curves over the field of rational numbers are related to modular forms. Without the modularity theorem Andrew Wiles could not prove Fermat's Last Theorem [17].

In this paper we show a link between directed graphs and propositional logic formulas. We prove a theorem which allows to use an algorithm from the field of propositional logic to check a graph property. Namely, we transform a graph into a SAT problem to check whether the graph is strongly connected or not.

The most prominent graph representations are:

- Implication graph [2] is a is a skew-symmetric directed graph, where vertices are literals (boolean variables, and their negation), edges represents implication. Note, that the binary clause $x \vee y$ is represented by two implications in the implication graph: $\neg x \supset y$, and $\neg y \supset x$, and so the implication graph is skew-symmetric, i.e., it is isomorphic to its own transpose graph.
- AIG, And-Inverter Graph [7] is directed acyclic graph where vertices are logical conjunction with two input edges, a marked edge means logical negation, the boolean variables are the input, the formula itself is the output.
- BDD, Reduced Ordered Binary Decision Diagram [5], which is a rooted, directed, acyclic graph consisting of vertices, which are boolean variables and terminal vertices, called 0-terminal, which terminates pathes where the formula evaluates to false; and 1-terminal, which terminates pathes, where the formula evaluates to true. Each non-terminal vertex has two child vertices called low child, corresponding edge is called 0-edge; and high child, corresponding edge is called 1-edge; which are possible values of the parent vertex. One has to merge any isomorphic subgraph and eliminate any vertex whose two children are isomorphic.
- ZDD (called also ZBDD in the literature), Zero-Suppressed Binary Decision Diagram [12], is a kind of binary decision diagram, where instead of the rule "eliminate any vertex whose two children are isomorphic" we use the rule "eliminate those vertices whose 1-edge points directly to 0-terminal". If a SAT problem has only a few solutions then ZDD is a better representation than BDD.

As we can see a great effort has been done in the direction from formulas to graphs. In this paper we study the other way, the direction from graphs to formulas. In our model vertices are boolean variables, and edges are implications. This means that our

model is similar to an implication graph, but in case of implication graphs vertices are literals. The intuition behind our model comes from the field of wireless sensor networks (WSN), where it is a relevant problem whether each sensor can communicate with all other ones through the network. If the network is represented by a directed graph where vertices are the sensor, and an edge represents that a sensor can send data to an other one, then this problem boils down to check whether the graph is strongly connected.

We wanted to solve this problem by a SAT solver, so we had to convert the above directed graph into a SAT problem. Since in our model each edge represents a logical implication we can generate a 2-SAT problem where each clause contains exactly one positive and one negative literal.

We have found that the graph is strongly connected if this 2-SAT problem has exactly two solutions: the first is the one where each boolean variable is true (which is called the white assignment), the second is the one where each boolean variable is false (which is called the black assignment). We call such a SAT problem to be a "black-and-white" SAT problem.

This means that if we add the negation of these two solutions (the black one and the white one) to the generated SAT problem, then it will be unsatisfiable and will be not 2-SAT any more.

In other words, a SAT problem is a black-and-white SAT problem if and only if it is satisfiable and has only two solutions, the white assignment and the black one. So it becomes unsatisfiable if we add the negation of these assignments, which are two full length clauses, the clause which contains only negative literals (which is called the black clause), and the clause which contains only negative literals (which is called the white clause).

In the field of directed graphs the problem to check strongly connectedness is a linear time problem [15]. The black-and-white 2-SAT problem is also a linear time problem as we are going to show that later in this paper. The question arises, while should we transform a graph into a SAT problem to check a property which can be checked in linear time in both fields? We think that our model is a new link between the two fields: We learned that black-and-white 2-SAT problems and strongly connected graphs are equivalent. The black-and-white SAT problem appears also as a special case of weakly nondecisive SAT problems, see Lemma 6. in [3] (in that paper we did not use the term "black-and-white SAT problem", this term is introduced in this paper). This suggests two things: the black-and-white SAT problem could be an interesting problem in general, and since there are weakly nondecisive 3-SAT problems, which are also black-and-white, there might be a 3-SAT representation of directed graphs.

2. Logical model of a directed graph

Let $\mathcal{D} = (\mathcal{V}, \mathcal{E})$ be a directed graph, where $\mathcal{V}$ is the set of vertices, and E is the set of edges. We say that $\mathcal{D}$ is a communication graph if and only if the elements of $\mathcal{V}$ are atomic formulas or each vertex is labeled by a different atomic formula. Note, that any directed graph is a communication graph because we can label each vertex by a different atomic formula. In this paper we assume that the elements of $\mathcal{V}$ are atomic formulas. In other words, if x is an element of $\mathcal{V}$, then for example $\neg x$ must not be an element of $\mathcal{V}$. From a communication graph we create the following model: We represent vertices by boolean variables, and edges by logical implication. The conjunction of these formulas gives the logical model of the directed graph. For example, if the vertex x_1 has edges to vertices x_2 and x_3, then the logical model is:

$$(x_1 \supset x_2) \wedge (x_1 \supset x_3). \tag{2.1}$$

This formula can be easily transformed to a 2-SAT problem by rewriting implications by the rule $x \supset y = \neg x \vee y$. Note, that although each edge in $\mathcal{D}$ is interpreted as logical implication, it is not an implication graph, because it does not contain negative literals. In other words, our model is not the same used by Aspvall et al. in [2]. They create a graph from a 2-SAT problem, we generate a 2-SAT problem from a communication graph. In their directed graph each variable is represented by a positive and a negative literal. In our case we have only positive literals in the directed graph.

We give the definition of our model in a more formal way. The 2-SAT representation $\mathcal{M}$ of a communication graph $\mathcal{D}$ is defined as follows:

$$\mathcal{M} := \bigwedge_{i=1, j=1, i \neq j}^{n} \mathcal{P}(x_i, x_j) \tag{2.2}$$

$$\mathcal{P}(x, y) = \begin{cases} \neg x \vee y, & \text{if } x \text{ has an edge to } y \\ \text{True}, & \text{otherwise} \end{cases} \tag{2.3}$$

Note, that $\mathcal{M}$ is a 2-SAT problem and has a nice property, each clause in it has exactly one positive and one negative literal, therefore, $\mathcal{M}$ is satisfiable. It has at least two solutions: one where each variable is true which is called the white assignment, and one where each variable is false which is called the black assignment.

If $\mathcal{M}$ has only these two solutions, and no other one, then $\mathcal{D}$ has an interesting property, namely it strongly connected. To check this we need the white-or-black constraint, which is the disjunction of the white assignment and the black assignment, that states that all vertices can be reached from all other ones. We denote the white-or-black constraint by $\mathcal{C}$ in this paper, and we define it as follows:

$$\mathcal{C} := \bigwedge_{i=1, j=1, i \neq j}^{n} x_i \supset x_j = (x_1 \wedge x_2 \wedge \cdots \wedge x_n) \vee (\neg x_1 \wedge \neg x_2 \wedge \cdots \wedge \neg x_n) \tag{2.4}$$

Its negation is called the black-and-white constraint, which is the conjunction of the black clause and the white clause:

$$\neg\mathcal{C} = (\neg x_1 \vee \neg x_2 \vee \cdots \vee \neg x_n) \wedge (x_1 \vee x_2 \vee \cdots \vee x_n) \tag{2.5}$$

By the formula $\mathcal{C}$ we state that there is a path from any vertex to any other one. To show that $\mathcal{D}$ is strongly connected we have to show that the logical representation of $\mathcal{D}$, which is $\mathcal{M}$, implies $\mathcal{C}$. Now we can use the fact that SAT solving is the dual of theorem proving. So instead of proving that "$\mathcal{D}$ is strongly connected if $\mathcal{M}$ implies $\mathcal{C}$", we prove that "$\mathcal{D}$ is strongly connected if $\mathcal{M} \wedge \neg\mathcal{C}$ is unsatisfiable". This means that we can use a SAT solver to check a property of a graph.

Motivated by this we define the following notion: F is a black-and-white SAT problem if and only if F is satisfiable and has exactly two solutions, the white and the black assignments, which implies that $F \wedge \neg\mathcal{C}$ is unsatisfiable. F is a black-and-white 2-SAT problem if and only if F is a 2-SAT and a black-and-white SAT problem. Lemma 6 in [3] suggests an alternative, more general definition: F is a black-and-white SAT problem if and only if F is satisfiable and has exactly two solutions, A and B such that $A = \neg B$. But we do not use this alternative definition in this paper.

First, we need an auxiliary lemma, which states that: there is a path from vertex x_i to x_j if and only if $x_i \supset x_j$ is subsumed by the logical model of the graph.

Lemma 1. *Let $\mathcal{D}$ be a communication graph. Let $\mathcal{M}$ be the 2-SAT representation of $\mathcal{D}$. Then $\mathcal{M}$ implies the formula $x_i \supset x_j$ if and only if there is path from vertex x_i to x_j in graph $\mathcal{D}$.*

Proof. We know that $\mathcal{M}$ is a special SAT instance where each clause contains exactly one positive and one negative literal, hence, each resolvent of clauses from $\mathcal{M}$ is a binary clause with one positive and one negative literal. The main idea of the proof is that if we have two clauses $\neg v_1 \vee v_2$ and $\neg v_2 \vee v_3$, i.e., there is a path in $\mathcal{D}$ from v_1 to v_3, then by resolution we can generate $\neg v_1 \vee v_3$. □

Based on this lemma we can prove the main theorem of this paper which states that the notion of strongly connected graphs and the notion of black-and-white 2-SAT problems are equivalent. This means that the 2-SAT representation of a strongly connected graph is a black-and-white 2-SAT problem, and the other way around, the directed graph representation of a black-and-white 2-SAT problem is a strongly connected graph.

Theorem 1. *Let $\mathcal{D}$ be a communication graph. Let $\mathcal{M}$ be the 2-SAT representation of $\mathcal{D}$. Then $\mathcal{M}$ is a black-and-white 2-SAT problem if and only if the graph $\mathcal{D}$ is strongly connected.*

Proof. Let $\mathcal{D}$ be a communication graph. Let $\mathcal{M}$ be the 2-SAT representation of $\mathcal{D}$. We show that both directions hold.

($\Rightarrow$): The main idea of the proof if the following, if a formula is satisfied by all solutions of a SAT problem, then it is implied by this SAT problem. We know, that $\mathcal{M}$

has only two solutions, the white and the black assignments, both of them satisfy all $x_i \supset x_j$ shaped formulas, i.e., they are implied by $\mathcal{M}$. From this and from Lemma 1. we obtain that in $\mathcal{D}$ there is a path from any vertex to any other one, i.e., $\mathcal{D}$ is strongly connected.

($\Leftarrow$): The main idea of the proof is the following, since $\mathcal{D}$ is strongly connected we know that there is a path $v_1 \supset v_2, v_2 \supset v_3 \ldots, v_{z-1} \supset v_z, v_z \supset v_1$ which is cyclic and contains all vertices from $\mathcal{D}$, the corresponding clause set is $\{(\neg v_1 \vee v_2), (\neg v_2 \vee v_3), \ldots, (\neg v_{z-1} \vee v_z), (\neg v_z \vee v_1)\}$, which is a subset of $\mathcal{M}$, and which can be satisfied only by the white and the black assignments. From this and since each clause in $\mathcal{M}$ is a binary clause, we obtain, that $\mathcal{M}$ is a black-and-white 2-SAT problem. □

To check whether a directed graph is strongly connected or not, we need linear time [15]. We are going to show that its 2-SAT representation can be solved also in linear time.

Theorem 2. *Let F be a black-and-white 2-SAT problem. Then we need linear time to show that $F \wedge \neg C$ is unsatisfiable.*

Proof. The main idea of the proof is that any SAT solver which uses variable branching and BCP, can show that $F \wedge \neg C$ is unsatisfiable by using 1 variable branching and 2 BCP steps as follows: Variable branching will result in a unit. Without lost of generality let us assume that it is a positive one. Then BCP will generate other positive units, because the binary clauses in F contain exactly one positive and one negative literal. BCP will finally result in a conflict, because $\neg C$ contains the negation of the white assignment. Then the other branch will result in a negative unit. This enables a BCP step which will generate negative units, and which will terminate in a conflict because $\neg C$ contains also the negation of the black assignment. Since BCP is a linear time method [19] we need linear time to show that $F \wedge \neg C$ is unsatisfiable. □

Note that DPLL algorithm is a suitable choice to solve the above problem because it uses variable branching and BCP. The question arises, why should we transform a graph into a SAT problem to check a property which can be checked in linear time in both fields? We think that our model is a new link between the two fields which might help to visualize SAT problems. This is not a problem in case of a 2-SAT problem, but in case of 3-SAT it is a problem. This paper and Lemma 6. in [3] together suggest that there might be a 3-SAT representation of directed graphs which is a black-and-white 3-SAT problem. If one finds that representation, then this could help to visualize 3-SAT problems as a directed graph.

3. Our model and the Aspvall model

In this section we show that the constraint used by our model is more general than the one used by the model of Aspvall et al. Fist we show some examples. In Figure 1.

we see a directed graph with 5 vertices. This graph is also a communication graph, since there is no negation sign in the vertices. A directed graph is strongly connected if there is a path between all pairs of vertices. A strongly connected component of a directed graph is a maximal strongly connected subgraph. There are 2 strongly connected components $([1,2,3,5],[4])$ in the following graph.

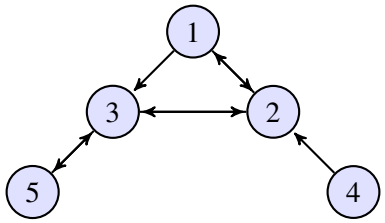

FIGURE 1. A directed graph with 5 vertices

The model of this graph is:

$$\begin{aligned}\mathcal{M} = (\neg x_1 \vee x_2) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_2 \vee x_1) \wedge (\neg x_2 \vee x_3) \wedge \\ (\neg x_3 \vee x_2) \wedge (\neg x_3 \vee x_5) \wedge (\neg x_4 \vee x_2) \wedge (\neg x_5 \vee x_3)\end{aligned} \tag{3.1}$$

From $\mathcal{M}$ we can create $D(\mathcal{M})$ by following the construction defined in [2] by Aspvall et al. Construction steps are the following:

i. For each variable x_i, we add two vertices named x_i and $\neg x_i$ to $D(\mathcal{F}_{\mathcal{M}})$.
ii. For each clause $(x_i \vee x_j)$ of $\mathcal{M}$, we add edges $\neg x_i \to x_j$ and $\neg x_j \to x_i$ to $D(\mathcal{M})$.

In Figure 2. we see the transformed $D(\mathcal{M})$ graph. Aspvall et al. [2] *(Theorem I.)* shows that $\mathcal{M}$ is satisfiable if and only if in $D(\mathcal{M})$ there is no vertex x_i such that it is in the same strong component as its complement $\neg x_i$. We can be sure that this property holds for our model $\mathcal{M}$, because it is always satisfiable.

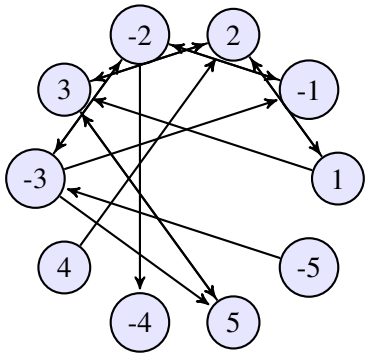

FIGURE 2. $D(\mathcal{F})$, where $\mathcal{F} = \mathcal{M}$

We define the so-called Aspvall kind constraints.

Let $\mathcal{C}_{ASP}(i,j) = (x_i \supset x_j) \wedge (x_i \supset x_j)$, where $1 \leq i,j \leq n$ and $i \neq j$. From this we obtain after simplification that $\neg\mathcal{C}_{ASP}(i,j) = (x_i \vee x_j) \wedge (\neg x_j \vee \neg x_i)$. We can add these clauses to the model to check whether there is a directed path from x_i vertex to x_j and vice versa: $D(\mathcal{M} \wedge \neg\mathcal{C}_{ASP}(i,j))$.

We add two constraints to the example shown in Figure 1. The first one is $\mathcal{C}_{ASP}(1,4) = (x_1 \supset x_4) \wedge (x_4 \supset x_1)$. Let $\mathcal{F} = \mathcal{M} \wedge \neg\mathcal{C}_{ASP}(1,4)$. So we add these two clauses $(x_1 \vee x_4) \wedge (\neg x_1 \vee \neg x_4)$ and these 4 edges, see the red arrows in Figure 3. We see on Figure 3. that in graph $\mathcal{D}(\mathcal{F})$ there exists no vertex x_i which is in the same strong component as its complement $\neg x_i$, hence, $\mathcal{F}$ is still satisfiable.

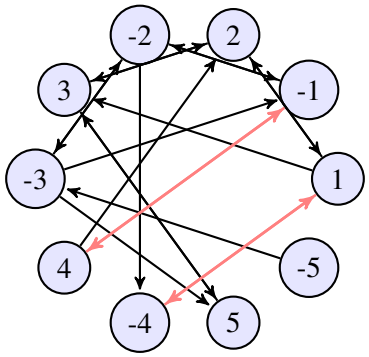

FIGURE 3. $D(\mathcal{F})$, where $\mathcal{F} = \mathcal{M} \wedge \neg\mathcal{C}_{ASP}(1,4)$

Now we add $\mathcal{C}_{ASP}(1,5) = (x_1 \supset x_5) \wedge (x_5 \supset x_1)$. Let $\mathcal{F}' = \mathcal{M} \wedge \neg\mathcal{C}_{ASP}(1,5)$. So we add these two clauses $(x_1 \vee x_5) \wedge (\neg x_1 \vee \neg x_5)$ and these 4 edges, see the red arrows in Figure 4. We can see on Figure 4. that in the graph $\mathcal{D}(\mathcal{F}')$ there exists a vertex x_i which is in the same strong component as its complement $\neg x_i$, hence, $\mathcal{F}'$ is unsatisfiable.

In our approach C is

$$\begin{aligned}(x_1 \supset x_2) \wedge (x_1 \supset x_3) \wedge (x_1 \supset x_4) \wedge \\ (x_2 \supset x_1) \wedge (x_2 \supset x_3) \wedge (x_2 \supset x_4) \wedge \\ (x_3 \supset x_1) \wedge (x_3 \supset x_2) \wedge (x_3 \supset x_4) \wedge \\ (x_4 \supset x_1) \wedge (x_4 \supset x_2) \wedge (x_4 \supset x_3)\end{aligned} \tag{3.2}$$

$$\begin{aligned}(\neg x_1 \vee x_2) \wedge (\neg x_1 \vee x_3) \wedge (\neg x_1 \vee x_4) \wedge (\neg x_1 \vee x_5) \wedge \\ (\neg x_2 \vee x_1) \wedge (\neg x_2 \vee x_3) \wedge (\neg x_2 \vee x_4) \wedge (\neg x_2 \vee x_5) \wedge \\ (\neg x_3 \vee x_1) \wedge (\neg x_3 \vee x_2) \wedge (\neg x_3 \vee x_4) \wedge (\neg x_3 \vee x_5) \wedge \\ (\neg x_4 \vee x_1) \wedge (\neg x_4 \vee x_2) \wedge (\neg x_4 \vee x_3) \wedge (\neg x_4 \vee x_5) \wedge \\ (\neg x_5 \vee x_1) \wedge (\neg x_5 \vee x_2) \wedge (\neg x_5 \vee x_3) \wedge (\neg x_5 \vee x_4)\end{aligned} \tag{3.3}$$

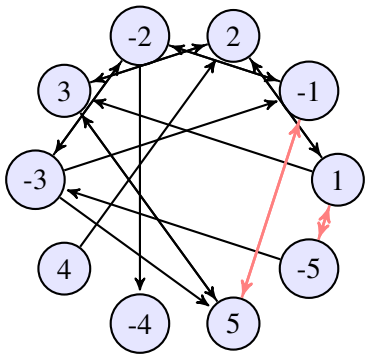

FIGURE 4. $D(\mathcal{F}')$, where $\mathcal{F}' = \mathcal{M} \wedge \neg\mathcal{C}_{ASP}(1,5)$

After simplification $\neg C$ is the following:

$$(x_1 \vee x_2 \vee x_3 \vee x_4 \vee x_5) \wedge (\neg x_1 \vee \neg x_2 \vee \neg x_3 \vee \neg x_4 \vee \neg x_5) \tag{3.4}$$

For any i and j we have that $\neg\mathcal{C}_{ASP}(i,j)$ implies $\neg\mathcal{C}$, thus $\neg\mathcal{C}_{ASP}(i,j)$ subsumes $\neg\mathcal{C}$. This means that $\neg\mathcal{C}_{ASP}(i,j)$ defines a stronger constraint to the model, thus $\neg\mathcal{C}$ defines a more general constraint than $\neg\mathcal{C}_{ASP}(i,j)$ to check whether a graph is strongly connected or not.

4. AN APPLICATION

4.1. *Wireless sensor networks*

Ad hoc wireless sensor networks (WSN) are used widely (for example, in military to observe environment). They have the advantage that they consist of sensors with low energy consumption, which can be deployed easily in a cheap way on areas which are out-of-the-way. These sensors are the nodes of WSN. They are capable of processing some limited information and using wireless communication. A big effort has been to do research on how to deploy them in an optimal way to keep efficient energy consumption and communication. Although there are many WSN solutions, the deployment of a WSN is still an active research field [8].

One of the important property of an ad hoc wireless network is node density. The dense layout makes the following properties available: high fault tolerance, high-coverage characteristics, but also cause some problems. The interference is high near dense node areas, and there are a lot of collisions in the case of messaging, which requires complicated operations of a MAC protocol. Because of too many possible routes, routing needs lots of resources [4].

The aim of monitoring techniques is reducing the cost of the distributed algorithms interpreted on the network. The graph, which represents the network, has to be

thinned because of cost-reduction by techniques like disconnecting of nodes, removing links, changing scopes, etc., but the network-quality characteristics (like scalability, coverage, fault tolerance, etc.) must not fall below a required level. The overall aim is to create a scalable, fault-tolerant rare topology, where the degree of the nodes are low, the maximum load is low, energy consumption is low and the paths are short. The following techniques are used to create an optimal topology: reducing the scope of nodes, removing some nodes, introducing a dominating set of nodes, clustering, and adding some new nodes to gain all-all communication [13, 18].

4.2. *Connectivity test by SAT representation*

If we have a WSN, then we can redefine communication graph as follows, we say that $\mathcal{D}$ is a communication graph if and only if the elements of $\mathcal{V}$, the vertices, represent the sensor nodes of the WSN, and elements of $\mathcal{E}$, the edges, represent a one way communication between two nodes.

We intend to examine which sensors can communicate with which ones, thus we can create the communication graph and its 2-SAT representation. In this approach it can be checked fairly quickly in a given state of a randomly distributed sensor network (assuming all sensors are awaken) whether it can be ensured that each sensor can communicate with every other one [6].

For example let us model a randomly distributed heterogeneous sensor network which has 10 nodes. For representing the input clause set for a SAT solver, the

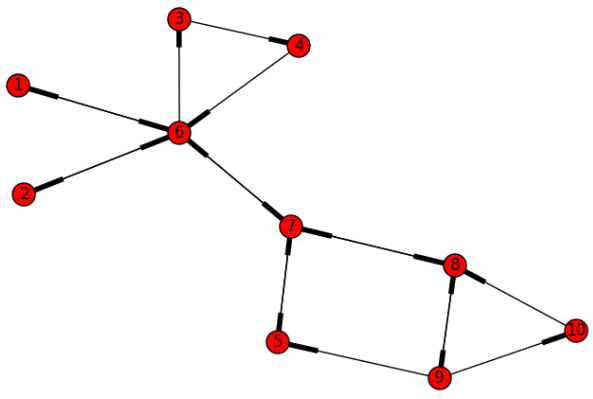

FIGURE 5. Heterogeneous sensor nodes with their communication graph

DIMACS CNF[1] 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”.
DIMACS CNF format is the following: `p cnf 10 20`

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

Our main goal is to examine the produced DIMACS CNF file with `MiniSat 2.2.0`[2], which is a complete SAT solver, which returns unsatisfiable (UNSAT). Thus the model fulfills the requirements, namely communication is ensured between any two nodes. In this example the result is UNSAT, thus the represented communication graph is strongly connected.

This model can also give a more detailed analysis. For example, if we take the sensors out of the model one by one, and with the remaining we test the communication problem again 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 sensors. The significance of this in graph theory is that the removal

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

[2] http://minisat.se

of this node makes the graph not connected anymore. In our example the removal of nodes 3, 4, 5, 6, 7, 8 and 9 would return with SAT, thus the malfunction of those sensors means the communication in that sensor network is inadequate.

It is generally true that if the graph is strongly connected, then a DPLL-based SAT solver returns the following values.

- number of solutions : 0
- number of conflicts : 2
- number of decisions : 1
- number of unit propagations : $u = 2n$, where n the number of literals (number of vertices)

These results also show that the black-and-white 2-SAT problem with the black-and-white constraint can be solved in linear time since the number of decisions is 1, and the number of unit propagations is $2n$.

5. Conclusions and future work

In this paper we have introduced a SAT representation that can be used for modeling directed graphs. The only restriction is that the names of the vertices should be boolean variables. This model, which is a 2-SAT problem, makes it possible to check whether a graph is strongly connected by adding two clauses, the black and the white ones, to the model and asking a SAT solver whether this formula is unsatisfiable. The two clauses are a constraint which state that it is not true that there is a path from any vertex to any other one. This constraint is more general than the one used by Aspvall et al. We have shown that the representation of a strongly connected graph is a black-and-white 2-SAT problem. The black-and-white SAT problem appears also as a special case of weakly nondecisive SAT problems, see Lemma 6. in [3]. This suggests two things: the black-and-white SAT problem could be an interesting problem in general, and since there are weakly nondecisive 3-SAT problems, which are also black-and-white, there might be a 3-SAT representation of directed graphs. Our model can be applied also in a natural way to networks of sensors, such as wireless sensor networks. If we use a SAT solver to solve our model, then it gives back not only the solution but also lots of numbers. An interesting question is that how one can use these metrics, like *number of solutions, conflicts, decisions, unit propagations*, to say something about the topology of the graph.

It seems the the following generalized notion is also interesting: F is a generalized black-and-white SAT problem if and only if F is satisfiable and for any assignment A: if A is a solution of F, then F has another solution B such that $A = \neg B$. For example, the empty clause set is a generalized black-and-white SAT problem. Another interesting example is the set of black and the white clauses. We are going to investigate this notion in a future study.

REFERENCES

[1] A. Biere, M. Heule, H. van Maaren, T. Walsh, *Handbook of Satisfiability*. Amsterdam: IOS Press, 2009.

[2] Bengt Aspvall, Michael F. Plass and Robert Endre Tarjan, "A Linear-Time Algorithm For Testing The Truth Of Certain Quantified Boolean Formulas," *Information Pprocessing Letters*, vol. 8, no. 3, pp. 121–123, 1979, doi: 10.1016/0020-0190(79)90002-4.

[3] C. Biró, G. Kusper, and T. Tajti, "How to generate weakly nondecisive sat instances," in *2013 IEEE 11th International Symposium on Intelligent Systems and Informatics (SISY)*," doi: 10.1109/SISY.2013.6662583, Sept 2013, pp. 265–269.

[4] Bolic, Miodrag, Simplot-Ryl, David Stojmenovic, Ivan (eds.), *RFID Systems - Research Trends and Challenges*. New York: John Wiley & Sons, 2010.

[5] R. E. Bryant, "Graph-based algorithms for boolean function manipulation," *IEEE Trans. Comput.*, vol. 35, no. 8, pp. 677–691, Aug. 1986, doi: 10.1109/TC.1986.1676819. [Online]. Available: http://dx.doi.org/10.1109/TC.1986.1676819

[6] Cs. Biro, G. Kusper, T. Radvanyi, S. Kiraly, P. Szigetvary, P. Takacs, "SAT Representation of Randomly Deployed Wireless Sensor Networks," *Proc. of ICAI09*, vol. 2, pp. 101–111, 2014, doi: 10.14794/ICAI.9.2014.2.101.

[7] L. Hellerman, "A catalog of three-variable or-invert and and-invert logical circuits," *IEEE Transactions on Electronic Computers*, vol. EC-12, no. 3, pp. 198–223, June 1963, doi: 10.1109/PGEC.1963.263531.

[8] I.F. Akyildiz, W. Su, Y. Sankarasubramaniam, E. Cayirci, "Wireless sensor networks: a survey ," *Computer Networks*, vol. 38, no. 4, pp. 393–424, 2002, doi: doi:10.1016/S1389-1286(01)00302-4.

[9] R. M. Karp, *Reducibility among Combinatorial Problems*. Boston, MA: Springer US, 1972, pp. 85–103, doi: 10.1007/978-1-4684-2001-29.

[10] R. P. Langlands, *Problems in the theory of automorphic forms to Salomon Bochner in gratitude*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1970, pp. 18–61, doi: 10.1007/BFb0079065.

[11] M. Davis, G. Logemann, D. Loveland, "A Machine Program for Theorem Proving," *Commun. ACM*, vol. 5, no. 7, pp. 394–397, 1962, doi: 10.1145/368273.368557.

[12] S.-i. Minato, "Zero-suppressed bdds for set manipulation in combinatorial problems," in *Proceedings of the 30th International Design Automation Conference*, ser. DAC '93, doi: 10.1145/157485.164890. New York, NY, USA: ACM, 1993, pp. 272–277. [Online]. Available: http://doi.acm.org/10.1145/157485.164890

[13] Paolo Santi, "Topology Control in Wireless Ad Hoc and Sensor Networks ," *ACM Computing Surveys* , vol. 37, no. 2, pp. 164–194, 2005, doi: 10.1145/1089733.1089736.

[14] S. A. Cook, "The Complexity of Theorem-Proving Procedures," *Proc. of STOC'71*, pp. 151–158, 1971, doi: 10.1145/800157.805047.

[15] M. Sharir, "A strong-connectivity algorithm and its applications in data flow analysis," *Computers And Mathematics with Applications*, vol. 7, no. 1, pp. 67 – 72, 1981, doi: http://dx.doi.org/10.1016/0898-1221(81)90008-0.

[16] A. Weil, "Über die bestimmung dirichletscher reihen durch funktionalgleichungen," *Mathematische Annalen*, vol. 168, no. 1, pp. 149–156, 1967, doi: 10.1007/BF01361551.

[17] A. J. Wiles, "Modular elliptic curves and fermat's last theorem," *ANNALS OF MATH*, vol. 141, pp. 443–551, 1995, doi: 10.2307/2118559.

[18] Yu Wang, "Topology Control for Wireless Sensor Networks," *Springer - Wireless Sensor Networks and Applications*, vol. 148, no. 2, pp. 113–147, 2008, doi: 10.1007/978-0-387-49592-7.

[19] H. Zhang and M. E. Stickel, "An efficient algorithm for unit propagation," in *In Proceedings of the Fourth International Symposium on Artificial Intelligence and Mathematics (AI-MATH'96)*, 1996, pp. 166–169.

Authors' addresses

Csaba Biró
Eszterházy Károly University, Hungary, Institute of Mathematics and Informatics
E-mail address: biro.csaba@uni-eszterhazy.hu

Gábor Kusper
Eszterházy Károly University, Hungary, Institute of Mathematics and Informatics
E-mail address: kusper.gabor@uni-eszterhazy.hu