On a Computing Model Based on Optical Interference

M. V. Panduranga Rao*

IISc-CSA-TR-2005-10
http://archive.csa.iisc.ernet.in/TR/2005/10/

Computer Science and Automation
Indian Institute of Science, India

July 2005

*pandurang@csa.iisc.ernet.in

On a Computing Model Based on Optical Interference

M. V. Panduranga Rao*

Abstract

We propose a computing model, the Two-Way Optical Interference Automata (2OIA), that makes use of the phenomenon of optical interference. We introduce this model to investigate the increase in power, in terms of language recognition, of a classicalDeterministic Finite Automaton (DFA) when endowed with the facility of optical interference. The question is in the spirit of Two-Way Finite Automata With Quantum and Classical States (2QCFA) [A. Ambainis and J. Watrous, Two-way Finite Automata With Quantum and Classical States, Theoretical Computer Science, 287 (1), 299-311, (2002)] wherein the classical DFA is augmented with a quantum component of constant size. We test the power of 2OIA against the languages mentioned in the above paper. We give efficient 2OIA algorithms to recognize languages for which 2QCFA machines have been shown to exist, as well as languages whose status vis-a-vis 2QCFA has been posed as open questions. Finally we show the existence of a language that cannot be recognized by any 2OIA but can be recognized by a Turing machine in polynomial time.

1 Introduction

Recent times have seen a heightened interest in harnessing the laws of physics for computing. For example, models that use quantum mechanical effects to achieve computational speedup have been investigated in great detail (see [8]).

In this paper we present a model of automata, the *Two-Way Optical Interference Automata* (OIA), that uses the phenomenon of optical interference to recognize languages. The question that we address is the following: Given

*pandurang@csa.iisc.ernet.in

a language $L \subseteq \{a, b\}^*$, an input $w \in \{a, b\}^*$, and a 2DFA augmented with $|w|$ "sources of interference", is it possible to decide efficiently if $w \in L$ by examining their interference patterns?

While this question seems interesting in its own right, we believe that this model might also be useful in obtaining bounds for various quantum finite automata (QFA), which make use of interference between probability amplitudes in quantum mechanical systems of small size [1, 3, 6, 7]. In what follows, we assume a knowledge of basic automata theory and quantum computing terminology. See [5] and [8] respectively.

Ambainis and Watrous [1] proposed a model called the 2-way Finite Automata with Quantum and Classical states (2QCFA), which is essentially a classical 2DFA that reads input off a read-only tape and is augmented with a quantum component of constant size: the number of dimensions of the associated Hilbert space does not depend on the input length. Unitary operations on the quantum component, which are performed interleaved with classical transitions, evolve the quantum state vector, producing interference in probability amplitudes of the vector. The classical component takes the result of measurement operations on the quantum state vector in deciding membership of a given input string. Ambainis and Watrous [1] showed 2QCFA that accept $L_{eq} = \{a^n b^n \mid n \in \mathbb{N}\}$ in polynomial time and palindromes ($L_{pal} = \{ww^R \mid w \in \{a, b\}^*$ where w^R is w reversed$\}$) in exponential time with bounded error.

In this paper, we augment the 2DFA with an array of sources of monochromatic light and a detector. In the spirit of 2QCFA, we produce interference between light waves emerging from these sources. In a sense, the model that we propose in this paper abstracts out the phenomenon of interference from quantum automata models. The interference produced by the sources is the analogue of unitary operations on the quantum part and detection of light by the detector is the analogue of the measurement operation. We do not address implementation issues in this paper. Indeed, the motivation behind the model is essentially theoretical.

The paper is organized as follows. The next section gives a brief introduction to some principles of optics pertinent to our model. In section 3 we define the model. Section 4 presents 2OIA algorithms for recognizing the following languages:

- $L_{centre} = \{w_1 a w_2 \mid w_1, w_2 \in \{a, b\}^*, |w_1| = |w_2|\}$.
- $L_{eq} = \{a^n b^n \mid n \in \mathbb{N}\}$.
- $L_{pal} = \{ww^R \mid w \in \{a, b\}^*$ where w^R is w reversed$\}$.

- $L_{bal} = \{w \in \{(,)\}^* \mid \text{parentheses in } w \text{ are balanced}\}$.
- $L_{sq} = \{a^n b^{n^2} \mid n \in \mathbb{N}\}$.
- $L_{pow} = \{a^n b^{2^n} \mid n \in \mathbb{N}\}$.

Section 5 shows the existence of a language that no 2OIA can recognize, but that can be recognized by a Turing machine in time polynomial in the size of the input. Section 6 concludes the paper and discusses open problems.

2 Preliminaries

We shall assume a knowledge of basic automata theory and briefly state the mathematical formalism for interference of monochromatic light of wavelength λ. The equation of a light wave at a point $\mathbf{p}$ in space can be described as $R = Ae^{i(\omega t+\phi)}$, where A is the amplitude, ω the angular frequency, and ϕ the phase associated with the wave at that point. The amplitude A at $\mathbf{p}$ is A_0/r, where r is its distance from the source and A_0 the initial amplitude at $r = 0$. The *intensity* of the wave is the average energy arriving per unit time per unit area. At any given point, it is proportional to the square of the amplitude of the wave. If two (or more) waves exist at the same point in space, they *interfere* with each other and give rise to a new resultant wave. Suppose the waves have the same angular frequency and are described by $R_1 = A_1 e^{i(\omega t+\phi_1)}$ and $R_2 = A_2 e^{i(\omega t+\phi_2)}$. Then, the resultant is given by $R = R_1 + R_2 = A_1 e^{i\omega t} e^{i\phi_1} + A_2 e^{i\omega t} e^{\phi_2}$. The amplitude of the resultant wave is the length of R. Since R is a complex number, the amplitude is $\sqrt{R\bar{R}}$. Thus, the intensity associated with the wave is proportional to $A_1^2 + A_2^2 + 2A_1A_2\cos(\phi_2 - \phi_1)$. If two interfering waves have equal amplitude and a phase difference $\phi_2 - \phi_1 = \pi$, the resultant intensity at that point is zero. A light detector placed at that point fails to detect any light. To use this formalism for calculating the intensity at a point, only the difference $\phi_2 - \phi_1$ is necessary. This phase difference might arise because of (a) the intrinsic phase difference at their source and (b) the difference Δ between the distances traveled by each before reaching that point. Hence, $\phi_2 - \phi_1 = \alpha + \frac{2\pi}{\lambda}\Delta$ where the first term is the difference in the intrinsic phases and the second is the difference between the distances in terms of phase. In general, if n monochromatic waves $\{R_i = A_i e^{i\omega t+\phi_i}\}_{i=1}^n$ interfere at a point, the resultant wave is described by the vector sum $R = \sum_{i=1}^n A_i e^{i\omega t+\phi_i}$. A useful point to note is that if the sources are simply different points on the same wavefront, they have zero intrinsic phase difference. Moreover, the phase of a wave can be shifted as required. For excellent expositions on the phenomenon of optical interference see [2] and [4].

3 The Model

Informally, the interference automaton proposed in this paper consists of a finite control, an optical arrangement and a read-only tape which contains the input w demarcated by end-markers ¢ and \$ in $|w|+2$ cells. The optical arrangement consists of a linear array of $|w|+2$ monochromatic light sources, each source corresponding to a tape cell, capable of emitting light with an initial relative phase of 0 or π. These sources can only be *toggled*, as will be explained in the formal definition. The distance between any two consecutive sources is the same, a constant independent of the size of the input.

We have a detector whose movement is dictated by the finite control. Since the control is finite, the movement of the detector has to be discretized. The easiest way to do this is to imagine that the detector moves along the lines of a grid placed before the array of sources as shown in figure 1.

The detector "points" in the direction of the source array, parallel to the vertical grid-lines. The maximum angle θ that an incoming ray of light can form with a vertical grid-line at the detector and still be detected is called the *field of vision* (FOV) of the detector. The ratio of the sides of each cell of the grid is given by $\tan\theta$. In this paper, we use $\theta = \pi/4$. Thus, each grid cell is a tiny square. We associate a coordinate system with the grid, with the vertical grid-lines at $x = 0, 1/2, 1, 3/2, \ldots, n+1$ and the horizontal grid-lines at $y = 0, 1/2, 1, 3/2, \ldots, n+1$. The light sources are placed at $(0,0), (1,0), \ldots, (n+1,0)$. While the sources at $(0,0)$ and $(n+1,0)$ correspond to the end markers ¢ and \$ respectively, those at $(1,0), \ldots, (n,0)$ correspond to $w_1, \ldots, w_n$ respectively, where $w = w_1 w_2 \ldots w_n$ is the input.

Thus, the position of the tape head is also referred to by its x-coordinate. The location of the detector is given by (i,j) with $i,j \in \{0, 1/2, 1, 3/2, \ldots, n+1\}$.

We now define the model formally.

Definition 1 *A 2-way Optical Interference Automaton (2OIA) is a 7-tuple* $(Q, \Sigma, Q_{acc}, Q_{rej}, q_0, \Sigma_I, \delta)$ *where*

- Q *is a finite set of states.*
- Σ *is a finite alphabet.*
- $\Gamma = \Sigma \bigcup \{ ¢, \$\}$ *is the tape alphabet.*
- $Q_{acc}, Q_{rej} \subseteq Q$ *are sets of accepting states and rejecting states respectively.*
- q_0 *is a special state designated as the start state.*

- $\Sigma_I = \{\underline{0}, \underline{1}\}$ *is the output alphabet of the detector.*

- *The* Instantaneous Description *of a 2OIA is given by the tuple* $(q, i, (j, k), S)$, *where* $q \in Q$, $i \in \{0, \ldots n+1\}$ *is the position of the head on the tape,* (j, k), $j, k \in \{0, 1/2, 1, 3/2, \ldots, n+1\}$ *gives the position of the detector, and* S *is a vector of length* $n+2$ *with entries from* $\{0, \pi, \times\}$, *signifying if a source corresponding to a tape cell is switched on with phase* 0 *or* π, *or switched off, respectively.*

  δ *is a transition function–*

  $\delta : Q \times \Gamma \times \Sigma_I \rightarrow Q \times D_t \times D_d \times (toggle(\phi) \cup -)$

  where $D_t = \{$*left, right, no move*$\}$, $D_d = \{$*left, right, up, down, no move*$\}$ *and*

  $$toggle(\phi) = \begin{cases} \text{switch on the current source with phase } \phi & \text{if the source is off} \\ \text{switch the current source off} & \text{if the source is already on} \end{cases}$$

  where $\phi \in \{0, \pi\}$ *and "–" indicates that no action is to be taken on the current source. By current source we mean the source associated with the tape cell currently being scanned by the head.*

  If $\delta(q, \sigma, \sigma_I) = (q', d_h, (d_l, d_u), p)$ *then we denote the change in the instantaneous description during this* one *step as* $(q, i, (j, k), S) \vdash_1 (q', i', (j', k'), S')$ *where* $i' = i + d_h$, $j' = j + d_l$, $k' = k + d_u$ *and*

  $$\begin{aligned} S'[m] &= S[m] \text{ for } m \neq i \\ &= toggle(p) \text{ for } m = i, \text{ if } p \in \{0, \pi\} \\ &= S[m] \text{ for } m = i \text{ if } p = - \end{aligned}$$

  Further, we place the following restriction on those moves of the transition function that involve toggling the sources:

  $$(q, i, (j, k), S) \vdash_2 (q', i', (j', k'), S') \tag{1}$$

  where $S[i] \neq S'[i]$, *is allowed only twice for each* i *during the computation.*

□

The detector has an output alphabet through which it can indicate whether it has detected any light. It responds with a $\underline{1}$ if it has, and with a $\underline{0}$ if it has not.

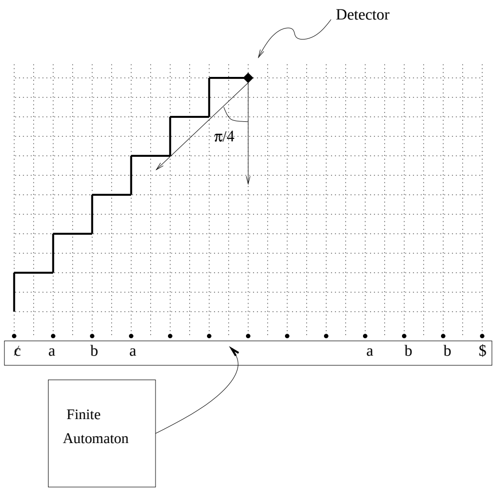

Figure 1: The arrangement. Also illustrates the algorithm for recognizing L_{centre}. The source at \$ comes into the FOV at the position where the detector is shown.

Depending on the current state q, the symbol $\sigma \in \Gamma$ currently being scanned, and the output of the detector, δ changes the state of the finite control to q' and moves the tape head by $d_t \in D_t$ and the detector by $d_d \in D_d$.

The primary use of the sources is to produce optical interference at the grid. However, unrestricted toggling would enable their use as memory elements: the detector when close to the array, could "read" an individual source while the finite control could "write" to it by toggling it. In order to avoid this, we place restrictions on moves that involve toggle operations. Any two consecutive steps that toggle a given source exactly once is considered a *non-transient* toggle operation. We restrict the number of such non-transient operations on a given source to two. Attempting a non-transient operation a third time on the same source *crashes* the machine. Since transient toggling of the kind $(q, i, (j, k), S) \vdash_1 (q', i, (j', k'), S') \vdash_1 (q', i', (j'', k''), S)$ cannot result in the source being used as a memory element, such operations are permitted any number of times.

The sum of the number of moves made by the head and the detector serves as a measure of the time taken by a 2OIA machine.

The following facts and conventions will be common to all 2OIA machines in this paper.

- Initially, all the sources are switched off.

- $Q_{acc} = \{q_{acc}\}$ and $Q_{rej} = \{q_{rej}\}$ will be the accepting and rejecting subsets. The machines accept by final state and $\underline{0}$ detector output.

- The transition function is specified as a matrix, the rows of which are indexed by $q \in Q$ and columns by (σ, I), with $\sigma \in \Gamma$ and $I \in \Sigma_I$. The matrix are elements tuples of the form $(q', t, (x, y), \phi)$ with $q' \in Q$, $t \in D_t$, $x \in \{\leftarrow, -, \rightarrow\}$, $y \in \{\uparrow, -, \downarrow\}$ and $\phi \in \{0, -, \pi\}$. In any field of a tuple, "$-$" stands for a "no change" in its value. Although D_t and D_d imply movement by only one unit (tape cell and grid-line respectively), movement by two units can be carried out by the finite control. Indeed, we use arrows to indicate movement by two units.

  An element indexed by row q and column $(a, \underline{1})$, say, $(q', \rightarrow, (\uparrow, \rightarrow), 0)$ is to be interpreted as follows. If the machine is in state q and the head is reading an a, and the detector outputs $\underline{1}$, then the source corresponding to the current cell is switched on with a phase 0, tape head moves right to the next tape cell, the detector moves first upwards and then to the right by two grid lines, and the machine enters state q'. In some places we show both movements in the same entry for the sake of brevity, but the two are not atomic. If the detector output changes after moving up but before moving to the right, the configuration changes as dictated by the transition function.

4 Language recognition

In this section we show recognition schemes for the languages mentioned in section 1. In all cases we show the existence of a locus of points on the grid where the intensity of light will be zero if and only if the input word is in the language. The idea behind all algorithms on this model is to move the detector to the locus.

We begin with a simple but important example.

4.1 L_{centre}

A word in $L_{centre} = \{w_1 a w_2 \mid w_1, w_2 \in \{a, b\}^*, |w_1| = |w_2|\}$ will be of odd length (say $2n + 1$). This can be easily verified with a DFA. We accept the string, if the $(n+1)^{th}$ element is a.

Theorem 1 *There exists a 2OIA machine that recognizes L_{centre} in time linear in size of the input string.*

Proof: As proof, we describe such a machine. Consider a 2OIA M_{centre} having a set of states $Q = \{q_0, q_1, q_2, q_{rej}, q_{acc}\}$ and a transition function as specified by the following table.

	¢,$\underline{0}$	¢,$\underline{1}$	a,$\underline{0}$	a,$\underline{1}$
q_0	$(q_0, \rightarrow, (-,-), 0)$	ϕ	ϕ	$(q_0, \rightarrow, (-,-), -)$
q_1	ϕ	$(q_2, -, (-,-), -)$	ϕ	$(q_1, \leftarrow, (-,-), -)$
q_2	ϕ	$(q_2, \rightarrow, (\rightarrow, \uparrow), -)$	$(q_{acc}, -, (-,-), -)$	$(q_2, \rightarrow, (\rightarrow, \uparrow), -)$

	b, $\underline{0}$	b,$\underline{1}$	\$, $\underline{0}$	\$, $\underline{1}$
q_0	ϕ	$(q_0, \rightarrow, (-,-), -)$	ϕ	$(q_1, \leftarrow, (-,-), \pi)$
q_1	ϕ	$(q_1, \leftarrow, (-,-), -)$	ϕ	ϕ
q_2	$(q_{rej}, -, (-,-), -)$	$(q_2, \rightarrow, (\rightarrow, \uparrow), -)$	ϕ	ϕ

The machine starts with the detector initially in position $(0, 1)$ and the tape head at 0, reading ¢. While in state q_0, the sources at ¢ and \$ are switched on with phase 0 and π respectively in one rightward scan of the input and back.

Lemma 1 *The detector will record zero intensity if and only if its x-coordinate is $n + 1$.*

Proof: The resultant wave at the detector is

$$\psi = \frac{A_0}{\sqrt{i^2 + j^2}} e^{i\phi_1} + \frac{A_0}{\sqrt{(2n + 2 - i)^2 + j^2}} e^{i\phi_2}$$

for $j \geq 1$. Given that $\phi_2 = \phi_1 + \pi$, the detector reads zero resultant intensity if and only if its x-coordinate is $n+1$. $\square$ Note that to begin with, the source corresponding to \$ is out of the detector's FOV. However, the intensity at the detector due to the source at ¢ is non-zero. The detector then moves upwards and to the right in such a way that at all times during the movement, it can detect light from the source at ¢. The head also moves to the right *in tandem* with the detector. When the detector reaches a certain x-coordinate, the source at \$ falls into the detector's FOV and the resultant intensity falls to zero. By the above lemma, this is the geometric centre of the input. If at this moment, the head reads an a, it accepts, else it rejects.

It is easy to see that the automaton takes $O(n)$ time to decide the language. $\square$

Using the idea behind the 2OIA algorithm for L_{centre}, we can recognize a related language, namely $L_{eq} = \{a^n b^n \mid n \in \mathbb{N}\}$. This language can be recognized with bounded error on 2QCFA [1].

Corollary 1 *The above algorithm can be used to recognize L_{eq} in linear time.*

Proof: The DFA first verifies that the input is indeed of the form a^*b^* and that the input is indeed of even length. Next, the centre is detected using the above algorithm. The input is of even length and therefore, the detector has to be initially positioned at $(1/2, 1)$. The DFA then checks if the current symbol being read is a and that to the right is b. If it is not, reject the input, else accept. This too takes time linear in the input size.

4.2 Palindromes

In this section we give a 2OIA machine that recognizes the language $L_{pal} = \{ww^R \mid w \in \{a, b\}^* \text{ where } w^R \text{ stands for } w \text{ reversed}\}$. We use light of wavelength $r\pi$ where r is any algebraic number greater than zero. This restriction on the wavelength yields algorithms that are more elegant and faster, and brings out the main features of the model better. Later in the section we give a machine that recognizes L_{pal} without this restriction.

Theorem 2 *There exists a 2OIA machine that decides L_{pal} in time linear in the input size.*

Proof: We describe a 2OIA M_{pal} that decides L_{pal}. M_{pal} has the state set $\{q_0 \dots q_{10}, q'_3 \dots q'_{10}, q_{acc}, q_{rej}\}$ and a transition function as shown in the following table.

	$¢,\underline{0}$	$¢,\underline{1}$	$a,\underline{0}$	$a,\underline{1}$
q_0	$(q_0, \rightarrow, (-, -), 0)$	ϕ	ϕ	$(q_0, \rightarrow, (-, -), -)$
q_1	ϕ	$(q_2, -, (-, -), -)$	ϕ	$(q_1, \leftarrow, (-, -), -)$
q_2	ϕ	$(q_2, \rightarrow, (\rightarrow, \uparrow), -)$	$(q_3, \rightarrow, (-, -), 0)$	$(q_2, \rightarrow, (\rightarrow, \uparrow), -)$
q_3	ϕ	ϕ	ϕ	$(q_4, \leftarrow, (-, -), -)$
q_4	ϕ	ϕ	$(q_5, \leftarrow, (-, \uparrow), -)$	$(q_4, -, (-, \downarrow), -)$
q_5	ϕ	$(q_6, \rightarrow, (-, -), -)$	ϕ	$(q_5, \leftarrow, (-, -), -)$
q_6	ϕ	ϕ	$(q_7, \rightarrow, (-, -), -)$	$(q_6, \rightarrow, (-, -), 0)$
q_7	ϕ	ϕ	$(q_7, \rightarrow, (-, -), \pi)$	ϕ
q_8	ϕ	ϕ	ϕ	ϕ
q_9	ϕ	ϕ	ϕ	ϕ
q_{10}	ϕ	ϕ	ϕ	ϕ
q'_3	ϕ	ϕ	ϕ	$(q_{rej}, -, (-, -), -)$
q'_4	ϕ	ϕ	ϕ	ϕ
q'_5	ϕ	$(q'_6, \rightarrow, (-, -), -)$	ϕ	$(q'_5, \leftarrow, (-, -), -)$
q'_6	ϕ	ϕ	ϕ	$(q'_6, \rightarrow, (-, -), -)$
q'_7	ϕ	ϕ	$(q'_7, \rightarrow, (-, -), -)$	ϕ

q'_8	ϕ	ϕ	ϕ	ϕ
q'_9	ϕ	ϕ	ϕ	ϕ
q'_{10}	ϕ	ϕ	ϕ	ϕ

	b, $\underline{0}$	b,$\underline{1}$	\$, $\underline{0}$	\$, $\underline{1}$
q_0	ϕ	$(q_0, \rightarrow, (-,-), 0)$	ϕ	$(q_1, \leftarrow, (-,-), \pi)$
q_1	ϕ	$(q_1, \leftarrow, (-,-), -)$	ϕ	ϕ
q_2	$(q'_3, \rightarrow, (-,-), 0)$	$(q_2, \rightarrow, (\rightarrow, \uparrow), -)$	ϕ	ϕ
q_3	ϕ	$(q_{rej}, -, (-,-), -)$	ϕ	ϕ
q_4	ϕ	ϕ	ϕ	ϕ
q_5	ϕ	$(q_5, \leftarrow, (-,-), -)$	ϕ	ϕ
q_6	ϕ	$(q_6, \rightarrow, (-,-), -)$	ϕ	ϕ
q_7	$(q_7, \rightarrow, (-,-), -)$	ϕ	$(q_8, -, (-,-), -)$	ϕ
q_8	ϕ	ϕ	$(q_9, -, (-, \uparrow), \pi)$	ϕ
q_9	ϕ	ϕ	$(q_8, -, (-,-), \pi)$	$(q_{10}, -, (-,-), \pi)$
q_{10}	ϕ	ϕ	$(q_{acc}, -, (-,-), -)$	$(q_{rej}, -, (-,-), -)$
q'_3	ϕ	$(q'_4, \leftarrow, (-,-), -)$	ϕ	ϕ
q'_4	$(q'_5, \leftarrow, (-, \uparrow), -)$	$(q'_4, -, (-, \downarrow), -)$	ϕ	ϕ
q'_5	ϕ	$(q'_5, \leftarrow, (-,-), -)$	ϕ	ϕ
q'_6	$(q'_7, \rightarrow, (-,-), -)$	$(q'_6, \rightarrow, (-,-), 0)$	ϕ	ϕ
q'_7	$(q'_7, \rightarrow, (-,-), \pi)$	ϕ	$(q'_8, -, (-,-), -)$	ϕ
q'_8	ϕ	ϕ	$(q'_9, -, (-, \uparrow), \pi)$	ϕ
q'_9	ϕ	ϕ	$(q'_8, -, (-,-), \pi)$	$(q'_{10}, -, (-,-), \pi)$
q'_{10}	ϕ	ϕ	$(q_{acc}, -, (-,-), \pi)$	$(q_{rej}, -, (-,-), -)$

The automaton, in states q_0, q_1 and q_2, finds the centre as in the previous section. Since the input is of even length, say $2n$, at this stage the head reads the n^{th} input symbol $w_n = \sigma \in \{a, b\}$ and the detector is on the grid-line $x = n + 1/2$. The case of w_n being a is handled by states $q_3 \dots q_{10}$, while w_n being b is handled by the states $q'_3 \dots q'_{10}$. By the arrangement of the sources and the fact that $\Sigma = \{a, b\}$, it is sufficient to show symmetry of any one $\sigma \in \Sigma$ about the centre of the input string. We show the proof for the case when $w_n = a$. The argument for the other case is symmetric.

If $w_n \neq w_{n+1}$, the input is rejected at the outset (see rows indexed by q_2 and q_3). Else, switch on the source corresponding to w_n and bring the detector to $(n + 1/2, 1/2)$ so that only the sources corresponding to w_n and w_{n+1} lie in the FOV of the detector (row q_4). This is done by using the fact that as the detector moves down along the grid-line $x = n+1/2$, the detector falls to $\underline{0}$ when the source corresponding to w_n moves out of its FOV, that is, at $y = 1/2$.

Then, the head starts from w_1, switching on all sources corresponding to a and with phase 0 until it reaches w_n (rows q_5 and q_6). We can deduce when the head has reached the centre because when the source at w_{n+1} is toggled, the detector records zero intensity.

As the head moves further to the right, all sources from w_{n+1} to w_{2n} that correspond to a are switched on with relative phase π (row q_7).

Now that all the sources corresponding to a have been switched on appropriately, we move the detector away from the x-axis along the grid-line $x = n + 1$, bringing two sources (one from each side of the centre) into its FOV at each step.

Let us now note a useful lemma.

Lemma 2 *Suppose sources to the left of $x = n + 1/2$ are switched on with phase 0 and those to the right with phase π. Then, the resultant intensity at $x = n + 1/2$ is zero if and only if either (1) for every j^{th} source to the left of $x = n + 1/2$ switched on, the j^{th} source to the right of $x = n + 1/2$ is also switched on and vice versa or (2) both sources are off.*

Therefore, as the detector is moved away, if the intensity is non-zero at any step i, we know that the input is not a palindrome, and reject it. Otherwise, we stop when ¢ and \$ fall into the FOV of the detector and accept. We can detect when this happens by periodically toggling the source corresponding to \$. This is done in states q_8 to q_{10}. All we need now is a proof of the above lemma.

Proof: (of lemma 2) The j^{th} source from the centre on either side may or may not be switched on, depending on the input. Let the boolean variables a_j and b_j indicate whether the j^{th} source is switched on to the left and right of $x = n + 1/2$ respectively. Thus, the resultant wave at $x = n + 1/2$ is

$$(a_1 - b_1)r_1e^{i\phi_1} + (a_2 - b_2)r_2e^{i\phi_2} + \ldots + (a_n - b_n)r_ne^{i\phi_n}$$

where $\phi_j = j\sqrt{2}\pi d/\lambda$ for $1 \leq j \leq n$, and d distance between two consecutive vertical (or horizontal) grid-lines.

Therefore, we have to prove that the resultant is zero if and only if $a_j = b_j$, for $1 \leq j \leq n$. One direction is trivial. For the other direction we use the following theorem of Lindemann (see [9]).

Theorem 3 *Given any distinct algebraic numbers $\phi_1, \phi_2, \ldots, \phi_n$, the values $e^{\phi_1}, e^{\phi_2}, \ldots, e^{\phi_n}$ are linearly independent over the field of algebraic numbers.*

Since we have chosen $\lambda = r\pi$ for some algebraic number r, we have $\phi_j = j\sqrt{2}d/r$ for $1 \le j \le n$. Therefore, the resultant can be zero if and only if $a_j = b_j$, $1 \le j \le n$. $\Box$

The running time of the algorithm is $O(n)$ as the head scans the input only a constant number of times and the detector movement is also $O(n)$. $\Box$

We now present a 2OIA that recognizes L_{pal} without restriction on the wavelength, but at the cost of increased time complexity.

Theorem 4 *There exists a 2OIA machine that recognizes L_{pal} in $O(n^2)$ time.*

Proof: Without loss of generality let w_n and w_{n+1} be a. If the centre is $x = x_c$, then for every a to the left of $x = x_c$ we try to find the corresponding a at the same distance to the right, and vice versa. For the sake of brevity, we describe a 2OIA machine M'_{pal} that checks only one way: if for every a to the left of the centre, there exists an a to the right. In doing so, sources corresponding to a's of only the first half of the input string will be toggled twice in a non-transient manner. Therefore, when the extended 2OIA for L_{pal} checks for the existence of an a to the left of x_c corresponding to every a to the right, the sources in the second half of the input can be toggled non-transiently. The set of states and the transition matrix for the complete case will involve a simple and symmetric extension of M'_{pal}.

Consider the 2OIA M'_{pal} with states $\{q_0 \ldots q_7, q'_3 \ldots q'_7, q_{acc}, q_{rej}\}$ and transition function as shown in the following table. The detector is initially at $(1/2, 1/2)$.

	¢,$\underline{0}$	¢,$\underline{1}$	a,$\underline{0}$	a,$\underline{1}$
q_0	$(q_0, \rightarrow, (-,-), 0)$	ϕ	ϕ	$(q_0, \rightarrow, (-,-), -)$
q_1	ϕ	$(q_2, -, (-,-), -)$	ϕ	$(q_1, \leftarrow, (-,-), -)$
q_2	ϕ	$(q_2, \rightarrow, (\rightarrow, \uparrow), -)$	$(q_3, -, (-,-), -)$	$(q_2, \rightarrow, (\rightarrow, \uparrow), -)$
q_3	$(q_{acc}, -, (-,-), -)$	ϕ	$(q_4, \rightarrow, (-,-), 0)$	ϕ
q_4	ϕ	ϕ	ϕ	$(q_5, -, (-,-), \pi)$
q_5	ϕ	ϕ	$(q_6, \leftarrow, (-,-), \pi)$	$(q_4, \rightarrow, (-,-), \pi)$
q_6	ϕ	ϕ	ϕ	$(q_7, -, (-,-), 0)$
q_7	ϕ	ϕ	$(q_3, \leftarrow, (-,-), -)$	$(q_6, \leftarrow, (-,-), 0)$
q'_3	$(q_{acc}, -, (-,-), -)$	ϕ	$(q'_3, \leftarrow, (-,-), -)$	ϕ
q'_4	ϕ	ϕ	ϕ	$(q'_4, \rightarrow, (-,-), -)$
q'_5	ϕ	ϕ	ϕ	ϕ
q'_6	ϕ	ϕ	ϕ	$(q'_6, \leftarrow, (-,-), -)$
q'_7	ϕ	ϕ	ϕ	ϕ

	b, $\underline{0}$	b,$\underline{1}$	\$, $\underline{0}$	\$, $\underline{1}$
q_0	ϕ	$(q_0, \rightarrow, (-,-), 0)$	ϕ	$(q_1, \leftarrow, (-,-), \pi)$
q_1	ϕ	$(q_1, \leftarrow, (-,-), -)$	ϕ	ϕ
q_2	$(q'_3, -, (-,-), -)$	$(q_2, \rightarrow, (\rightarrow, \uparrow), -)$	ϕ	ϕ
q_3	$(q_3, \leftarrow, (-,-), -)$	ϕ	ϕ	ϕ
q_4	ϕ	$(q_4, \rightarrow, (-,-), -)$	ϕ	$(q_{rej}, -, (-,-), -)$
q_5	ϕ	ϕ	ϕ	ϕ
q_6	ϕ	$(q_6, \leftarrow, (-,-), -)$	ϕ	ϕ
q_7	ϕ	ϕ	ϕ	ϕ
q'_3	$(q'_4, \rightarrow, (-,-), 0)$	ϕ	ϕ	ϕ
q'_4	ϕ	$(q'_5, -, (-,-), \pi)$	ϕ	$(q_{rej}, -, (-,-), -)$
q'_5	$(q'_6, \leftarrow, (-,-), \pi)$	$(q'_4, \rightarrow, (-,-), \pi)$	ϕ	ϕ
q'_6	ϕ	$(q'_7, -, (-,-), 0)$	ϕ	ϕ
q'_7	$(q'_3, \leftarrow, (-,-), -)$	$(q'_6, \leftarrow, (-,-), 0)$	ϕ	ϕ

The automaton begins by finding the centre of the input (rows q_0, q_1 and q_2).

Note that at this point, the detector is at $(n+1/2, n+1/2)$. By lemma 1, if only two sources are switched on, and with opposite phases, the detector reads $\underline{0}$ if and only if they are equidistant from the centre. Thus, searching for matching pair involves the following steps:

- For an a to the left, switched on with 0, search right for the corresponding a as follows. On encountering an a, the corresponding source is switched on (the source being off initially, toggling switches it on) with a phase π. If the detector reads $\underline{0}$, we have found the a that we were looking for, and the head returns left. Else, we continue to search to the right. In any case, the source is switched off again. If the head hits \$ without the detector reading a $\underline{0}$ it implies that input is not a palindrome and is rejected (rows q_3, q_4 and q_5 for $w_n = a$ and q'_3, q'_4 and q'_5 for $w_n = b$).

- If the a that we were looking for in the previous step is found, the corresponding source is switched off (making the detector again read $\underline{1}$), and the head returns to the left a. The catch here is that the finite control cannot remember the position of the left a. However, since the source corresponding to the right a has been switched off, the only source that is switched on is the one corresponding to the left a. Thus, during the leftward scan, if toggling a source corresponding to an a drops the detector reading to $\underline{0}$ again, we have found the left a (rows q_6 and q_7 for $w_n = a$ and q'_6 and q'_7 for $w_n = b$).

- Search left for another a. If found, repeat the above process. If not, that is, if the head hits ¢, accept (row q_3 for $w_n = a$ and q_3' for $w_n = b$).

Since every symbol in the input is scanned at most $O(n)$ times, and the detector moves by only $O(n)$ steps during the execution, the machine takes $O(n^2)$ time.

4.3 Balanced Parentheses

We now show a 2OIA machine that recognizes L_{bal} using a combination of the techniques for palindromes given in the previous subsection.

Theorem 5 *There exists a 2OIA machine that recognizes L_{bal} in $O(n^3)$ time, where n is the input length.*

Proof: Let us first define two useful terms.

Definition 2 *A string in $\{(,)\}^*$ is called a* simple nest *if it consists of n (s followed by n)s, for $n \geq 1$.*

Definition 3 *A string in $\{(,)\}^*$ is called a* compound nest *if it consists of n '('s followed by m simple or compound nests, followed by n)s, for $m > 1$ and $n \geq 1$.*

A string in L_{bal} is a concatenation of these two types of substrings. We define a 2OIA M_{bal} that deals with the two cases separately. It has set of states $Q = \{q_0 \dots q_{15}, q_{rej}, q_{acc}\}$ and transition function as shown in the following table.

	¢,$\underline{0}$	¢,$\underline{1}$	(,$\underline{0}$	(,$\underline{1}$
q_0	$(q_0, \rightarrow, (-,-), 0)$	ϕ	ϕ	$(q_0, \rightarrow, (-,-), 0)$
q_1	ϕ	$(q_2, \rightarrow, (-,-), -)$	ϕ	$(q_1, \leftarrow, (-,-), -)$
q_2	ϕ	ϕ	ϕ	$(q_2, \rightarrow, (\rightarrow,-), -)$
q_3	$(q_{rej}, -, (-,-), -)$	$(q_4, \rightarrow, (-,\downarrow), -)$	$(q_3, \leftarrow, (-,\uparrow), -)$	$(q_4, \rightarrow, (-,\downarrow), -)$
q_4	ϕ	ϕ	$(q_4, \rightarrow, (-,-), 0)$	$(q_4, \rightarrow, (-,-), 0)$
q_5	ϕ	ϕ	$(q_{11}, -, (-,-), -)$	ϕ
q_6	ϕ	ϕ	ϕ	ϕ
q_7	$(q_{rej}, -, (-,-), -)$	ϕ	ϕ	$(q_8, -, (-,-), 0)$
q_8	ϕ	ϕ	$(q_8, \rightarrow, (-,-), -)$	$(q_7, \leftarrow, (-,-), 0)$
q_9	ϕ	ϕ	$(q_{10}, -, (-,-), 0)$	ϕ
q_{10}	ϕ	ϕ	$(q_{11}, \leftarrow, (\rightarrow,-), 0)$	$(q_9, \rightarrow, (-,-), 0)$
q_{11}	ϕ	ϕ	ϕ	ϕ
q_{12}	ϕ	ϕ	ϕ	ϕ
q_{13}	$(q_{acc}, -, (-,-), -)$	ϕ	$(q_{13}, \leftarrow, (-,-), -)$	$(q_{rej}, -, (-,-), -)$

	), $\underline{0}$	),$\underline{1}$	\$, $\underline{0}$	\$, $\underline{1}$
q_0	ϕ	$(q_0, \rightarrow, (-,-), \pi)$	ϕ	$(q_1, \leftarrow, (-,-), -)$
q_1	ϕ	$(q_1, \leftarrow, (-,-), -)$	ϕ	ϕ
q_2	$(q_3, \leftarrow, (-,-), -)$	$(q_2, \rightarrow, (\rightarrow,-), -)$	ϕ	$(q_{13}, \leftarrow, (-,-), -)$
q_3	$(q_4, \rightarrow, (-,\downarrow), -)$	$(q_4, \rightarrow, (-,\downarrow), -)$	ϕ	ϕ
q_4	ϕ	$(q_5, -, (-,-), -)$	ϕ	ϕ
q_5	$(q_6, -, (-,\uparrow), -)$	$(q_5, \rightarrow, (-,-), \pi)$	$(q_{13}, \leftarrow, (-,-), -)$	ϕ
q_6	$(q_7, -, (-,-), \pi)$	$(q_6, -, (\leftarrow,\uparrow), -), -)$	ϕ	ϕ
q_7	ϕ	$(q_7, \leftarrow, (-,-), -)$	ϕ	ϕ
q_8	$(q_9, \rightarrow, (-,-), -)$	ϕ	ϕ	ϕ
q_9	$(q_{10}, -, (-,-), \pi)$	ϕ	$(q_{13}, \leftarrow, (-,-), -)$	ϕ
q_{10}	$(q_6, -, (-,\uparrow), \pi)$	$(q_8, -, (-,-), \pi)$	ϕ	ϕ
q_{11}	ϕ	$(q_{12}, -, (-,-), \pi)$	ϕ	ϕ
q_{12}	$(q_2, \rightarrow, (-,-), \pi)$	$(q_{12}, -, (\rightarrow,\downarrow), -)$	ϕ	ϕ
q_{13}	$(q_{13}, \leftarrow, (-,-), -)$	ϕ	ϕ	ϕ

Initially, the sources corresponding to the '($'$s and ')$'$s are switched on with phase 0 and π respectively (while in state q_0). The head reads ¢ and the detector is placed at $(1/2, 1/2)$.

The head moves to the right ignoring the '($'$s on the way, with the detector moving in tandem (row q_2).

Simple nests are dealt with as follows:

- Since every '($'$ is switched on with phase 0 and ')$'$ with π, if the intensity at the detector is zero when the head is reading a ')$'$, we conclude that the detector's current x-coordinate is the centre of a simple nest.

- The detector moves up and the head to the left until the leftmost '($'$ and the rightmost ')$'$ of the nest lie in the FOV of the detector. Thus, the movement stops when (a) the detector outputs a $\underline{1}$ and/or (b) the head reads ¢ or a ')$'$. If the detector outputs $\underline{0}$ while the head is reading ¢, it implies imbalance: since the source for \$ is not switched on, it means that ¢ has been balanced by a). Therefore, the input is rejected. If the detector outputs a $\underline{1}$, the edge of the current nest has been detected. The head moves one cell to the right and the detector one step down, again outputing a $\underline{0}$ (rows q_2, q_3 and q_4).

- Now the head, moving right, toggles (in effect, switches off) all the symbols of this nest. By the same argument as in the previous section, the detector output changes to $\underline{1}$ as soon as the first symbol is toggled.

It reverts to $\underline{0}$ only when all the symbols corresponding to this nest are switched off (rows q_4 and q_5).

An important point to note is that the sources corresponding to a "recognized nest" (simple or compound) are never switched on in a non-transient manner again during computation. We now turn to compound nests. For the rest of the proof we abuse the notation a bit by calling those parentheses in the compound nest that are not a part of an inner simple nest as the compound nest itself. A compound nest is recognized as follows.

- To begin with, the head reads the innermost ')' of the compound nest, say at cell x, and the detector is at $(x, 1/2)$. The detector travels to the left and away from the source array until it reaches a position on the grid where it reads $\underline{0}$ (row q_6). This is easy because of the fact that all sources corresponding to simple nests and smaller compound nests contained inside the current compound nest have been switched off.

- Once the centre x_c of the innermost '(' and ')' of the compound nest has been located, the head moves to the left to detect the corresponding '(' and switch it off. At this instant, when the innermost pair of the compound nest has been discovered and switched off, the detector reads a $\underline{0}$. If the head hits ¢ before this happens, it implies imbalance, namely excess ')''s and the machine rejects. The head returns to the right to the ')' of the pair (rows q_7 and q_8). These two steps are performed in the same way as described earlier for simple nests.

- If the next symbol is again a), we repeat the above two steps. If not, the detector has to be brought into position $(x - 1/2, 1/2)$, where x is the x-coordinate of the next symbol (rows $q_9 \ldots q_{12}$). If it is '(', this is the beginning of a new nest. If it is \$, we have accounted for all ')'s. All we need to do now is to check if any '(' is left unpaired. At this point, all balanced parentheses are switched off. Thus, if during a leftward scan, the detector outputs $\underline{1}$, we conclude that there exists an extra '(' and reject the input. If however, the detector does not output a $\underline{1}$ before reaching ¢, we accept the input (row q_{13}).

Therefore, the machine accepts if and only if the input has balanced parentheses.

Recognizing a simple nest of $2m$ symbols takes at most $O(m)$ moves of the head and $O(m)$ moves of the detector. For a compound nest of m symbols, having l symbols of enclosed simple nests, at most $O(m^2 l)$ moves of the head and $O(l)$ moves of the detector are required. Since m and l are bounded from above by n, the machine takes at most $O(n^3)$ time.

4.4 $a^n b^{n^2}$

Theorem 6 *There exists a 2OIA that recognizes $a^n b^{n^2}$ in $O(|w|^3)$ time, where $|w|$ is the length of the input.*

Proof: Intuitively, the 2OIA works as follows. It "measures out" blocks of b's of length n and keeps a count of the blocks measured out thus far using the a's. For every source corresponding to an a switched on, we measure out and mark a block of b's. The first and last source of each block is switched on with phase 0 and π respectively and the interference at the centre of the block last measured out is used to measure out the next block. An input x is in the language if and only if the total number of blocks measured out is exactly equal to the number of a's in x.

The 2OIA consists of states $\{s_0, s_1, s_2\} \cup \{f_1, \ldots, f_7\} \cup \{b_1, \ldots, b_6\} \cup \{f'_1, \ldots, f'_6\} \cup \{q_{acc}, q_{rej}\}$. The transition function is defined as follows:

	$¢,\underline{0}$	$¢,\underline{1}$	$a,\underline{0}$	$a,\underline{1}$
s_0	$(s_0, \rightarrow, (\rightarrow, -), -)$	ϕ	$(s_0, \rightarrow, (-, -), 0)$	$(s_0, \rightarrow, (-, -), -)$
s_1	ϕ	ϕ	ϕ	$(s_2, \rightarrow, (-, -), \pi)$
s_2	ϕ	ϕ	ϕ	ϕ
f_1	ϕ	ϕ	ϕ	ϕ
f_2	ϕ	ϕ	ϕ	ϕ
f_3	ϕ	ϕ	ϕ	ϕ
f_4	ϕ	ϕ	ϕ	ϕ
f_5	ϕ	ϕ	ϕ	ϕ
f_6	ϕ	ϕ	ϕ	ϕ
f_7	ϕ	ϕ	ϕ	ϕ
b_1	ϕ	$(b_2, \rightarrow, (\rightarrow, -), -)$	$(b_1, \leftarrow, (\leftarrow, -), -)$	$(b_1, \leftarrow, (\leftarrow, -), -)$
b_2	ϕ	ϕ	$(b_3, \rightarrow, (\rightarrow, -), 0)$	$(b_2, \rightarrow, (\rightarrow, -), -)$
b_3	ϕ	ϕ	$(b_4, \rightarrow, (\rightarrow, -), -)$	$(b_6, -, (-, -), -)$
b_4	ϕ	ϕ	$(b_4, \rightarrow, (\rightarrow, -), -)$	$(b_4, \rightarrow, (\rightarrow, -), -)$
b_5	ϕ	ϕ	ϕ	ϕ
b_6	ϕ	ϕ	ϕ	ϕ
b_7	ϕ	ϕ	ϕ	ϕ
f'_1	ϕ	ϕ	ϕ	ϕ
f'_2	ϕ	ϕ	ϕ	ϕ
f'_3	ϕ	ϕ	ϕ	ϕ
f'_4	ϕ	ϕ	ϕ	ϕ
f'_5	ϕ	ϕ	ϕ	ϕ
f'_6	ϕ	ϕ	ϕ	ϕ

	$b,\underline{0}$	$b,\underline{1}$	$\$,\underline{0}$	$\$,\underline{1}$
s_0	ϕ	$(s_1, \leftarrow, (-,-), -)$	ϕ	$(q_{rej}, -, (-,-), -)$
s_1	ϕ	ϕ	ϕ	ϕ
s_2	$(f_2, -, (\rightarrow, -), 0)$	$(s_2, -, (\rightarrow, \uparrow), -)$	ϕ	ϕ
f_1	$(f_2, -, (\rightarrow, -), -)$	$(f_1, -, (\leftarrow, \uparrow), -)$	ϕ	ϕ
f_2	$(f_3, -, (\rightarrow, -), -)$	$(f_2, -, (\rightarrow, -), -)$	ϕ	ϕ
f_3	$(f_4, \leftarrow, (\leftarrow, -), -)$	$(f_3, \rightarrow, (\rightarrow, -), -)$	ϕ	$(q_{rej}, -, (-,-), -)$
f_4	ϕ	$(f_5, -, (-,-), \pi)$	ϕ	q_{rej}
f_5	$(f_6, \rightarrow, (-,-), -)$	$(f_4, \rightarrow, (-,-), \pi)$	ϕ	ϕ
f_6	$(f_7, \leftarrow, (-,-), 0)$	ϕ	$(q_{rej}, -, (-,-), -)$	ϕ
f_7	$(b_1, -, (-,-), -)$	$(f_7, -, (\rightarrow, \downarrow), -)$	ϕ	ϕ
b_1	$(b_1, \leftarrow, (\leftarrow, -), -)$	$(b_1, \leftarrow, (\leftarrow, -), -)$	ϕ	ϕ
b_2	ϕ	ϕ	ϕ	ϕ
b_3	ϕ	ϕ	ϕ	ϕ
b_4	$(b_4, \rightarrow, (\rightarrow, -), -)$	$(b_4, \rightarrow, (\rightarrow, -), -)$	$(b_5, \leftarrow, (\leftarrow, -), -)$	ϕ
b_5	$(b_5, \leftarrow, (\leftarrow, -), -)$	$(f_1, -, (-,-), -)$	ϕ	ϕ
b_6	$(b_6, \rightarrow, (\rightarrow, -), -)$	$(b_6, \rightarrow, (\rightarrow, -), -)$	$(b_7, \leftarrow, (\leftarrow, -), -)$	ϕ
b_7	$(b_7, \leftarrow, (\leftarrow, -), -)$	$(f'_1, -, (-,-), -)$	ϕ	ϕ
f'_1	$(f'_2, -, (\rightarrow, -), -)$	$(f'_1, -, (\leftarrow, \uparrow), -)$	ϕ	ϕ
f'_2	$(f'_3, -, (\rightarrow, -), -)$	$(f'_2, -, (\rightarrow, -), -)$	ϕ	ϕ
f'_3	$(f'_4, \leftarrow, (\leftarrow, -), -)$	$(f'_3, \rightarrow, (\rightarrow, -), -)$	ϕ	$(q_{rej}, -, (-,-), -)$
f'_4	ϕ	$(f'_5, -, (-,-), \pi)$	ϕ	$(q_{rej}, -, (-,-), -)$
f'_5	$(f'_6, \rightarrow, (-,-), -)$	$(f'_4, \rightarrow, (-,-), \pi)$	ϕ	ϕ
f'_6	q_{rej}	ϕ	$(q_{acc}, -, (-,-), -)$	ϕ

To begin with, the sources corresponding to the first and last a are switched on with phase 0 and π respectively, and the first block of b's is measured out and marked.

The algorithm consists of several iterations, each of which consists of the following steps:

1. **Bookkeeping phase (states $b_1 - b_7$):** The i^{th} ($i > 1$) iteration begins by switching the source corresponding to the i^{th} a with phase 0. The detector is close to the source array so that it can read each source individually. Starting from the beginning of the input, the detector and the source are moved to the right in tandem, ignoring sources that are already switched on. The first source that registers a $\underline{0}$ at the detector is switched on. That done, the detector and the head set out to find the rightmost unmarked block, which actually corresponds to

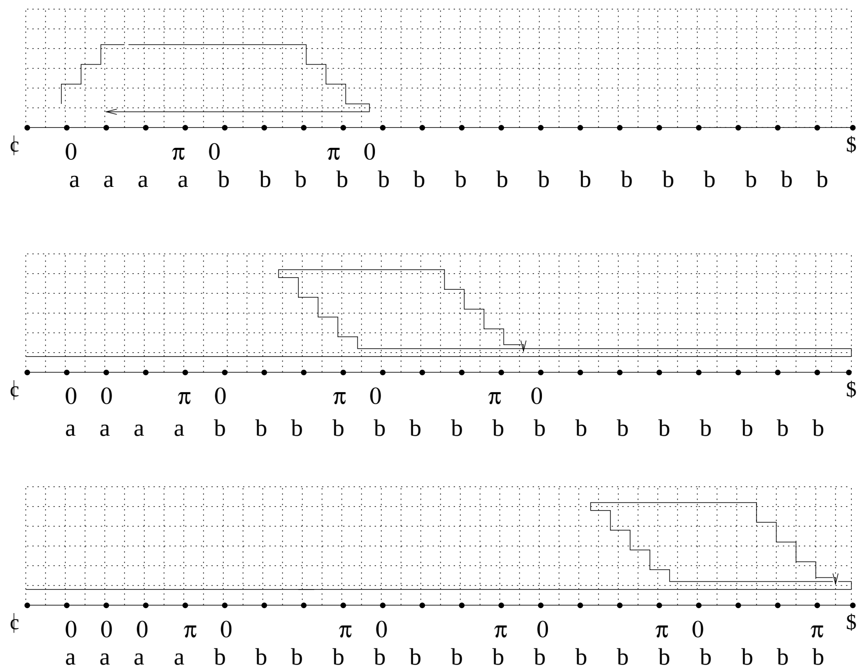

Figure 2: Recognizing $a^n b^{n^2}$. The bold lines along the grid-lines show the trajectory of the detector during various phases.

this a. This is accomplished by first travelling to the end of the input string, again with the detector and head in tandem, and returning left until the head reads a $\underline{1}$. The coordinates of the head and the detector at this moment are $(in+1)$ and $(in+1/2, 1/2)$ respectively. Thus, the $(i-1)^{th}$ block is used for marking out the i^{th} block (see next step). The first iteration, in which the first block of b's is marked out, uses the block of n a's, (states $s_0 - s_2$) in a similar fashion.

2. **Marking the next block (states $f_1 - f_5$ and $f_1' - f_6'$ for the last block):** The detector is taken to $((i-1)n + (n+1)/2, (n+1)/2)$, the centre of the $(i-1)^{th}$ block. Now, the right end of the i^{th} block is detected as follows. The detector moves to the right. The first $\underline{0}$ registered is ignored and the detector keeps moving to the right till the detector reads a $\underline{0}$ for the second time. This happens when the source at $in+1$ falls out of the FOV of the detector. The detector is moved back by 1/2 so that the source again falls into the FOV of the detector and it registers a $\underline{1}$. The detector is therefore at the centre of the next block, that is, the block that is to be marked. The head is now advanced to the right toggling every source on the way by phase π. If the detector records $\underline{0}$, we have found the right end and the head is at $(i+1)n$. On the other hand, if the detector records $\underline{1}$, the source is toggled back again: this is not the right end of the block. If the head hits \$ without recording $\underline{0}$, the input is rejected.

3. **Bringing the detector back in tandem with the detector (states $f_6 - f_7$):** The detector has to be brought to $((i+1)n, 1/2)$, so that the detector and the head can go back in tandem to the start of the input to begin the next iteration. This is done by switching the source corresponding to the next source by phase 0. The detector is then moved to the right and closer to the source array in steps (see figure 2). The detector registers $\underline{1}$ as soon as it moves right from the centre of the i^{th} block. However, it registers $\underline{0}$ again, when its co-ordinate is $(i+1)n - 1/2$. But since it has also been moving closer to the array, by this time, the detector is at a distance of 1 unit from it.

Let the input be $a^n b^m$ for some $n, m \in \mathbb{N}$. Three cases arise:

1. $m = n^2$: After the last a is switched on, the head and the detector set out to find an unmarked block of b's. If a complete block of n b's is found and the symbol immediately after the block is \$, the machine accepts.

2. $m < n^2$: If $m = l^2 < n^2$, then the 2OIA is not in the f' states, indicating that the machine is not looking for the last block yet. Therefore, if the head hits $ immediately after marking a block, the input is rejected. If m is not a perfect square, then the head hits $ while it is still marking a block, and the input is rejected.

3. $m > n^2$: There are residual b's even after marking the block corresponding to the last a, and the input is rejected.

Thus, the machine described above accepts a string if and only if it is of the form $a^n b^{n^2}$ for $n \in \mathbb{N}$. Further, since the head and the detector make one scan of the input for each a, the total time taken is $O(|w|^3)$.

Corollary 2 *There exists a 2OIA that can recognize $a^n b^{2^n}$ in time $O(2^n)$.*

Proof: The 2OIA recognizing $a^n b^{2^n}$ works in a similar fashion. The detector doubles the length of the blocks in each iteration and the a's are used for keep track of the number of blocks of b's.

5 An Upper Bound

Theorem 7 *There exists a polynomial time Turing Machine recognizable language that no 2OIA can recognize.*

Proof: Let $f : \{0,1\}^n \rightarrow \{0,1\}^m$ $m \geq n$ be any polynomial time computable one-one function. Consider a language $L_{fn} \subset \{0,1,\#\}^*$ defined as

$L_{fn} = \{x = \#x_1\#x_2\#\ldots\#x_k \mid$ every $x_i \in \{0,1\}^*$ is either $f(x_{i'})$ for some $i' \neq i$ or $x_j = f(x_i)$ is also present in $x\}$.

If no 2OIA can recognize $\#x_i\#f(x_i)$, for some polynomial time computable function f, we are done. Now suppose that it can be recognized by some 2OIA.

Given x_i and x_j, checking whether $x_j = f(x_i)$ or not can be done by either computing $f(x_i)$ and comparing it with x_j or, if f is invertible, by computing $f^{-1}(x_j)$ and comparing x_i. Since the only enhancement to the finite automaton is the optical arrangement, sources of either x_i or x_j have to be switched on non-transiently to compute and store the bits of $f(x_i)$ or $f^{-1}(x_j)$.

Without loss of generality, let the computation of $f(x_i)$ be done using the sources corresponding to the symbols of x_j. Since x_i is not fixed, the finite control can store neither $|x_i|$ nor x_i itself. Therefore, the value of each bit of

x_i has to be carried individually to x_j by the finite control. However, to do this, the machine has to keep track of the bits that have been picked and those that are yet to be picked, by non-transiently switching on the corresponding sources. In any case, the sources of both x_i and x_j have to be switched on non-transiently . Consequently, non-transient use of these sources is ruled out for the rest of the computation. Thus, if it turns out that $x_j \neq f(x_i)$, verifying other possibilities, viz. $x_j = f(x_{i'})$ for $i \neq i'$, $x_{i'} = f(x_j)$ for any i', $x_{j'} = f(x_i)$ for $j \neq j'$ and $x_i = f(x_{i'})$ for any i', is impossible. Therefore, the 2OIA cannot recognize L_{fn}.

6 Conclusions and Open Problems

In this paper, we proposed a model of computing based on optical interference. We showed machines of this model that recognize some non-trivial languages. The paper leaves the following questions open.

- Does there exist an elegant characterization of this model?
- If the number of initial phases is more than just two (0 and π), then what is the increase in power?
- How does this model compare with 2QCFA in terms of language recognition? Does the set of languages recognized by one model include that recognized by the other? If 2QCFA is strictly more powerful than 2OIA, then our results imply that all the languages posed as open for 2QCFA by Ambainis and Watrous [1] can be recognized by them. If the inclusion is the other way, then 2OIA is an upper bound on the power of 2QCFA. In particular, it would imply that L_{fn} can be recognized by no 2QCFA.
- If the output alphabet of the detector is expanded, that is, if the detector can report different levels of intensity, then what is the increase in power?

References

[1] A. Ambainis and J. Watrous. Two-way finite automata with quantum and classical states. *Theor. Comput. Sci.*, 287(1):299–311, 2002.

[2] M. Born and E. Wolf. *Principles of Optics.* Cambridge University Press, 1999.

[3] Alexander Brodsky and Nicholas Pippenger. Characterizations of 1-way quantum finite automata. *SIAM J. Comput.*, 31(5):1456–1478, 2002.

[4] R. P. Feynman, R. B. Leighton, and M. Sands. *The Feynman Lectures in Physics, Vol. 1.* Addison Wesley, Reading, MA, 1970.

[5] J. E. Hopcroft and J. D. Ullman. *Introduction to Automata Theory, Languages, and Computation.* Addison Wesley, Reading, MA, 1979.

[6] Attila Kondacs and John Watrous. On the power of quantum finite state automata. In *Annual IEEE Symposium on Foundations of Computer Science*, pages 66–75, 1997.

[7] Cristopher Moore and James P. Crutchfield. Quantum automata and quantum grammars. *Theor. Comput. Sci.*, 237(1-2):275–306, 2000.

[8] M. Nielson and I. Chuang. *Quantum Computation and Quantum Information.* Cambridge University Press, 2000.

[9] I. M. Niven. *Irrational Numbers.* New York: Wiley, 1956.