The SAT01 framework for NP problems [*]

Stanislav Busygin
busygin@gmail.com
http://www.stasbusygin.org

Abstract

We consider an NP-complete problem SAT01 having a range of remarkable properties. First, it is equivalent to the weighted independent set problem on a graph Γ with vertex weights w, where the required independent set weight equals the maximum possible $\kappa(\Gamma, w)$ value m, and hence is decidable by computation of the weighted Lovász number unless $\alpha(\Gamma, w) < \vartheta(\Gamma, w) = \kappa(\Gamma, w) = m$. Second, it admits a suitable constraint propagation technique able to simplify many SAT01 instances. Third, a multitude of NP problems reduce to SAT01 in a natural way, without excessive dimensionality growth. At that, the obtained SAT01 formulation tends to have a clear interpretation within notion of the original problem. Finally, we outline a rationale for choosing SAT01 as the framework for NP problems from an information theory viewpoint.

Keywords: SAT01, NP-complete, maximum weight independent set problem, Lovász number, constraint propagation, polynomial reduction, SAT, Hamilton cycle problem, graph isomorphism, subgraph isomorphism, extended 15-puzzle, Hanoi tower puzzle, blocksworld problem, factoring.

1 Introduction

The problem we are going to consider may be seen from two viewpoints. On one hand, it is a variation of boolean satisfiability, but on the other hand it comes from 0-1 linear programming. Both viewpoints are useful and will be utilized in this text. So, to avoid confusions we should declare some specific conventions we are going to use.

First, we identify the boolean domain $\{false, true\}$ with $\{0, 1\}$. This means that any arithmetic operation may be applied to truth values (with a numerical resulting value possibly outside $\{0, 1\}$) and any logical operation may be considered over $\{0, 1\}$. Also, it means that an expression may mix arithmetic and logical operations. For the sake of simplicity, we will not use truth values at all, always substituting 0 instead of $false$ and 1 instead of $true$ even in pure logical formulas. Next, we do not make difference between clauses of conjuctive forms and equations/inequalities of systems over 0-1 variables. Indeed, equations and inequalities comprising a system are joined by $\wedge$-operator. On the other hand, a logical clause is nothing more than an $f(x) = 1$ equation over 0-1 variables.

Taking into account these remarks, we define *SAT1* problem as a boolean satisfiability problem whose clauses have the form $\sum_k x_k = 1$ (so, *1* after *SAT* stands for the requirement of exactly one true variable per clause.) In the matrix form, we are given a 0-1 matrix $A = (a_{ij})_{m\times n}$ and it is asked whether there is a 0-1 vector $x = (x_j)_{j=1\ldots n}$ satisfying

$$Ax = e, \tag{1}$$

[*]Copyright © Stanislav Busygin, 2003-2007. All rights reserved.

where e is the all-one vector ($a_{ij} = 1$ if x_j is in the i-th clause and $a_{ij} = 0$ otherwise.) SAT1 is NP-complete. It follows immediately from NP-completeness of "one-in-three" satisfiability [4] (that corresponds to SAT1 with all clauses having length 3.) Another interpretation of SAT1 is the *set partitioning problem with indifferent objective.* Namely, we can consider columns of matrix A to be subsets M_j of the *ground set* $\{1 \dots m\}$ such that $i \in M_j$ iff $a_{ij} = 1$. Then, any SAT1 solution x determines a partition of the ground set into the subsets $\{M_j : \ x_j = 1\}$. We mention that the optimization set partitioning with a linear objective $cx \to \min$ is a well-known NP-hard problem of 0-1 programming, and was intensively studied, e.g. [3, 10, 12].

Next, we define an extension of SAT1 allowing clauses of the form $(\overline{x_j} \vee \overline{x_k})$ expressing *contradictions* among variable pairs. We name this extended problem *SAT01* (*0* stands for possibility to have zero true variables in these new clauses.) Note that if x_j and x_k are contained in some common equation of (1), the relation $(\overline{x_j} \vee \overline{x_k})$ trivially holds. So, such a 2-clause is meaningful only if the subsets M_j and M_k do not intersect. Since the pairwise contradictions of variables play the crucial part in SAT01 framework, we introduce a special object expressing the whole set of these relations. The *contradiction graph* $\Gamma(X, B)$ is a graph of n vertices $X = \{1 \dots n\}$ corresponding to the SAT01 variables and there is an edge $(j, k) \in B$ iff x_j and x_k appear together in a common clause (i.e. either in an equation of (1) or as $(\overline{x_j} \vee \overline{x_k})$ clause.) Also, for a 0-1 vector x of length n define

$$I(x) = \{j : \ x_j = 1\}. \tag{2}$$

Obviously, if x satisfies (1) and $I(x)$ is an independent set of the contradiction graph, then x is a solution to SAT01. So, a 0-1 $m \times n$ matrix A and a graph $\Gamma(X, B)$, $X = \{1 \dots n\}$ defines an instance SAT01(A, Γ):

$$\begin{cases} Ax = e, \\ I(x) \text{ is an independent set of } \Gamma(X, B). \end{cases} \tag{3}$$

If there is no solution to (3) with $x_j = x_k = 1$, we say that x_j and x_k are *contradictory* to each other. In particular, if $(j, k) \in B$, then x_j and x_k will be called *contradictory á priori.*

Claim 1 *If x satisfies (3) and each variable x_j is contained in at least one of the equations:*

$$\forall j \in \{1 \dots n\} \ \exists i \in \{1 \dots m\} \ a_{ij} = 1, \tag{4}$$

then $I(x)$ is a maximal independent set of $\Gamma(X, B)$.

Proof. Assume $I(x)$ is not maximal and there is $k \notin I(x)$ such that $\forall j \in I(x) \ (j, k) \notin E$. Then $x_k = 0$ and it is contained in at least one equation of (3). However, this equation is satisfied and so some $x_j \in I(x)$ is contained in it too. Thus, $(j, k) \in B$ and hence $I(x)$ must be maximal. QED.

Throughout the paper we will consider only SAT01 instances obeying (4). Otherwise, without loss of satisfiability the instance may be reduced assigning all variables not contained in equations to 0.

2 SAT01 as maximum weight independent set problem

In the contradiction graph $\Gamma(X, B)$ we introduce vertex weights $w = (w_j)_{j=1 \dots n}$ such that

$$w_j = \sum_{i=1}^{m} a_{ij}. \tag{5}$$

In other words, vertex j has a weight equal to the number of equations containing x_j, or w_j is cardinality of the subset M_j. As usually, denote the weighted independence number of $\Gamma(X, B)$ by $\alpha(\Gamma, w)$.

Lemma 1 $\alpha(\Gamma, w) \leq m$. *There is a one-to-one correspondence between solutions to SAT01*(A, Γ) *and* $\Gamma(X, B)$ *independent sets of weight* m.

Proof. Let $S \subseteq X$ and $\sum_{j \in S} w_j > m$. By the definition of w, it is equivalent to

$$\sum_{j \in S} \sum_{i=1}^{m} a_{ij} > m.$$

As A is a 0-1 matrix, it means that there are totally more than m 1-entries in the columns, whose indices are in S. However, there are only m rows. So, according to the pigeonhole principle, there is at least one row with more than one 1-entry in those columns. Thus, S contains at least one pair of adjacent vertices and so it is not an independent set. Hence $\alpha(G, w) \leq m$.

Since a SAT01(A, Γ) solution x defines a partition of the m-element ground set into subsets $\{M_j : \; x_j = 1\}$ and $w_j = |M_j|$, it is easy to see that $I(x)$ has the weight m. Conversely, any independent set $S \subseteq X$ of weight m uniquely defines a SAT01(A, Γ) solution x, where $x_j = 1$ if $j \in S$ and $x_j = 0$ if $j \in X \setminus S$. At that, $S = I(x)$. So, there is a bijection between SAT01 solutions x and the independent sets $I(x)$ having the weight m. QED.

Now we establish the main result on the weighted Lovász number $\vartheta(\Gamma, w)$ of SAT01 contradiction graphs. Recall that for any graph G the "sandwich" relation

$$\alpha(G, w) \leq \vartheta(G, w) \leq \kappa(G, w) \tag{6}$$

holds, where

$$\begin{aligned} \alpha(G, w) &= \max_x \{w^T x \mid x \in \mathcal{STAB}(G)\}, \\ \vartheta(G, w) &= \max_x \{w^T x \mid x \in \mathcal{TH}(G)\}, \\ \kappa(G, w) &= \max_x \{w^T x \mid x \in \mathcal{QSTAB}(G)\}, \end{aligned}$$

since $\mathcal{STAB}(G) \subseteq \mathcal{TH}(G) \subseteq \mathcal{QSTAB}(G)$ [5, 11].

Lemma 2 *For any SAT01*(A, Γ), $\kappa(\Gamma, w) \leq m$.

Proof. Suppose $x \in \mathcal{QSTAB}(\Gamma)$. $\mathcal{QSTAB}$ is defined as

$$\mathcal{QSTAB}(G) = \{x \geq 0 \mid \sum_{v \in Q} x_v \leq 1 \text{ for all cliques } Q \text{ of } G\}.$$

This implies that $Ax \leq e$ holds since all inequalities of this system are constraints of $\mathcal{QSTAB}(\Gamma)$. Suming them up, we obtain $w^T x \leq m$ with respect to the definition of w. Hence $\kappa(\Gamma, w) \leq m$ as claimed. QED.

Theorem 1 *For any SAT01*(A, Γ), $\vartheta(\Gamma, w) \leq m$.

Proof. This follows immediately from Lemma 2 and the "sandwich" inequality (6). QED.

Therefore, we may say that the inequality $\vartheta(\Gamma, w) < m - \epsilon$ for some fixed $0 < \epsilon < 1$ designates an easily recognizable subclass of unsatisfiable SAT01 instances. In the other cases, SAT01 is equivalent to deciding whether $\alpha(\Gamma, w) = m$ provided $m - \epsilon \leq \kappa(\Gamma, w) \leq m$.

3 SAT01 propagation

SAT01 formalism is also convenient for a constraint propagation technique exploring contradictory relations among variables. Together with the unit propagation and variables reduction rules it allows for simplification of many SAT01 instances and augmentation of the contradiction graph by new edges. The latter may decrease $\vartheta(\Gamma, w) - \alpha(\Gamma, w) > 0$ gap and improve the chance that an unsatisfiable SAT01 instance is recognizable via $\vartheta(\Gamma, w) < m$ inequality.

The *SAT01 propagation* technique is based upon three rules.

1. *Unit propagation.* If an equation of (3) has only one variable, this variable must equal 1; if a variable is assigned 1, all variables contradicting to it must be assigned 0 and reduced; if all variables of an equation are eventually assigned 0, this equation is unsatisfiable and so the whole instance is unsatisfiable too.

2. *Clique analysis.* If there is a variable contradicting to all variables of an equation, it must be assigned 0 and reduced (otherwise the equation cannot be satisfied.) In other words, if some equation does not correspond to a maximal clique of $\Gamma(X, B)$, all variables corresponding to vertices fully connected with the equation clique must be reduced.

3. *Contradiction analysis.* If there are two variables x_j and x_k not contradictory á priori and an equation $\sum_\ell x_\ell = 1$ such that $\forall \ell \; ((j, \ell) \in B) \vee ((k, \ell) \in B)$, then x_j and x_k are contradictory, so a new edge (j, k) should be introduced in $\Gamma(X, B)$.

These three rules are applied iteratively until no more variables can be reduced and no new edges can be introduced. Since each of the rules can be checked in polynomial time and the total amount of operations over an instance cannot be more than the number of variables plus the number of non-edges of the contradiction graph, we conclude that SAT01 propagation can be performed in polynomial time. The exact complexity, however, significantly depends on implementation of the procedure.

4 Polynomial reduction to SAT01

From the theory of NP-completeness it is known that all NP-complete problems are polynomially equivalent. However, if we arbitrarily take two NP-complete problems, most probably it will not be easy to reduce them to each other. Moreover, if we find such a reduction, we will face a need to introduce a lot of artificial variables or "gadgets" within the new formulation. These gadgets tend to have no interpretation within the original problem. Besides, it gives a significant increase for the problem size. So, reductions within NP seemed having no practical use for solving methods.

However, all reductions to SAT01 we considered give natural new variables and such constraints that, in essence, are reformulation of original constraints in terms of contradictions. In general, we consider any problem to be a certain set of contradictions among demands that exist in the problem universe. From this viewpoint SAT01 is a very natural framework for NP problems. Here we show some illustrative examples.

4.1 CNF satisfiability problem (SAT)

CNF-SAT (or simply SAT) was historically the first problem proven to be NP-complete [1, 2]. It consists in determining existence of a satisfiable assignment for a boolean formula $f(x)$ given in conjuctive normal form. To reduce SAT to SAT01, we replace each clause $C = \ell_1 \vee \ell_2 \vee \ldots \vee \ell_k$ of $f(x)$ by an equation

$$\ell_1 + (\overline{\ell_1} \wedge \ell_2) + \ldots + (\overline{\ell_1} \wedge \overline{\ell_2} \wedge \ldots \wedge \ell_k) = 1. \tag{7}$$

We introduce all conjunctions of literals appearing in the obtained equations as new 0-1 variables. To determine á priori contradictions among them we look for pairs containing a common variable of the original SAT problem. If the variable is negated in one conjunction of the pair but not negated in the other, then the pair is contradictory á priori, so we connect them by an edge in the SAT01 contradiction graph. We note that if $f(x)$ has m clauses and L_i is the length of i-th clause, the obtained SAT01 instance has m equations and not more than $\sum_{i=1}^{m} L_i$ variables.

Claim 2 *The obtained SAT01 instance is satisfiable iff $f(x)$ is satisfiable.*

Proof. If x is a solution to the original SAT instance, all equations (7) are satisfied because exactly one of the left-hand side items equal 1. Namely, it is j-th item if ℓ_j if the first true literal of C.

Conversely, if we have a solution to the SAT01 instance, we select the conjunctions of literals that are equal to 1 and assign all of those literals to satisfy the conjunctions. Due to construction of the SAT01 contradiction graph, any case when both a literal ℓ and its negation $\overline{\ell}$ should be true is impossible. This literals assignment gives proper values to the subset of variables $\{x_j\}$ appeared in them. The remaining free variables may be assigned arbitrary. The obtained assignment satisfy $f(x)$ because each clause C has at least one true literal ℓ_j if the j-th left-hand side item of the corresponding equation (7) equals 1. QED.

4.2 Hamilton cycle problem (HCP)

Hamilton cycle problem is an NP-complete problem asking whether there is a simple cycle in a given graph $G(V, E)$, $|V| = n$, involving all its vertices [1]. Introduce a two-dimensional array of 0-1 variables $X = (x_{ij})_{n \times n}$ and construct a SAT01 instance over them. Let x_{ij} denote "i-th vertex of an HC is j-th vertex of G". The HC definition may be, in essence, formulated as follows:

- There is one and only one i-th vertex of the cycle.
- j-th vertex of G is met in the cycle exactly once.
- If j-th vertex of G is i-th vertex of the cycle and k-th vertex of G is $((i \bmod n) + 1)$-th vertex of the cycle, then $(j, k) \in E$.

The first two conditions mean that X must be a permutation matrix, that is

$$\begin{cases} \forall i = 1 \ldots n \ \sum_{j=1}^{n} x_{ij} = 1, \\ \forall j = 1 \ldots n \ \sum_{i=1}^{n} x_{ij} = 1. \end{cases} \tag{8}$$

The third condition imposes additional contradictions among the variables. Namely, if

$$(p = (i \bmod n) + 1) \wedge ((j, q) \notin E), \tag{9}$$

then x_{ij} and x_{pq} are contradictory á priori.

Claim 3 *G is hamiltonian iff the obtained SAT01 instance is satisfiable.*

Proof. Assume G is hamiltonian and $C = (v_1, v_2, \dots, v_n)$ is its hamilton cycle. Assign $x_{1v_1} = x_{2v_2} = \dots = x_{nv_n} = 1$ and all the other variables $x_{ij} = 0$. It is easy to see that this assignment obeys (8) and does not create contradictions by (9). So, if G has a hamilton cycle, the SAT01 instance is satisfiable.

Conversely, if $X = (x_{ij})$ is a solution to the SAT01 instance, it is a permutation matrix according to (8). So, each of its rows contains a 1-entry. Let them be $x_{1v_1} = x_{2v_2} = \dots = x_{nv_n} = 1$. Then, if no two of these variables are contradictory according to (9), $C = (v_1, v_2, \dots, v_n)$ is its hamilton cycle. QED.

The obtained SAT01 instance involves n^2 variables and $2n$ equations. We note that it defines HC up to a starting vertex shift. That is, though, for example, $(v_1, v_2, \dots, v_n)$ and $(v_2, \dots, v_n, v_1)$ are the same HC, the corresponding SAT01 solutions are different. Thus, we may additionally simplify the obtained SAT01 instance assigning one arbitrary x_{ij} to 1 from the beginning (say, $x_{11} = 1$.)

4.3 Graph isomorphism

Graph isomorphism is, apparently, the only natural NP problem, which is not known to be either polynomial or NP-complete. It considers two graphs $G_1(V_1, E_1)$ and $G_2(V_2, E_2)$ and asks for a bijection $f : V_1 \to V_2$ such that it also creates a bijection for edges: $(f(v_1), f(v_2)) \in E_2$ iff $(v_1, v_2) \in E_1$. Without loss of generality, let $|V_1| = |V_2| = \{1 \dots n\}$ (if $|V_1| \neq |V_2|$, a graph isomorphism is trivially impossible.) Similar to the HCP$\to$SAT01 reduction, we introduce a two-dimensional array of 0-1 variables $X = (x_{ij})_{n \times n}$ and construct a SAT01 instance over them. Let x_{ij} denote $f(i) = j$. Since f is a bijection, X must again be a permutation matrix and satisfy (8). The edge bijection condition implies that if

$$(((i,p) \in E_1) \wedge ((j,q) \notin E_2)) \vee (((i,p) \notin E_1) \wedge ((j,q) \in E_2)), \tag{10}$$

then x_{ij} and x_{pq} are contradictory á priori (i.e. if i maps to j and p maps to q, then (i,p) adjacency in G_1 must be the same as (j,q) adjacency in G_2.)

Claim 4 *G_1 and G_2 are isomorphic iff the obtained SAT01 instance is satisfiable.*

Proof. It follows directly from the fact that the graph isomorphism definition has been merely reformulated with the SAT01 notation. QED.

So, we used n^2 variables and $2n$ equations to reduce the graph isomorphism problem to SAT01.

4.4 Subgraph isomorphism problem

This one may be seen as a generalization of the graph isomorphism problem where it is asked whether one of the graphs is isomorphic to some vertex-induced subgraph of the other graph. This problem is NP-complete [1]. So, let $G_1(V_1, E_1)$, $V_1 = \{1 \dots m\}$ and $G_2(V_2, E_2)$, $V_2 = \{1 \dots n\}$ be two given graph and we look for an injection $f : V_1 \to V_2$ preserving vertex adjacency: $(f(v_1), f(v_2)) \in E_2$ iff $(v_1, v_2) \in E_1$. We again introduce a two-dimensional array of 0-1 variables $X = (x_{ij})_{m \times n}$, but impose only one group of equation constraints:

$$\forall i = 1 \dots m \ \sum_{j=1}^{n} x_{ij} = 1. \tag{11}$$

The condition (10) for additional are contradictory á priori pairs of variables remains the same.

Claim 5 *G_1 is isomorphic to some vertex-induced subgraph of G_2 iff the obtained SAT01 instance is satisfiable.*

Proof. Again, the claim is obviously true since (11) together with the variable contradiction condition (10) just reformulates the subgraph isomorphism definition in terms of 0-1 variables. QED.

This reduction gives mn variables and m equations in the obtained SAT01 instance.

4.5 Quasigroup completion problem (QCP)

Quasigroup completion problem (or latin square completion) consists in completion an $n \times n$ array with integers from $\{1 \dots n\}$ to a latin square, provided some of its cells are prefilled. This problem is NP-complete [6]. To reduce it to SAT01 we express the concept of latin square via a three-dimensional array of 0-1 values. Namely, let a 0-1 variable $x_{ijk},\ i, j, k \in \{1 \dots n\}$ denote "Cell (i, j) is filled with number k". The array of these variables determines a latin square if and only if

$$\begin{cases} \forall i, j \ \sum_k x_{ijk} = 1, \\ \forall i, k \ \sum_j x_{ijk} = 1, \\ \forall j, k \ \sum_i x_{ijk} = 1. \end{cases} \tag{12}$$

Let the QCP input be a matrix $L = (\ell_{ij})_{n \times n}$ such that $\ell_{ij} = k \in \{1 \dots n\}$ if the cell (i, j) is prefilled with k, and $\ell_{ij} = 0$ otherwise. Correspondingly, the number of holes h is the total number of entries (i, j) such that $\ell_{ij} = 0$. Without loss of generality we assume that this input does not immediately violate the latin square constraints. That is, $\ell_{ij} = \ell_{iq} > 0,\ j \neq q$ or $\ell_{ij} = \ell_{pj} > 0,\ i \neq p$ cases never occur. Otherwise, the QCP instance is trivially unsatisfiable. We consider only those variables $X = (x_{ijk})$ for which

$$(\ell_{ij} = 0)\&(\forall p: \ \ell_{pj} \neq k)\&(\forall q: \ \ell_{iq} \neq k) \tag{13}$$

holds. That is, we select the variables not chosen in and not contradicting á priori to the given prefilling.

Claim 6 *The system (12) taken over variables (x_{ijk}) whose indices obey (13) is satisfiable iff the QCP instance given by L is satisfiable.*

Proof. Validity of this claim directly follows from the meaning of variables (x_{ijk}) and the condition (13) on chosen (i, j, k) triples. QED.

So, QCP for an $n \times n$ array is reduced to a SAT01 instance having not more than n^3 variables.

4.6 Extended 15-puzzle

The extended 15-puzzle is the generalization of the well-known 15-puzzle for $m \times n$ board. It is NP-hard [7]. So if we fix the number of turns T, we obtain an NP-complete problem. To reduce it to $SAT01$ we introduce 0-1 variables $X = (x_{tkij})$, $t = 0 \dots T$, $k = 1 \dots mn - 1$, $i = 1 \dots m$, $j = 1 \dots n$. x_{tkij} means that on turn t number k is in position (i, j). The system of equations is

$$\forall t, k \ \sum_i \sum_j x_{tkij} = 1 \tag{14}$$

(any time any number takes one and only one position). The variable conflicts not induced by equations are

- x_{tkij} contradicts x_{tlij} for $k \neq l$ (not more than one number may be in one position)
- x_{tkij} contradicts $x_{t+1\,kpq}$ if $|i-p|+|j-q|>1$ (a number cannot move off farther than to one of four nearest positions)
- x_{tkij} contradicts $x_{t+1\,lij}$ for $k \neq l$ (at least two turns are needed to replace a number by another)

After all, we determine the initial configuration assigning x_{0kij} appropriately and the goal configuration assigning $x_{Tkij} = 1$ for $k = (i-1)n + j$.

Claim 7 *The obtained SAT01 instance is satisfiable iff $m \times n$ extension of the 15-puzzle is solvable within T turns.*

Proof. The claim follows directly from the construction of the considered SAT01 instance. QED.

So, we reduced $m \times n$ extension of the 15-puzzle to a SAT01 instance of $mn(mn-1)(T+1)$ variables and $(mn-1)(T+1)$ equations assuming not more than T steps are required to solve the puzzle.

4.7 Hanoi tower puzzle

Let a 0-1 variable x_{tkij} means "on turn t block k is moved from pile i to pile j." At that, $t = 1 \ldots T$, $k = 1 \ldots n$, $i = 1 \ldots 3$, $j = 1 \ldots 3$, where T is the fixed number of turns to solve the puzzle and n is the number of blocks. Assuming $x_{tkii} = 1$ for block k remaining in pile i on turn t, we obtain the following system of equations

$$\begin{array}{ll} \forall t & \sum_k \sum_i \sum_{j \neq i} x_{tkij} = 1 \\ \forall t\ \forall k & \sum_i \sum_j x_{tkij} = 1 \end{array} \tag{15}$$

The first means that for each turn there is one and only one movement. The second means that each block takes one and only one place. Additional á priori contradictions are

- x_{tkij} contradicts $x_{t+1\,kpq}$ whenever $j \neq p$ (a block's movements must be a chain);
- x_{tkij} contradicts x_{tljj} when $i \neq j$ and $l > k$ (a block cannot be put down on a pile if there is a smaller block);
- x_{tkij} contradicts x_{tlii} when $i \neq j$ and $l < k$ (a block cannot be moved if there is a smaller block in the same pile that anyway must be above).

We also should define the initial and the goal configuration assigning some variables á priori. For the classic conditions of the puzzle, when we need to transfer all blocks from the first pile to the third, $x_{1211} = x_{1311} = \ldots = x_{1n11} = 1$, all x_{11ij} except x_{1112} and x_{1113} are zeroes; all x_{T1ij} except x_{T113} and x_{T123} are zeroes, $x_{T233} = x_{T333} = \ldots = x_{Tn33} = 1$.

Claim 8 *The obtained SAT01 instance is satisfiable iff the Hanoi tower puzzle is solvable within T turns.*

So, we reduced the Hanoi tower puzzle for n blocks and T turns allowed to a SAT01 instance of $9Tn$ variables and $T(n+1)$ equations.

4.8 Blocksworld problem

This well-known AI planning problem has been considered in a number of papers, e.g. [9, 13, 14], concerning SAT-based approach to AI planning. We present here an alternative SAT01-based approach to the problem.

Thus, let a 0-1 variable x_{tij} means "on turn t block i is on j". At that $t \in \{0 \ldots T\}$, $i \in \{1 \ldots n\}$, $j \in \{1 \ldots n\} \cup \{\text{table}\}$, $i \neq j$, where T is the last turn for obtaining the goal configuration and n is the number of blocks. On each odd turn we handle some clear block to an arm and on each even turn we put the handled block down on another clear block or on the table. So, introducing variables 0-1 variables a_{ti} for all odd t (which mean "on turn t block i is in the arm") we may obtain the following equations:

$$\begin{array}{lll} \forall \text{ even } t \quad \forall i & & \sum_j x_{tij} = 1 \\ \forall \text{ odd } t \quad \forall i & a_{ti} + & \sum_j x_{tij} = 1 \\ \forall \text{ odd } t & & \sum_i a_{ti} = 1. \end{array} \tag{16}$$

The first and the second mean that on each turn each block takes one and only one position, the third means that one and only one block is in the arm on each odd turn. Now as usual we complete the contradiction graph by not equation-induced variable conflicts.

- x_{tij} contradicts x_{tkj} whenever $i \neq k$ and j is not the table (not more than one block may be on any other block);
- x_{tij} contradicts $x_{t+1\,il}$ whenever $j \neq l$ (at least two turns are needed to move a block to another position (the first one is for handling and the second one is for putting down));
- x_{tij} contradicts $x_{t+1\,kj}$, $x_{t+2\,kj}$, $x_{t+3\,kj}$ whenever $i \neq k$ and j is not the table; if t is odd, x_{tij} contradicts $x_{t+4\,kj}$ as well (four turns on even t and five turns on odd t are needed to change the block above through handling)
- a_{tj} contradicts $x_{\tau ij}$ whenever $|t - \tau| \leq 2$ (the temporary distance between the handling of a block and a moment when there is another block on it is at least three turns).

So, we reduced a blockworld instance for n blocks and $T + 1$ turns to a SAT01 instance of $n^2(T + 1) + n(T/2 - 1)$ variables and $nT + n(T/2 - 1)$ equations.

4.9 Bitwise arithmetics

Coding bitwise arithmetic is crucial, for example, for reduction of the factoring problem. After a SAT01 equivalent of bitwise addition/subtraction is obtained, Factoring→SAT01 reduction is pretty straightforward, but still somewhat ponderous to describe explicitly in a paper not specifically dedicated to this problem. So, we confine the present description to bitwise addition/subtraction reduction to SAT01.

Thus, consider bitwise addition of two numbers: $a + b = c$. Let all the numbers have n bits in the binary form. So, we have 0-1 variables $a_0, \ldots, a_{n-1}, b_0, \ldots, b_{n-1}, c_0, \ldots, c_{n-1}$. For zeroth bits we have

$$c_0 = (a_0 \wedge \overline{b_0}) \vee (\overline{a_0} \wedge b_0).$$

Since

$$(\overline{a_0} \wedge \overline{b_0}) + (a_0 \wedge \overline{b_0}) + (\overline{a_0} \wedge b_0) + (a_0 \wedge b_0) = 1,$$

we can form the following equation for c_0:

$$c_0 + (\overline{a_0} \wedge \overline{b_0}) + (a_0 \wedge b_0) = 1. \tag{17}$$

So, $(\overline{a_0} \wedge \overline{b_0})$ and $(a_0 \wedge b_0)$ are introduced as new 0-1 variables.

Next, we introduce new 0-1 variables $d_0, \ldots, d_{n-2}$ for carries. d_0 must obey the equation

$$d_0 + (\overline{a_0} \wedge \overline{b_0}) + (a_0 \wedge \overline{b_0}) + (\overline{a_0} \wedge b_0) = 1. \tag{18}$$

So, $(a_0 \wedge \overline{b_0})$ and $(\overline{a_0} \wedge b_0)$ are also new 0-1 variables.

Equations for middle bits are more complex since there is a carry from the previous bit. We must introduce new 0-1 variables $(\overline{a_j} \wedge \overline{b_j} \wedge \overline{d_{j-1}}), \ldots, (\overline{a_j} \wedge b_j \wedge d_{j-1})$, and using them we impose the following constraints on c_j and d_j:

$$\begin{array}{l} c_j + (\overline{a_j} \wedge \overline{b_j} \wedge \overline{d_{j-1}}) + (a_j \wedge b_j \wedge \overline{d_{j-1}}) + (a_j \wedge \overline{b_j} \wedge d_{j-1}) + (\overline{a_j} \wedge b_j \wedge d_{j-1}) = 1 \\ d_j + (\overline{a_j} \wedge \overline{b_j} \wedge \overline{d_{j-1}}) + (a_j \wedge \overline{b_j} \wedge \overline{d_{j-1}}) + (\overline{a_j} \wedge b_j \wedge \overline{d_{j-1}}) + (\overline{a_j} \wedge \overline{b_j} \wedge d_{j-1}) = 1 \end{array} \tag{19}$$

For the last bit there is no carry and thus, there are only four new 0-1 conjunctive variables. However, there are two equations again (the second one prohibits a carry):

$$\begin{array}{c} c_{n-1} + (\overline{a_{n-1}} \wedge \overline{b_{n-1}} \wedge \overline{d_{n-2}}) = 1 \\ (\overline{a_{n-1}} \wedge \overline{b_{n-1}} \wedge \overline{d_{n-2}}) + (a_{n-1} \wedge \overline{b_{n-1}} \wedge \overline{d_{n-2}}) + (\overline{a_{n-1}} \wedge b_{n-1} \wedge \overline{d_{n-2}}) + (\overline{a_{n-1}} \wedge \overline{b_{n-1}} \wedge d_{n-2}) = 1. \end{array} \tag{20}$$

So, (17,18,19,20) define the necessary SAT01 system of equations. Finally, we must reveal all contradictory á priori pairs of the introduced variables as in the conversion from SAT: those having a common conjunct but with opposite negation signs cannot equal 1 simultaneously.

5 Informational foundation of SAT01 formalism

First of all, what is a problem? A problem (in the broad sense, including non-mathematical) is an equation $P(x) = true$ where x is a looked-for object of some universe and P is a given predicate defined over the universe. Indeed, whenever we face a problem, we are looking for something called a “solution” that may be a value of a variable, a proof of a theorem, an action in our life, etc. Under an appropriate formalization we can rigorously define the universe of objects having the form of a solution and – when the problem is mathematical – the objective predicate P too. Now we assume that P can be represented as a conjunction of simple demands over x (clauses) that easily can be satisfied separately. It is completely true for NP problems. At that the main hardness is to satisfy all clauses at once. Here are some examples.

- *SAT.* Each clause is an OR-clause. It can be satisfied in $2^k - 1$ ways where k is its length.
- *A system of linear equations and inequalities.* Each equation/inequality is a clause. An appropriate assignment satisfying a single constraint is obvious (e.g., assign all but one contained variable to zero and solve $ax = b$ equation for the last free variable.)
- *HCP.* x is a sequence of graph vertices. The first group of clauses asserts that any two successive vertices (including the last to the first) must be joined by an edge. And the final clause is the assertion: “the sequence x is a permutation of all graph vertices”.

Now, when a problem is easy and when it is hard? The first easy case is when contradictions between clauses are not significant. That is, when we decide how to satisfy one clause, we do not make any obstacles to satisfy others. This case is not interesting because the problem should have a lot of solutions to be such one, whereas actual hard problems have a few only or even none solutions. So, the second case if each particular clause gives lots information about feasible values of contained in it variables. Imagine that in SAT we have not $\vee$- but $\wedge$-clauses i.e., as soon as a variable is included in a clause, its value is defined. Then to solve the problem we need just verify is there a contradictory pair of literals. A less extreme example is when each clause strongly determines one variable. In this case the problem should probably be easy too, as many variables can be reduced just in input.

Thus, we may say that in a suitable formulation for NP problems every clause must be as informative as possible. Now, calculate information given by one $\vee$-clause of length k. There are 2^k possibilities to assign k boolean variables, the clause rejects one only, so

$$\mathcal{I}_{cnf} = -\log_2 \frac{2^k - 1}{2^k} = k - \log_2(2^k - 1). \tag{21}$$

This value vanishes very quickly when $k \to \infty$. From the algorithmic viewpoint it means that for arbitrary k-SAT it is almost useless to process particular subsets of clauses – if such a subset is not very large, you can take almost none information about solution from it; but if the subset is large, it should be very difficult to grasp all its clauses simultaneously!

Hence we may conclude: *$\vee$-clauses are the worst as they are almost uninformative*! And perhaps, *SAT is the worst choice to overcome NP-hardness!*

Now we do similar informational computations for SAT01. First, for a SAT1 equation k of 2^k possibilities remain, so

$$\mathcal{I}_{sat1} = -\log_2 \frac{k}{2^k} = k - \log_2 k. \tag{22}$$

It is easy to see that this value tends to infinity for $k \to \infty$. Moreover, as it is impossible to allow less than k assignments for a clause of length k in general case (otherwise some variables can be immediately reduced), SAT1 equations are most informative constraints over 0-1 variables!

An additional binary contradiction has information

$$\mathcal{I}_{sat0} = -\log_2 \frac{3}{4} \approx 0.415 \text{ bits}. \tag{23}$$

6 Conclusions

We have presented a promising formalism providing an opportunity to consider NP problems from a unified viewpoint and decide their hardness on the basis of the contradiction graph's Lovász number value in particular instances. We also have substantiated this choice by an information theory argument. Hopefully, the SAT01 analysis will find a broad application among researchers of particular NP problems as an alternative approach to investigate properties of those problems. Apart from this, SAT01 research should stimulate development of faster methods for computation of the Lovász number and its improvements [8, 15]. In conjunction with the presented results such methods should give numerous new opportunities for solving various NP problems efficiently in practice.

References

[1] M. Garey and D. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (Freeman & Co., 1979).

[2] S.A. Cook, The complexity of theorem-proving procedures, In: *Conference Record of 3rd Annual ACM Symposium on Theory of Computing*, Shaker Heights, Ohio (1971) 151–158.

[3] E. Balas and M. Padberg, Set partitioning: a survey, *SIAM Review*, **18** (1976) 710–760.

[4] T. J. Schaefer, The complexity of satisfiability problems, *Conference Record of the 10th Annual ACM Symposium on Theory of Computing*, 1–3 May, San Diego, California (1978) 216–226.

[5] L. Lovász, On the Shannon capacity of a graph, *IEEE Trans. Inform. Theory* **25**:1 (1979) 1–7.

[6] C. Colbourn, The complexity of completing partial latin squares, *Discrete Applied Mathematics*, **8** (1984) 25–30.

[7] D. Ratner and M. Warmuth, Finding a shortest solution for the $(N \times N)$-extension of the 15-puzzle is intractable, *Journal of Symbolic Computation*, **10** (1990) 111–137.

[8] L. Lovász and A. Schrijver, Cones of matrices and set-functions and 0-1 optimization, *SIAM J. Optim.* **1**:2 (1991) 166–190.

[9] H. Kautz and B. Selman, Planning as satisfiability, In *Proc. ECAI-92*, (1992) 359–363.

[10] K. Hoffman and M. Padberg, Solving airline crew scheduling problems by branch-and-cut, *Management Science*, **39** (1993) 657–682.

[11] D.E. Knuth, The Sandwich Theorem, *Elec. J. Comb.* **1** (1994).

[12] D. Levine, A parallel genetic algorithm for the set partitioning problem, *Technical Report MCS-P458-0894*, Argonne National Laboratory (1994).

[13] H. Kautz and B. Selman, Pushing the envelope: planning, propositional logic, and stochastic search, In: *Proc. AAAI-96* (1996).

[14] H. Kautz and B. Selman, The role of domain-specific knowledge in the planning as satisfiability framework, In: *Proc. AIPS-98* (1998).

[15] E. de Klerk and D. Pasechnik, Approximation of the stability number of a graph via copositive programming, *SIAM J. Optim.* **12**:4 (2001) 875–892.