G. Jäger

Computer Science Institute, University of Kiel, D-24118 Kiel, Germany

An Effective SAT Encoding for Magic Labeling

Key words: Magic Labeling, Boolean Satisfiability, Backtracking.

1 Introduction

This work presents a Boolean satisfiability (SAT) encoding for a special problem from combinatorial optimization. In the last years much progress has been made in the optimization of practical SAT solvers (see the SAT competition [5]). This has made SAT encodings for combinatorial problems highly attractive. In this work we propose an encoding for the combinatorial problem *Magic Labeling* which has important applications in the field of wireless networks [4]. It is defined as follows. Let an undirected, unweighted graph $G = (V, E)$ be given with vertex set V and edge set E, where $|V| = n$ and $|E| = m$. A *labeling* is a one-to-one mapping $\lambda : V \cup E \to \{1, 2, \ldots, m + n\}$. Define the *weight* $\omega(e)$ of an edge $e \in E$ as the sum of the label of e and of the labels of its two endpoints. An *edge-magic total labeling (EMTL)* is a labeling λ for which a constant $h \in \mathbb{N}$ exists such that $\omega(e) = h$ for each edge $e \in E$. Similarly, define the *weight* $\omega(v)$ of a vertex $v \in V$ as the sum of the label of v and of the labels of all edges incident to v. A *vertex-magic total labeling (VMTL)* is a labeling λ for which a constant $k \in \mathbb{N}$ exists such that $\omega(v) = k$ for each vertex $v \in V$. Finally, a *totally magic labeling (TML)* is a labeling λ for which (not necessarily equal) constants $h, k \in \mathbb{N}$ exist such that λ is edge-magic with constant h and vertex-magic with constant k. h and k are called *magic constants.* A vertex $v \in V$ and an edge $e \in E$ are denoted *neighboring*, if e is incident to v. Note that different EMTLs exist for the same graph, and the same holds for VMTLs. Surveys of results for magic graphs are given in [4]. We consider the following three problems for a given graph:
(1) Does an EMTL exist with given magic constant $h \in \mathbb{N}$?
(2) Does a VMTL exist with given magic constant $k \in \mathbb{N}$?
(3) Does a TML exist with given magic constants $h, k \in \mathbb{N}$?

The obvious method for these problems would be to use a backtracking approach. In Section 2 we propose a general algorithm based on a SAT encoding and extend

Email address: `gej@informatik.uni-kiel.de` (G. Jäger).

the encoding and the resulted algorithm to the following problems:
(4) Enumerate all EMTLs with given magic constant $h \in \mathbb{N}$?
(5) Enumerate all VMTLs with given magic constant $k \in \mathbb{N}$?
(6) Enumerate all TMLs with given magic constants $h, k \in \mathbb{N}$?
In Section 3 we compare the performance of this SAT based algorithm and of a backtracking algorithm for the problems (1) and (2).

2 SAT Encoding for Magic Labeling

In this section we consider the problems (1) to (3). Let $G = (V, E)$ with $|V| = n$, $|E| = m$ and $r := n + m$. For our convenience, we define a fixed ordering on the set $V \cup E$ by the numbers $1, 2, \ldots, n + m$, i.e., each number of $\{1, 2, \ldots, n + m\}$ represents an edge or a vertex of the graph. For the encoding we use r^2 Boolean variables $x_{i,j}$ with $1 \le i, j \le r$, where we set $x_{i,j} =$
$\begin{cases} \textsc{True}, & \text{if edge/vertex } i \text{ receives label } j \\ \textsc{False}, & \text{if edge/vertex } i \text{ does not receive label } j \end{cases}$
Labeling Clauses: For receiving a feasible labeling we need the following conditions. First each edge/vertex needs to have exactly one label. This leads to the condition that for $i = 1, 2, \ldots, r$ exactly one $j \in \{1, 2, \ldots, r\}$ exists with $x_{i,j} = \textsc{True}$. Second each label has to be used by exactly one edge/vertex. This leads to the condition that for $j = 1, 2, \ldots, r$ exactly one $i \in \{1, 2, \ldots, r\}$ exists with $x_{i,j} = \textsc{True}$. All $2r$ restrictions have the same structure, namely that exactly one of the r involved Boolean variables is set to TRUE and the rest to FALSE. To represent this, we introduce $2r^2$ auxiliary variables $y_1, y_2, \ldots, y_{2r^2}$, with r y's for one restriction. W.l.o.g., consider the first restriction, which contains the Boolean variables $x_{1,1}, x_{1,2}, \ldots, x_{1,r}$, and the corresponding auxiliary variables $y_1, y_2, \ldots, y_r$. For $1 \le k \le r$ we use y_k to represent that at least one of $x_{1,1}, x_{1,2}, \ldots, x_{1,k}$ is TRUE. Precisely, the y variables are defined as $y_1 = x_{1,1}$ or equivalently $(\neg x_{1,1} \vee y_1) \wedge (x_{1,1} \vee \neg y_1)$, and $y_k = x_{1,k} \vee y_{k-1}$ or equivalently $(y_k \vee \neg x_{1,k}) \wedge (y_k \vee \neg y_{k-1}) \wedge (\neg y_k \vee x_{1,k} \vee y_{k-1})$ for $k = 2, 3, \ldots, r$. In addition, we need to enforce that only one $x_{1,i}$ with $1 \le i \le r$ can be TRUE. This means, if $x_{1,k}$ is TRUE, none of the x_{1i} for $1 \le i < k \le r$ can be TRUE. This is formulated as $\neg y_{k-1} \vee \neg x_{1k}$ for $k = 2, \ldots, r$. Finally y_r must be TRUE.
Magic Clauses: Furthermore we have to add clauses which ensure that the conditions of EMTL/ VMTL/TML are fulfilled. The following two conditions occur:
EMTL/TML: Set $l := 2$. For given $h \in \mathbb{N}$ and a given edge the sum of $l+1$ labels (namely the label of the edge and of its l endpoint vertices) equals h.
VMTL/TML: For given $k \in \mathbb{N}$ and a given vertex with degree $l \in \mathbb{N}$ the sum of $l + 1$ labels (namely the label of the vertex and of its l incident edges) equals k.
Observe that both conditions have the following structure: For given constants $c, l \in \mathbb{N}$ the sum of $l + 1$ labels equals c. For $l \in \mathbb{N}$ let W be the set containing all possible l-tuples $\overrightarrow{w} = (w_1, w_2, \ldots, w_l)$ with $w_i \in \{1, 2, \ldots, r\}$ for $1 \le i \le l$ and $w_i \ne w_j$ for $1 \le i < j \le l$. Now let a constant $c \in \mathbb{N}$ be given and an edge or

vertex f with corresponding $l \in \mathbb{N}$, i.e., if f is an edge, then $l = 2$, and otherwise l is the degree of f. We want to fulfill the magic condition for f. This means that the sum of the label of f and of its neighboring elements is c. Let $f_1, f_2, \ldots, f_l$ be the neighboring elements of f. For this l compute the set W (which is easy for small l) and choose an arbitrary element $\overrightarrow{w} \in W$ with $w := \sum_{i=1}^{l} w_i$. Then label $f_1, f_2, \ldots, f_l$ by $w_1, w_2, \ldots, w_l$, and consider the four cases:
Case 1**:** $i \in \{1, 2, \ldots, l\}$ exists with $c-w = w_i$; **Case** 2**:** $w \geq c$; **Case** 3**:** $w < c-r$; **Case** 4**:** Otherwise.
As all labels are different and are contained in the set $\{1, 2, \ldots, r\}$, it is clear that for the Cases 1, 2, or 3 no labeling of f exists such that the sum of the labels of $f, f_1, f_2, \ldots, f_l$ is c. In these cases we add the clause $\neg x_{f_1,w_1} \vee \neg x_{f_2,w_2} \vee \cdots \vee \neg x_{f_l,w_l}$ meaning that labeling $f_1, f_2, \ldots, f_l$ by $w_1, w_2, \ldots, w_l$ is not possible. For Case 4 such a labeling is possible, but only if f is labeled with $c - w$. This leads to the clause $\neg x_{f_1,w_1} \vee \neg x_{f_2,w_2} \vee \cdots \vee \neg x_{f_l,w_l} \vee x_{f,c-w}$. Thus for each $\overrightarrow{w} \in W$ we have an additional clause. Clearly, the number of possible sums and therefore the number of magic clauses becomes rather large, if we consider VMTLs or TML for dense graphs. In these cases the resulted algorithm has bad performance (see Section 3).
Enumerating All Magic Labelings: The SAT based representation allows us to enumerate all magic labelings using a technique of Jin, Han, Somenzi [3], which is applicable to general SAT instances. The main idea of this technique is to add new clauses to a SAT model with purpose to enumerate all SAT solutions. In our case we start with the presented SAT encoding. If this SAT encoding is satisfiable, we receive a first magic labeling $\lambda : \{1, 2, \ldots, r\} \rightarrow \{1, 2, \ldots, r\}$. This means that in the SAT solution exactly the Boolean variables $x_{1,\lambda(1)}, x_{2,\lambda(2)}, \ldots, x_{r,\lambda(r)}$ are set to TRUE. Then we explicitly forbid this magic labeling by adding the new clause $\neg x_{1,\lambda(1)} \vee \neg x_{2,\lambda(2)} \vee \cdots \vee \neg x_{r,\lambda(r)}$ to the current SAT instance. For the updated SAT instance there are two possibilities. If the instance is satisfiable, this leads to another magic labeling, as the first one is not allowed. If not, the first magic labeling was the only one. This process can be iterated, until all or a determined number of magic labelings has been found.

3 Experimental Results

In this section we compare our algorithm (called SAT-MAGIC) with a natural backtracking algorithm (called BACK-MAGIC). In BACK-MAGIC all vertices and edges are labeled in a fixed order, and if a partial labeling makes a magic labeling impossible, then a backtracking step occurs, i.e., a previous labeling of a edge/vertex is changed, and the search continues at this step. All algorithms have been implemented in C++, where we make use of an effective SAT solver implemented by Eén and Sörensson, called MINISAT [2]. The experiments were carried out on a PC with an Athlon 1900MP CPU with 2GB of memory. We test random graphs with size $n = 10, 15$, where $p = 10\%, 20\%, 30\%, 40\%$ edges are

Size n	10								15							
Type	EMTL				VMTL				EMTL				VMTL			
Density p (%)	10	20	30	40	10	20	30	40	10	20	30	40	10	20	30	40
Suc. Back (%)	40	20	10	0	100	0	0	0	0	0	0	0	0	0	0	0
Suc. Sat (%)	100	100	100	80	100	100	0	0	80	60	50	0	100	0	0	0

Table 1
Comparison of Sat Magic and Back Magic

chosen randomly and uniformly distributed from all possible $n \cdot (n-1)/2$ ones. As only 3 connected TMLs are known [1], we do not consider TMLs, but only EMTLs and VMTLs. Note that for each single instance we can easily compute a lower bound $lb \in \mathbb{N}$ and an upper bound $ub \in \mathbb{N}$ for possible magic constants. Then we choose $\min\{10, ub - lb + 1\}$ values of this interval $[lb, ub]$ and for each value we receive a single instance of the form (1) or (2). Thus we have $16 = 2 \cdot 2 \cdot 4$ test classes, where each test class consists of up to 10 single instances.
In Table 3 for both algorithms and for each test class a percentage value is given describing how many instances of this test class can be solved in 600 seconds. The results clearly demonstrate the superiority of Sat-Magic in comparison to Back-Magic. As expected, Sat-Magic behaves rather bad for VMTLs with large density.

References

[1] A. Baker, J. Sawada: *Magic Labelings on Cycles and Wheels.* In B. Yang, D.-Z. Du, C.A. Wang (Eds.): Proc. 2nd Annual International Conference on Combinatorial Optimization and Applications (COCOA). Lecture Notes in Comput. Sci. **5165**, 361-373, 200

[2] N. Eén, N. Sörensson: *An Extensible SAT-Solver.* In E. Giunchiglia, A. Tacchella (Eds.): Proc. 6th International Conference on Theory and Applications of Satisfiability Testing (SAT). Lecture Notes in Comput. Sci. **2919**, 502-518, 2003.

[3] H. Jin, H. Han, F. Somenzi: *Efficient Conflict Analysis for Finding All Satisfying Assignments of a Boolean Circuit.* In N. Halbwachs, L.D. Zuck (Eds.): Proc. 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS). Lecture Notes in Comput. Sci. **3440**, 287-300, 2005.

[4] W.D. Wallis: *Magic Graphs.* Birkhäuser, Boston, 2001.

[5] International SAT Competition: "http://www.satcompetition.org/".