

# On a random walk strategy for the Q2SAT problem

K. Subramani \*  
 LCSEE,  
 West Virginia University,  
 Morgantown, WV  
 {ksmani@csee.wvu.edu}

## Abstract

In this paper, we discuss a simple, randomized algorithm for the problem of checking whether a Q2CNF formula has a model, i.e., the Q2SAT problem. The algorithm is based on the coin-flipping strategy outlined in [Pap91] for the 2SAT problem. We show that a similar strategy works for the Q2SAT problem, in that the expected number of coin flips to discover a model for a satisfiable Q2CNF formula on  $n$  variables is  $n^2$  and the probability of not finding a model within  $2 \cdot n^2$  coin flips is less than one half.

## 1 Introduction

This paper is concerned with randomized strategies for the Q2SAT problem, i.e., whether a Q2CNF formula has a model. We design a simple, randomized algorithm for the same. Our algorithm is based on the coin-flipping strategy outlined in [Pap91], for the problem of checking whether a 2CNF formula has a satisfying assignment. In particular, our algorithm shows that the expected number of coin flips to discover a model for a satisfiable Q2CNF formula is  $n^2$  and that the probability that a model will not be discovered in  $2 \cdot n^2$  steps is at most one half.

The rest of this paper is organized as follows: Section §2 provides a formal description of the Q2SAT problem. The motivation for our work as well as related approaches in the literature are discussed in Section §3. Section §4 describes the random walk strategy for 2CNF formulae with a detailed analysis. In Section §5, we show that the analysis in Section §4 can be used to analyze the Q2SAT problem as well. We conclude in Section §6, by summarizing our work in this paper and pointing out avenues for future research.

## 2 Statement of Problem

Let  $\phi(x, y)$  be a 2CNF formula on the boolean variables  $S = \{x_1, x_2, \dots, x_n, y_1, y_2, \dots, y_n\}$ . Let  $Q(x, y)$  denote an arbitrary quantifier string, i.e., it associates a quantifier with each variable in  $S$  and imposes a linear ordering on them. Without loss of generality, we assume that  $x$  variables are existentially quantified and the  $y$  variables are universally quantified. The Q2SAT problem is concerned with checking the satisfiability of the specification  $\Phi = Q(x, y) \phi(x, y)$ .

The Q2SAT (or QBF) problem is better understood through the following 2 person game. Let  $\mathbf{X}$  and  $\mathbf{Y}$  be two players called the Existential player and the Universal player respectively. The game is played with  $\mathbf{X}$  and  $\mathbf{Y}$  making a sequence of moves, where a move by  $\mathbf{X}$  is an assignment of **{true, false}** for one of the  $x$  variables, while a move by  $\mathbf{Y}$  is an assignment of **{true, false}** for one of the  $y$  variables. The game is played in the sequential order specified by  $Q(x, y)$ ; for instance if  $Q(x, y) = \exists x_1 \forall y_1 \forall y_2$ , then  $\mathbf{X}$  decides  $x_1$ , after which  $\mathbf{Y}$  decides  $y_1$  and  $y_2$  and so on. Finally, the moves made by  $\mathbf{X}$  can be based on the moves already made by  $\mathbf{Y}$ ; likewise the moves made by  $\mathbf{Y}$  can be based on the moves already made by  $\mathbf{X}$ . The game is over, when both players have made all their moves. If the values that have been guessed result in  $\phi(x, y)$  being satisfied, then the game is said to have

---

\*This research was conducted in part at Aalborg Universitet, where the author was supported by a CISS Faculty Fellowship.

been won by  $\mathbf{X}$  and the specification  $\Phi$  is said to be satisfiable; likewise the strategy employed by  $\mathbf{X}$  is said to be a winning strategy. Otherwise the game is said to have been won by  $\mathbf{Y}$  and  $\Phi$  is said to be unsatisfiable. Note that a winning strategy for  $\mathbf{X}$  is also called a model for  $\Phi$ .

It is clear that a solution to a specification such as  $\Phi = Q(x, y) \phi(x, y)$  is in general, not a  $\{\mathbf{true}, \mathbf{false}\}$  assignment to the existentially quantified variables, but a strategy for the existential player  $\mathbf{X}$  to win the game, against all possible strategies adopted by the universal player  $\mathbf{Y}$ .

### 3 Motivation and Related Work

The problem of testing the satisfiability of CNF formulas (or SAT) is one of the more ubiquitous problems within the fields of Computer Science and Operations Research. Applications of this problem abound from areas as diverse as Econometrics and Planning to Graph Theory and Combinatorial Optimization [sat]. From the perspective of computational complexity, SAT was immortalized in [Coo] as the first natural NP-complete problem. Quantified Boolean Formulas (QBFs) are the natural extension of CNF formulas to full first order logic and have tremendous applications in Artificial Intelligence planning. It is well known that there is a restriction of QBF that is complete for each level of the polynomial hierarchy [GJ91].

In [Sch78] the complexities of various satisfiability problems were identified. It was shown that 2SAT and Q2SAT were polynomial time solvable; however no algorithmic procedure was discussed. [APT79] gave the first algorithms for these two problems; in fact, their algorithms run in linear time.

Satisfiability problems involving 3 or more literals per clause are NP-complete in general and it is unlikely that there exist efficient randomized algorithms for the same. Typical randomized approaches to satisfiability testing in such cases include involving taking a step at random in the simulated annealing method to find a global minimum or making a random decision on which branch of the search tree to explore. These approaches are not easily amenable to theoretical analysis although empirically, they have achieved reasonable success. In [Pap91], the first randomized algorithm for the satisfiability problem, which had provable polynomial time convergence and bounded error was presented. Their strategy is easily modeled as a one dimensional random walk. Applications of the random walk strategy to harder versions of Satisfiability with detailed implementation profiles are described in [WS02].

Our work is concerned with extending the coin-flipping strategy to Q2CNF formulas. To the best of our knowledge, such strategies have not been studied for QBFs.

### 4 A Randomized Algorithm for 2SAT

In [Pap91], a polynomial time Monte Carlo algorithm was designed for the 2SAT problem, i.e., for the the problem of checking whether a given 2CNF formula has a satisfying assignment. Their strategy is described by Algorithm (4.1).

As described above, Step (12 :) of Algorithm (4.1) will never be reached. In a practical implementation, a counter is set, so that the **while** loop is exited after a certain number of steps.

[Pap91] shows that if  $\phi$  is satisfiable, then Algorithm (4.1) finds a satisfying assignment with probability greater than one half. Their analysis is based on the following simple, but useful observations:

1. If the current truth value assignment  $T$ , to the variables  $\{x_1, x_2, \dots, x_n\}$  of the 2CNF formula  $\phi$ , does not satisfy  $\phi$ , then exists a clause  $C \in \phi$ , such that both its literals are set to **false** under  $T$ . Let us say that  $\phi$  is satisfiable and there exists a unique assignment  $\hat{T}$  that satisfies  $\phi$ . Let us also say that the current assignment differs from  $\hat{T}$ , in exactly  $k$  variables. In any truth assignment that satisfies  $\phi$ , at least one of the 2 literals in  $C$  must be set to **true**; thus in  $\hat{T}$  at least one of the literals of  $C$  is set to **true**. It follows that if  $T$  is altered by picking one of the 2 literals in  $C$  (uniformly and at random) and flipping its value, with probability at least  $\frac{1}{2}$ , the resulting truth assignment  $T'$  is closer to  $\hat{T}$  than  $T$  is. In fact, with probability  $\frac{1}{2}$ ,  $T'$  is one variable closer (than  $T$ ) to  $\hat{T}$  and with probability  $\frac{1}{2}$ ,  $T'$  is one variable farther away from a satisfying assignment than  $T$  is.

**Function** 2SAT-SOLVE( $\phi$ )

```

1: Let  $T$  be the initial truth assignment to the variables  $\{x_1, x_2, \dots, x_n\}$  of  $\phi$ .
2: if ( $T$  is a satisfying assignment) then
3:   return(" $\phi$  is satisfiable")
4: end if
5: while ( $T$  is not a satisfying assignment) do
6:   Pick a clause  $C$  that is falsified in  $\phi$  by  $T$ .
7:   Choose a literal at random from the two literals in  $C$  and flip its value in  $T$ , so that  $C$  is satisfied.
8:   if ( $T$  now satisfies  $\phi$ ) then
9:     return(" $\phi$  is satisfiable")
10:  end if
11: end while
12: return(" $\phi$  is probably unsatisfiable")

```

**Algorithm 4.1:** Randomized algorithm for the 2SAT problem

2. Consequently, the strategy of choosing an unsatisfied clause and flipping one of its literals at random, can be modeled as a one-dimensional random walk with a reflecting and absorbing barrier. The random walk is over the integers in the range  $[0..n]$ , with 0 as the absorbing barrier (indicating that the current assignment  $T$  disagrees with  $\hat{T}$  on 0 variables and  $n$  serving as the reflecting barrier (indicating that  $T$  disagrees with  $\hat{T}$  on all  $n$  variables).

We now provide a detailed convergence analysis of Algorithm (4.1); our description is modeled on the analysis in [Ros00].

**Lemma 4.1** *Let  $X$  and  $Y$  denote two random variables; let  $E[X|Y]$  denote that function of the random variable  $Y$ , whose value at  $Y = y$  is  $E[X|Y = y]$ . Then,*

$$E[X] = E[E[X|Y]].$$

*In other words,*

$$E[X] = \sum_y E[X|Y = y] \cdot Pr[Y = y].$$

**Proof:** See pages 101 – 103 of [Ros00].  $\square$

Essentially, Lemma (4.1) allows us to calculate the expectation of a random variable  $X$ , by taking the weighted average of the conditional expectation of  $X$ , given that  $Y = y$ , with the conditional expectation being weighted with the probability that  $Y = y$ .

Assume that  $\phi$  has a satisfying assignment  $\hat{T}$ . Let  $t(i)$  denote the expected number of flips from the current assignment to get to  $\hat{T}$ , assuming that the current assignment is exactly  $i$  values away from  $\hat{T}$ .

As argued above and using Lemma (4.1), it is not hard to see that:

$$\begin{aligned}
t(0) &= 0 \\
t(i) &= \frac{1}{2} \cdot [(t(i-1) + 1) + (t(i+1) + 1)] \\
t(n) &= 1 + t(n-1)
\end{aligned} \tag{1}$$

Note that if we start off with  $T = \hat{T}$ , i.e.,  $T$  agrees with  $\hat{T}$ , on all  $n$  variables, then we do not need any flips. Likewise, if the initial assignment  $T$ , differs from  $\hat{T}$  on all  $n$  variables, then flipping any literal must increase the number of variables on which  $T$  and  $\hat{T}$  agree. Finally, for  $i \neq 0, 1$ , note that with probability one half, the new assignment will differ from  $\hat{T}$ , in one more variable than  $T$ ; likewise, with the same probability, it will differ from  $\hat{T}$ , in one less variable than  $T$ ; applying Lemma (4.1), we get the desired weighted average.

In [Pap94], it is shown that the System (1) can be solved to yield  $t(n) = n^2$ , i.e., the expected number of literal flips is  $n^2$ .

From classical probability, we know that

**Theorem 4.1** Let  $X$  be a random variable that assumes non-negative values only. For any  $a > 0$ ,

$$\Pr[X \geq a \cdot E[X]] \leq \frac{1}{a} \quad (2)$$

Theorem (4.1) is known as Markov's inequality ([Ros00]) and it can be applied to our flipping algorithm to conclude that if there is a satisfying assignment, the probability that this assignment is not discovered after  $2 \cdot n^2$  literal flips times is less than  $\frac{1}{2}$ .

## 5 A coin-flipping strategy for Q2SAT

There are 2 preconditions that need to be met, for the analysis of Section §4 to hold for a randomized algorithm for satisfiability testing:

1. **Y<sub>1</sub>** : For a literal to be flipped, it must have exactly 2 assignable values, at the time at which it is assigned.
2. **Y<sub>2</sub>** : Given that the current assignment (model) differs from the satisfying assignment (model) in exactly  $i$  variables, flipping a literal should result in exactly one of two events, viz.,
  - (a) The resulting assignment differs from the satisfying assignment in  $(i - 1)$  variables,
  - (b) The resulting assignment differs from the satisfying assignment in  $(i + 1)$  variables,

Event (2a) should have probability *at least* one half, while event (2b) should have probability *at most* one half.

Note that **Y<sub>1</sub>** is trivially met by simple CNF formulas, since every variable must be either **true** or **false**, in any satisfying assignment. In case of QBFs though, a solution is a vector of Skolemized boolean functions and hence **Y<sub>1</sub>** does not hold, in general. Condition **Y<sub>2</sub>** cannot be met if the clause that is not satisfied by the current assignment has more than 2 literals; indeed, both [Pap94] and [Ros00] provide an instance of a HornCNF formula that takes exponential time to converge, under randomized literal flipping.

We now show that the above 2 conditions can be met in case of Q2CNF decidability. Let  $\Phi = Q(x, y) \phi(x, y)$  denote a Q2CNF formula, with the existentially quantified variables being drawn from the set  $V_1 = \{x_1, x_2, \dots, x_n\}$  and the universally quantified variables being drawn from the set  $V_2 = \{y_1, y_2, \dots, y_n\}$ .  $Q(x, y)$  imposes a linear ordering on the set  $V_1 \cup V_2$ ; each clause  $C \in \phi(x, y)$  has exactly 2 literals.

We first establish that **Y<sub>1</sub>** is met. Observe that

1. Satisfiable Q2CNF formulas have simple models, i.e., models in which every existentially quantified variable  $x_j$ , is assigned **true**, **false**,  $y_i$  or  $\bar{y}_i$ , where  $y_i$  is a universally quantified variable that preceded  $x_j$  in  $Q(x, y)$ ; indeed the linear time algorithm in [APT79], which checks whether a Q2CNF formula  $\Phi$  has a model, produces a simple model, if the Q2SAT instance is satisfiable. Further, note that that verifying a simple model for a Q2CNF formula is a linear time procedure.
2. Now process the clauses in  $\phi$  sequentially. Let  $C$  be an arbitrary clause in  $\phi(x, y)$ . Note that
  - (a) If  $C$  is of the form  $(y_i, y_j)$ , i.e., both its literals universally quantified, then  $\Phi$  does not have a model.
  - (b) If  $C$  is of the form  $(x_j, y_k)$ , and  $x_j$  occurs after  $y_k$  in  $Q(x, y)$ , then  $x_j = 1$  or  $\bar{y}_k$  in any model that satisfies  $\Phi$ . Proceed to the next clause.

However, if  $y_k$  succeeds  $x_j$  in  $Q(x, y)$ , or if  $x_j$  is also paired with  $y_p$ ,  $p \neq k$ , in a clause distinct from  $C$ , then  $x_j$  *must* be **true** in any model for  $\Phi$ . (If the existential literal is  $\bar{x}_j$ , instead of  $x_j$ , replace **true** with **false**.) Delete  $x_j$  from  $Q(x, y)$  and all clauses containing  $x_j$ . Likewise, replace all clauses of the form  $(\bar{x}_j, u)$ , with the unit clause  $(u)$ . If  $u$  is universally quantified, then  $\Phi$  does not have a model; if it is existentially quantified, it must be set to **true** in any model and the process of altering clauses and  $Q(x, y)$  is repeated.

- (c) If  $C$  is of the form  $(x_i, x_j)$ , the following cases arise:

- i. Both  $x_i$  and  $x_j$  have been assigned in Step (2b); in this case, there is nothing to be done;
- ii.  $x_i$  is not assigned, but  $x_j$  is assigned to **true**. Delete this clause from  $\phi(x, y)$ , since it has no bearing on the existence of a model.
- iii.  $x_i$  is not assigned, but  $x_j$  is assigned to **false**. Set  $x_i$  to **true**; delete  $x_i$  from  $Q(x, y)$  and reprocess all clauses, as in the latter half of Step (2b).
- iv. Neither  $x_i$  nor  $x_j$  has been assigned. Proceed to the next clause.

When Step (2 :) terminates, we will be left with a collection of clauses (say  $\phi_{eh}$ ) of the form  $(x_i, x_k)$ , i.e., both literals will be existential.  $\phi_{eh}$  is called the existential hull of  $\Phi$ ; in any model for  $\Phi$ , the variables in  $\phi_{eh}$  are assigned either **true** or **false**! This assertion follows immediately from the proof of correctness for Q2CNF decidability in [APT79]. We have thus established that condition **Y<sub>1</sub>** is met by Q2CNF formulas.

We proceed to show that condition **Y<sub>2</sub>** is also met. Assume that  $\Phi$  has a model  $\hat{T}$ . Let the current assignment  $T$ , differ from  $\hat{T}$ , in exactly  $i$  variables and let  $C$  be a clause, that is not satisfied by  $T$ . First observe that under our assignment scheme,  $C$  cannot be of the form  $(x_i, y_j)$ ; hence  $C$  must be of the form  $(x_i, x_j)$ . Since  $C$  is falsified, at least one of the 2 variables is set to a different value in  $\hat{T}$ . Further, there are only 2 possible values for each of these variables, as per the above discussion. It therefore follows that picking one of the variables at random and flipping it, moves  $T$  closer to  $\hat{T}$  with probability at least one half.

Algorithm (5.1) is a formal description of our strategy:

**Function** Q2SAT-SOLVE( $\Phi = Q(x, y) \phi(x, y)$ )

- 1: Process each clause in  $\phi(x, y)$  till every existentially quantified variable is set to one of: (a)**true**, (b)**false**, (c)**true**| $\bar{y}_j$ , (d) **true**|**false**.
- 2: {There are a few more symmetric categories, but we ignore them to simplify the exposition. The crucial observation is that each variable has at most 2 valid values.}
- 3: Let  $T$  be the initial truth assignment to the variables  $\{x_1, x_2, \dots, x_n\}$ .
- 4: **if** ( $T$  is a model for  $\Phi$ ) **then**
- 5:     **return**(" $\Phi$  is satisfiable")
- 6: **end if**
- 7: **while** ( $T$  is not a model) **do**
- 8:     Pick a clause  $C$  that is falsified in  $\phi$  by  $T$ .
- 9:     Choose a literal at random and flip its value in  $T$ , so that  $C$  is satisfied.
- 10:    **if** ( $T$  is now a model for  $\phi$ ) **then**
- 11:     **return**(" $\Phi$  is satisfiable")
- 12:    **end if**
- 13: **end while**
- 14: **return**(" $\Phi$  is probably unsatisfiable")

**Algorithm 5.1:** Randomized algorithm for the Q2SAT problem

Using an analysis, that is almost identical to the one in Section §4, we can conclude that

***Theorem 5.1** Algorithm (5.1) is a randomized algorithm for the Q2SAT problem. Given a satisfiable Q2CNF formula, the probability that it finds a model in at most  $2 \cdot n^2$  literal flips is at least one half.*

## 6 Conclusion

In this paper, we proposed and analyzed a randomized algorithm for the Quantified Satisfiability problem, restricted to 2CNF formulas. Our strategy is similar to the technique developed in [Pap91], in that we use a coin-flip to decide how to proceed at each stage, however our analysis is significantly different. While randomized strategies have been used as heuristics towards satisfiability testing in QBFs, to the best of our knowledge, our work represents the first attempt to analytically bound the running time and quality of the output, albeit for a restricted class of QCNF formulas.

A number of interesting questions arise, as a consequence of our work in this paper; some of the problems on which we are currently working are enumerated below:

1. Can the random walk technique be extended to larger classes of CNF formulas? In [Ros00] and [Pap94], it is shown that the random walk technique will not perform well even in case of Horn clauses. They provide specific instances of HornSAT, in which the expected convergence time for the random walk is exponential in the number of variables. Clearly, the random walk needs to be augmented with additional heuristics, if it has work on such formulas.
2. The boolean equivalence problem is defined as the problem of checking whether two CNF formulae,  $\phi_1$  and  $\phi_2$  are equivalent, i.e., whether  $\phi_1 \Leftarrow \phi_2$  and  $\phi_2 \Leftarrow \phi_1$ . This problem is **coNP-complete**, when the formulae are 3CNF formulas, but can be solved in polynomial time in case of 2CNF formulas. A natural extension of our work in this paper concerns the existence of a random walk strategy for the boolean equivalence problem restricted to 2CNF formulas.
3. The problem of checking lattice point non-emptiness of polyhedra represents an obvious generalization of the boolean satisfiability problem, in that SAT problems can be reduced to  $\{0,1\}$  integer programming problems in the obvious way. It would be interesting to see if the random walk technique can be used to check feasibility in 2SAT polytopes [Sub02].
4. The problems of UNIQESAT and CRITICALSAT are solvable in polynomial time, in case of 2CNF formulas; however, the only known method is through multiple calls to 2SAT oracles. Developing randomized algorithms for these problems would be instructive.

## References

- [APT79] Bengt Aspvall, Michael F. Plass, and Robert Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas. *Information Processing Letters*, 8(3), 1979.
- [Coo] S. A. Cook. The complexity of theorem-proving procedure. In *ACM STOC 3, NY*.
- [GJ91] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*. W. H. Freeman Company, San Francisco, 1991.
- [Pap91] C. H. Papadimitriou. On selecting a satisfying truth assignment. In IEEE, editor, *Proceedings: 32nd annual Symposium on Foundations of Computer Science, San Juan, Puerto Rico, October 1–4, 1991*, pages 163–169, 1109 Spring Street, Suite 300, Silver Spring, MD 20910, USA, 1991. IEEE Computer Society Press.
- [Pap94] Christos H. Papadimitriou. *Computational Complexity*. Addison-Wesley, New York, 1994.
- [Ros00] Sheldon M. Ross. *Probability Models*. Academic Press, Inc., 7th edition, 2000.
- [sat] Proceedings of SAT 2001, SAT 2002 and SAT 2003.
- [Sch78] T.J. Schaefer. The complexity of satisfiability problems. In Alfred Aho, editor, *Proceedings of the 10th Annual ACM Symposium on Theory of Computing*, pages 216–226, New York City, NY, 1978. ACM Press.
- [Sub02] K. Subramani. On identifying simple and quantified lattice points in the 2sat polytope. In et. al. Jacques Calmet, editor, *Proceedings of the 5<sup>th</sup> International Conference on Artificial Intelligence and Symbolic Computation (AISC)*, volume 2385 of *Lecture Notes in Artificial Intelligence*, pages 217–230. Springer-Verlag, July 2002.
- [WS02] W. Wei and Bart Selman. Accelerating random walks. In *International Conference on Constraint Programming (CP), LNCS*, 2002.