An optical solution for the SAT problem

Mihai Oltean, Oana Muntean
Department of Computer Science,
Faculty of Mathematics and Computer Science,
Babeş-Bolyai University, Kogălniceanu 1,
Cluj-Napoca, 400084, Romania.
{moltean,oanamuntean}@cs.ubbcluj.ro
http://www.cs.ubbcluj.ro/∼moltean/optical

November 2, 2010

Abstract

We describe a delay-based optical device for solving the the Satisfiability problem.

keywords: NP-complete, optical computing, SAT

1 Introduction

Recently, an increased number of problems have been proposed to be solved using optical computers. Hamiltonian path [1, 11, 14], Travelling Salesman [1, 3, 4], subset sum [1, 12, 13], exact cover [12], Diophantine equations [12], 3-SAT [1, 15], SAT [7], prime factorization [9, 10], security [6], arithmetic operations [5], Boolean algebra [8] are just few of the problems whose solution can be found by using an optical device.

Here we show how to solve the Satisfiability (SAT) problem, which is an NP-complete problem [2]. The underlying mechanism is to use delays for encoding possible solutions. The problem was also attacked in [1] and in [15] in a different manner: by using masks and filters for wavelengths.

The paper is organized as follows: Properties of the signal useful for our research and the operations performed on that signal are described in section 2. Section 3 contains a brief description of the problem. Section 4 deeply describe the proposed device. Reconstructing the solution is described in section 5. A short discussion of the weaknesses of this method is given in section 6. Section 7 concludes our paper.

2 Delay-based systems

Two properties of signal are of great interest for our research. Most types of signal that we know (light, sound, electric etc) have these properties.

- The speed of the signal has a limit. We can delay any signal by forcing it to pass through a cable of a certain length.
- The signal can be easily divided into multiple signals of smaller intensity/power.

The following manipulations of the signals are performed within the devices:

- When the signal passes through an arc it is delayed by the amount of time assigned to that arc.
- When the signal passes through a node it is divided into a number of signals equal to the out degree of that node. Every obtained signal is directed toward one of the nodes connected to the current node. In this way we add parallelism to our devices. This feature is actually the source for a major drawback: due to repetitive divisions the strength of the signal decreases exponentially and more and more powerful signals are required for larger and larger instances of the problems.

3 The SAT problem

We have n Boolean variables (x_1, x_2, ... ,x_n) and m clauses of form ($C = x_k \vee x_j \vee$ ...), where a variable can also appear negated. We are asking to find if there is an assignment for variables which satisfies a formula:
$F = C_1 \wedge C_1 \wedge ....C_m$

In the first step we are not interested in finding the value of the variables. Rather, we are interested in finding whether such assignment does exist. Only in section 5 we will show how a solution can be reconstructed.

4 The device

Our device is represented as an incomplete matrix. On each column we have all the variables (as they appear) inside a clause. Thus, we have a matrix with m columns and max n elements on each row.

Example

Consider the formula:
$F = (x_1 \vee x_2 \vee \overline{x}_3) \wedge x_1 \wedge (\overline{x}_2 \vee \overline{x}_3) \wedge (x_1 \vee \overline{x}_3)$

We represent it as a matrix with 4 columns:

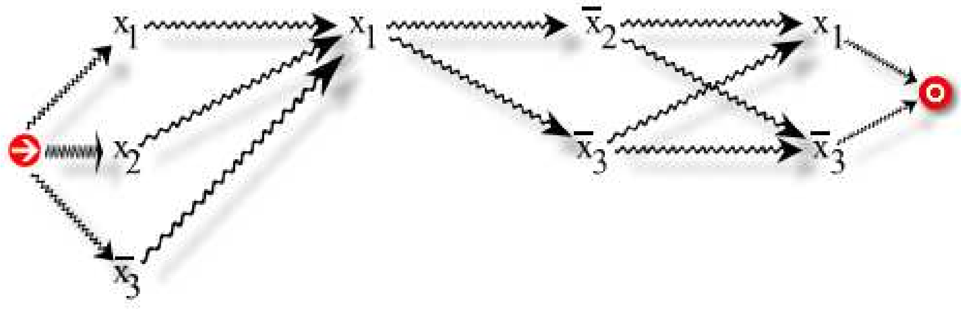
$$\begin{array}{cccc} x_1 & x_1 & \overline{x}_2 & x_1 \\ x_2 & & \overline{x}_3 & \overline{x}_3 \\ \overline{x}_3 & & & \end{array}$$

Next we have to design a device where the signal will flow. For this purpose we do the followings:

- We place each element of this matrix in a Cartesian coordinates system.
- We connect each element with all elements from the next column.
- We also add 2 extra elements (nodes) called *Start* and *Destination*.

The device is represented in Figure 1.

Figure 1: Optical implementation for the SAT problem. On each column we have a clause. On the left side we have the *Start* node and on the right side we have the *Destination* node. Arcs connecting nodes have 0 delays



If we find a path that links *Start* with *Destination* and which does not contain both the variable and its negation it means that the formula is satisfiable. We call that path **satisfiable path**.

For our example, a satisfiable path is: $Start, x_1, x_1, \overline{x}_2, x_1, Destination$.

The big challenge is to identify such path.

4.1 The delaying system

When solving the SAT problem with an optical system we assign delays to elements of the graph. Those delays will accumulate as the signal traverse the graph. At the end we will know that we had a solution only if we have received a signal having a certain (precomputed) delay.

Here we assign some delays to nodes in order to easily recognize the satisfiable paths.

To arcs connecting nodes we assign a negligible delay (very small - or even 0) compared to all other delays assigned to nodes.

4.1.1 The first variable

Let us now focus on variable x_1 and its negation $\overline{x}_1$.

Delay	Content of path	Satisfiable?
0	neither x_1 or $\overline{x}_1$ appear	YES
1	x_1	YES
2	$x_1 + x_1$	YES
3	$x_1 + x_1 + x_1$	YES
4	$\overline{x}_1$	YES
5	$\overline{x}_1 + x_1$	NO
6	$\overline{x}_1 + x_1 + x_1$	NO
7	$\overline{x}_1 + x_1 + x_1 + x_1$	NO
8	$\overline{x}_1 + \overline{x}_1$	YES
9	$\overline{x}_1 + \overline{x}_1 + x_1$	NO
10	$\overline{x}_1 + \overline{x}_1 + x_1 + x_1$	NO
11	Not possible	-
12	$\overline{x}_1 + \overline{x}_1 + \overline{x}_1$	YES
13	$\overline{x}_1 + \overline{x}_1 + \overline{x}_1 + x_1$	NO
14	Not possible	-
15	Not possible	-
16	$\overline{x}_1 + \overline{x}_1 + \overline{x}_1 + \overline{x}_1$	YES

Table 1: The delays. The *Start* and *Destination* have not been printed. Also, other variables have not been displayed. *Notpossible* is because we have only 4 clauses, so the path cannot have more than 4 variables.

Suppose that we have a formula with 4 clauses. What are the minimal numbers that we can assign as delays for x_1 and $\overline{x}_1$ in order to safely decide that x_1 and its negation $\overline{x}_1$ do not (both) appear on a path from *Start* to *Destination*?

For instance, if we assign a delay of 1 for x_1 and a delay of 2 for $\overline{x}_1$ we get some trouble recognizing if the path whose delay is 4 is satisfiable or not. In this case, we can have a path containing 4 times x_1 (which is satisfiable) or a path containing 2 times x_1 and 1 time $\overline{x}_1$ (which is not satisfiable). Note that (having 4 clauses) x_1 can appear at most 4 times and $\overline{x}_1$ can appear at most (4 - number of appearances of x_1) times.

Let's try another arrangement: $delay(x_1) = 1$ and $delay(\overline{x}_1) = 4$. The number of clauses is again 4.

The analysis of paths' delay is given in Table 1.

If we are talking in the number of clauses (m) we have the following delays: $delay(x_1) = 1$ and $delay(\overline{x}_1) = m$. There is no problem if $delay(\overline{x}_1) = m * delay(x_1)$ because we have only m clauses and we cannot have both m times x_1 and 1 time $\overline{x}_1$.

In Table 2 we have the generalized values from Table 1.

If we look to the values in Table 2 we can see that only some delays encode a satisfiable path. More specific: only values $0...m$, $2m$, $3m$, ... m^2 encode a satifiable path. All other values encode either a path that is either not possible (more than m variables on it) or is not satisfiable (contains both x_1 and $\overline{x}_1$).

Delay	Content of path	Satisfiable?
0	neither x_1 or $\overline{x}_1$ appear	YES
$1...m-1$	x_1 only	YES
m	$\overline{x}_1$ only	YES
$m+1...2m-1$	both $\overline{x}_1 + x_1$	NO
$2m$	$\overline{x}_1$ only	YES
...	...	...
$m * m$	only $\overline{x}_1$	YES

Table 2: The generalized delays (m is the number of clauses) for a single variable. Some delays are not possible (for instance $3m - 1$ - see Table 1 for an example.)

Having said that we have ended the discussion about the first variable.

4.1.2 The second variable

Let us move our attention to the next variable: x_2. We cannot assign delays less or equal to m^2 because this will lead to conflicts to the previous variable. What we can do is to assign: delay(x_2) = m^2 and delay($\overline{x}_2$) = m^3 (obtained with the same reasoning as in the case of the first variable).

In this case the delays encoding a satisfiable path are: $m^2, 2m^2, 3m^2, ..., m^3$, $2m^3, 3m^3, ..., m^4$.

4.1.3 The other variables

For the next variable (x_3) we can assign: delay(x_3) = m^4 and delay($\overline{x}_3$) = m^5 (obtained with the same reasoning as in the case of the first variable).

In this case the delays encoding a satisfiable path are: $m^4, 2m^4, 3m^4, ..., m^5$, $2m^5, 3m^5, ..., m^6$.

For the k^{th} variable we can assign: delay(x_k) = m^{2k-2} and delay($\overline{x}_k$) = m^{2k-1} (obtained with the same reasoning as in the case of the first variable).

In this case the delays encoding a satisfiable path are: $m^{2k-2}, 2m^{2k-2}, 3m^{2k-2}$, ..., m^{2k-1}, $2m^{2k-1}, 3m^{2k-1}, ..., m^{2k}$.

4.1.4 Combining delays

The SAT formula is made up of clauses containing multiple variables (not only single variables as we discussed so far in sections 4.1.1, 4.1.2 and 4.1.3).

What we have to do is to add the delays. If we take into account only variables x_1 and x_2 (and their negation) we have to add each delay for variable x_1 (or its negation) to each delay for variable x_2 (or its negation). In total are $4m^2$ moments when a signal traversing a satisfiable path arrives in the destination.

We obtain the following moments when the signal traversing a satisfiable path can arrive in the destination:

m^2,

m^2+1, m^2+2, m^2+3, m^2+m, m^2+2m, m^2+2m, ... , $2m^2$,
$2m^2+1$, $2m^2+2$, $2m^2+3$, $2m^2+m$, $2m^2+2m$, $2m^2+2m$, ... , $3m^2$,
....
m^3+1, m^3+2, m^3+3, m^3+m, m^3+2m, m^3+2m, ... , m^3+m^2,
$2m^3$, $2m^3+1$, $2m^3+2$, $2m^3+3$, $2m^3+m$, $2m^3+2m$, $2m^3+2m$, ... , $2m^3+m^2$,
...,
m^4, m^4+1, m^4+2, m^4+3, m^4+m, m^4+2m, m^4+2m, ... , m^4+m^2.

This set of delays is only for 2 variables (including their negation). For n variable we would have $(2m)^n$ possible moments for listening for a solution.

The main advantage is that these delays do not depend on the actual problem to be solved. These delays do not take into account what literals we have in each clause. This is very important because we can construct the arcs encoding delays and time filters once and we can use them again and again for different instances of the problem (with m clauses and n literals). This is a big advantage.

5 Reconstructing the solution

Once we have a satisfiable delay lets see how to reconstruct which variables are inside it and what is their form (positive or negated).

First of all we discuss some special cases, where we have exactly delay of m or m^3 or m^5 etc units.

For instance if we have a delay of exact m it means that we have exactly m times x_1. We cannot have a delay of exact m which contains $\overline{x}_1$ because the path contains m nodes (excluding *Start* and *Destination*) and those nodes will increase the delay to more than m (which is already induced by $\overline{x}_1$).

If the total delay is m^3 it means that we have x_2 m times. It cannot contain $\overline{x}_2$ because the path must have other $m-1$ nodes which will add more delay.

Thus, for the particular cases when the total delay is of form m^{2k-1}, $(1 \leq k \leq n)$ we know exactly what nodes the satisfiable path contains. These particular cases will be handled separately.

Lest see how we discover the content of the path for all other cases. First we divide the delay by : m^{2n} (this value corresponds to the delay of $\overline{x}_n$ - being the highest delay possible). We have 2 cases:

- If the quotient is 1 ... m it means that $\overline{x}_n$ belongs to the path. It means that variable x_n will be assigned value 0.
- If the quotient is 0 it means that $\overline{x}_n$ does not belong to the path.

Next we take the remainder of the previous division and we repeat the process with delay corresponding to the previous variable. The order is $\overline{x}_n$, x_n, $\overline{x}_{n-1}$, x_{n-1}, ... $\overline{x}_1$, x_1. If the quotient is greater than 0, it means that the variable (positive or negated) belongs to the path. We assign a value to the variable based on the form in which has appeared (0 if negated, 1 if positive). If a variable has not appeared in the path, it can take any value (0 or 1) because this has no influence over the value of formula F from section 3.

6 Weaknesses

The proposed device has some weaknesses:

- it contains exponential delays,
- it requires time-filters for events that appear in time,
- the number of possible moments when a solution can appear is exponential in the number of clauses. This is different from other problems (see [12]) where only one moment was enough for detecting the solution,
- no proof yet for the optimality of the delay system. Are shorter delays possible ?

7 Conclusion

We have shown how SAT can be solved with an optical device.

References

[1] Shlomi Dolev, Hen Fitoussi: The Traveling Beams Optical Solutions for Bounded NP-Complete Problems. FUN 2007, LNCS 4475, pp. 120-134 (2007)

[2] Garey, MR., Johnson, DS.: Computers and intractability: A guide to NP-Completeness. Freeman & Co, San Francisco, CA (1979)

[3] Haist, T., Osten, W.: An Optical Solution For The Traveling Salesman Problem. Opt. Express, Vol. 15, pp. 10473-10482 (2007)

[4] Haist, T., Osten, W.: An Optical Solution For The Traveling Salesman Problem:erratum. Opt. Express, Vol. 15, 12627-12627 (2007)

[5] Haist, T., Osten, W.: Ultrafast Digital-Optical Arithmetic Using Wave-Optical Computing. OSC'08, LNCS 5172, pp. 33-45 (2008)

[6] Haist, T., Osten W.: Proposal for Secure Key Distribution Using Classical Optics. OSC'09, LNCS 5882, pp. 99-101 (2009)

[7] Head, T.: Parallel Computing by Xeroxing on Transparencies. Algorithmic Bioprocesses, part 9, pp. 631-637 (2009)

[8] Head, T.: Using Light to Implement Parallel Boolean Algebra. DLT'10, LNCS 6224, pp. 231-242 (2010)

[9] Nitta, K., Katsuta, N., Matoba, O.: Improvement of a System for Prime Factorization Based on Optical Interferometer. OSC'09, LNCS 5882, pp. 124-129 (2009)

[10] Kouichi Nitta, Nobuto Katsuta, Osamu Matoba: A Method for Modulo Operation by Use of Spatial Parallelism. LNCS 5172, pp. 98-103 (2008)

[11] Oltean, M.: A light-based device for solving the Hamiltonian path problem. Unconventional Computing, Calude C. (et al.) (Eds), LNCS 4135, Springer-Verlag, 217-227 (2006)

[12] Oltean, M., Muntean, O.: Solving NP-Complete Problems with Delayed Signals: An Overview of Current Research Directions, OSC 2008, Springer-Verlag, LNCS 5172, pp. 115-127 (2008)

[13] Hasan, R., Rahman, S.: Computing a Solution for the Subset Sum Problem with a Light Based Device. OSC'09, LNCS 5882, pp. 70-76 (2009)

[14] Shaked, NT., Messika, S., Dolev, S., and Rosen, J.: Optical solution for bounded NP-complete problems, Applied Optics, Vol. 46, 711-724 (2007)

[15] Goliaei, S., Jalili S.: An Optical Wavelength-Based Solution to the 3-SAT Problem. OSC'09, LNCS 5882, pp. 77-85 (2009)

[16] Woods, D., Naughton T.J.: Parallel and Sequential Optical Computing. OSC'08, LNCS 5172, pp. 70-86, (2008)