Automata on Infinite Objects

B.Tech Seminar Report

by

Himanshu Jain
Roll No: 99005018

under the guidance of

Prof. Supratik Chakraborty

Department of Computer Science and Engineering
Indian Institute of Technology
Bombay

Abstract

This report is a study of a few finite automata which operate on infinite words (ω-words). In particular Büchi automata and Muller automata have been considered. Boolean properties of these automata (and related constructions) have been studied. Also monadic second order theory of one successor (S1S) have been studied. Expressiveness of Büchi, Muller automata (deterministic and non-deterministic versions) and S1S have been compared. Inter-conversion between these formalisms have been studied.

1 Introduction

Automata on infinite objects (ω-words) were introduced in the early 1960's by Büchi [3], motivated by issues in mathematical logic, viz., the decidability of his monadic second order theory of one successor (S1S). This logical theory was actually referred to by Büchi as the Sequential Calculus because of its potential utility for describing the behavior of sequential circuits. Somewhat later, Muller [4] proposed that automata on infinite strings might themselves be applicable to describing the behavior of circuits, in particular, nonstabilizing asynchronous digital circuits.

An ordinary sequential program starts executing in an initial state corresponding to a given set of input data values. It then executes for some finite number of steps, and then terminates in a particular final state corresponding to a certain set of output data values. On the other hand a continuously operating program is characterized by behavior that is ideally nonterminating. Examples of such programs include computer operating systems, network communication protocols, air traffic control systems. For these programs, termination represents an abnormal condition while normal behavior entails executing forever. Thus, such programs are not naturally described using the initial state/final state semantics used for ordinary sequential programs. The semantics of these programs is instead described in terms of infinite computation sequences recording the states the program goes through as it executes.

If we can model the set of infinite computation sequences, as the language of some finite automaton (on ω-words), then various properties about the program becomes the decision problems on the derived automata. These decision problems involve testing whether the language of given automaton is empty or not. In the next few sections, we consider the study of few such existing automata on infinite words.

In section 2 we give some definitions and notations, which will be used in later sections. In section 3 we begin the study of Büchi automaton and there boolean properties. Construction for intersection and complementation are described through examples.

In section 4 we look at how to specify the properties of infinite sequences in S1S. Then we look at Büchi's Theorem which establishes the connection between S1S and Büchi automaton (BA). An example is given to demonstrate BA to S1S conversion & vice versa.

In section 5 Muller automaton and McNaughton's Theorem is described. There relation with S1S & BA is discussed. In section 6 we draw a set of conclusions from the seminar.

2 Definitions & Notations

- Given a finite alphabet A we define two sets A^* and A^w, where A^* denotes the set of finite words and A^w denotes the set on w-sequences (or w-words, words with cardinality=$|\aleph|$) over A.
- An w-word over A^w is written in form $\alpha = \alpha(0)\alpha(1)...$ where $\alpha(i) \in A$.
- Finite words are denoted as u,v,w,.., empty word by ϵ, ..sets of finite words by U, V, W. Letters α, β, σ.. are used for w-words.
- Notation for Segments of w-words are,
  $\alpha(m,n) := \alpha(m)..\alpha(n-1), m \leq n$
- Logical connectives used are $\neg, \vee, \wedge, \rightarrow, \exists, \forall$. We use $\exists^w n$ for "there exists infinitely many n" and $\exists^{<w} n$ for "finitely many n"
- Important sets on $W \subseteq A^*$

  pref(W) $:= \{u \in A^* | \exists v$, such that $uv \in W\}$

  $W^w := \{\alpha \in A^w | \alpha = w_0 w_1 ...$ with $w_i \in W, \forall i \geq 0\}$

  $\overrightarrow{W} := \{\alpha \in A^w | \exists^w n$ s.t $\alpha(0,n) \in W\}$
- For an w-sequence $\sigma = \sigma_0 \sigma_1 ...$ from S^w, the "infinity set of σ" is defined as In(σ)$:=\{s \in S | \exists^w n, \sigma(n) = s\}$

3 Büchi Automaton

3.1 Definition

Büchi automata are essentially finite automata equipped with an acceptance condition that is appropriate for ω-words.

A Büchi automaton over the alphabet A is of the form $\mathcal{A} = (Q, q_0, \delta, F)$ with the finite state set Q, initial state $q_0 \in Q$, transition relation $\delta \subseteq Q \times A \times Q$, and a set $F \subseteq Q$ of final states.

A run of $\mathcal{A}$ on an ω-word $\alpha = \alpha(0)\alpha(1)....$ from A^w is a sequence $\sigma = \sigma(0)\sigma(1)....$ such that $\sigma(0) = q_0$ and $(\sigma(i), \alpha(i), \sigma(i+1)) \in \delta$ for $i \geq 0$; the run is successful if $In(\sigma) \cap F \neq \phi$, i.e. some state of F occurs infinitely in it. $\mathcal{A}$ accepts α if there is a successful run of $\mathcal{A}$ on α.
Then $L(\mathcal{A}) = \{\alpha \in A^w | \mathcal{A}$ accepts $\alpha\}$ is said to be the w language recognized by $\mathcal{A}$, L is said to be Büchi recognizable.

Figure 1 shows a Büchi automaton which accepts all strings α s.t between any two occurrences of letter a there is an even number of letters b, c.

3.2 Important Characterization

Given some fixed Büchi automaton $\mathcal{A} = (Q, q_0, \delta, F)$, let $W_{ss'}$ represent the regular set of strings in A^*, which takes the $\mathcal{A}$ from start state s to final state s' i.e,

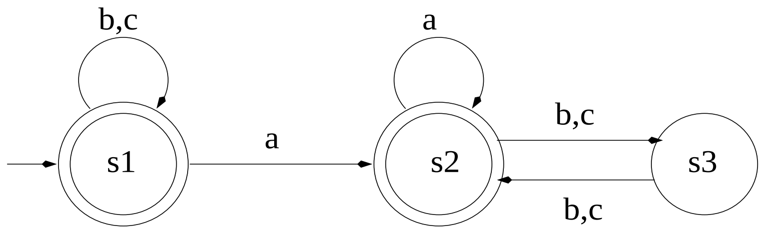

Figure 1: Büchi Automaton $\mathcal{A}$ on alphabet $A = \{a, b, c\}$

$W_{ss'} = \{w \in A^* | s \rightarrow_w s'\}.$

By definition of Büchi Acceptance, the ω-language recognized by $\mathcal{A}$ can be written as,
$\mathcal{L}(\mathcal{A}) = \bigcup_{s \in F} W_{q_0 s}.(W_{ss})^w$

For example, for the automaton in Figure 1. We can write the $L(\mathcal{A}) = W_{11}.W_{11}^w \cup W_{12}.W_{22}^w$ (Subscripts 1,2,3 are used in place of states $s1, s2, s3$).

Theorem 1 [1] *An ω-language $L \subseteq A^w$ is said to be Büchi-recognizable iff L is a finite union of sets $U.V^w$ where $U, V \subseteq A^*$ are regular sets of finite words.*

3.3 Closure Properties of Büchi Recognizable languages

1. If $V \subseteq A^*$ is regular then V^w is Büchi recognizable.
2. If $U \subseteq A^*$ is regular and $L \subseteq A^w$ is Büchi recognizable, then $U.L$ is Büchi Recognizable.
3. If $L_1, L_2 \subseteq A^w$ are Büchi Recognizable, then $L_1 \cup L_2$ and $L_1 \cap L_2$ are Büchi Recognizable.

We will use BA as an acronym for Büchi Automaton in the following discussion. Construction of BA in case 1 & 2 above is quite straightforward. The construction for intersection of two Büchi automata is given below.

Proposition 1: Let $\mathcal{A}_1, \mathcal{A}_2$ be BA. Then there exists a BA $\mathcal{A}$ such that $L(\mathcal{A}) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$.
Construction : Let $\mathcal{A}_1 = (Q_1, q_{01}, \delta_1, F_1)$ and $\mathcal{A}_2 = (Q_2, q_{02}, \delta_2, F_2)$. We define BA for $\mathcal{A}$ as $(Q_1 \times Q_2 \times \{1, 2\}, (q_{01}, q_{02}, 1), \delta, F_1 \times Q_2 \times 1)$, where δ is defined as follows:

$$\delta((q_{i1}, q_{j2}, 1), a) = (\delta_1(q_{i1}, a), \delta_2(q_{j2}, a), 1) \ \textit{if} \ q_{i1} \notin F_1 \ \textit{else} \ (\delta_1(q_{i1}, a), \delta_2(q_{j2}, a), 2)$$
$$\delta((q_{i1}, q_{j2}, 2), a) = (\delta_1(q_{i1}, a), \delta_2(q_{j2}, a), 2) \ \textit{if} \ q_{j2} \notin F_2 \ \textit{else} \ (\delta_1(q_{i1}, a), \delta_2(q_{j2}, a), 1)$$

We can think of BA $\mathcal{A}$ as having two tracks 1 & 2, $\mathcal{A}_1$ runs on one track 1 and $\mathcal{A}_2$ on the other. The third component in the state tuple for $\mathcal{A}$ can be thought as a pointer which points to either track 1 or track 2. Initially the pointer points to first track. Whenever we hit a final state on track 1 , the pointer is changed to track 2. Similarly, whenever we hit a final state on track 2 , the pointer is changed to track 1. The acceptance condition guarantees that both the tracks visit final states infinitely often, since a run accepts iff it goes through some state in $F_1 \times Q_2 \times 1$ infinitely often. This means that first track visits some final state infinitely often with pointer pointing to first track. But whenever we hit such a final state on track one, we change the pointer to track 2 on the next transition. The pointer is restored back to track 1 only when we hit some final state on track 2. Thus the second track must also visit some final state infinitely often.

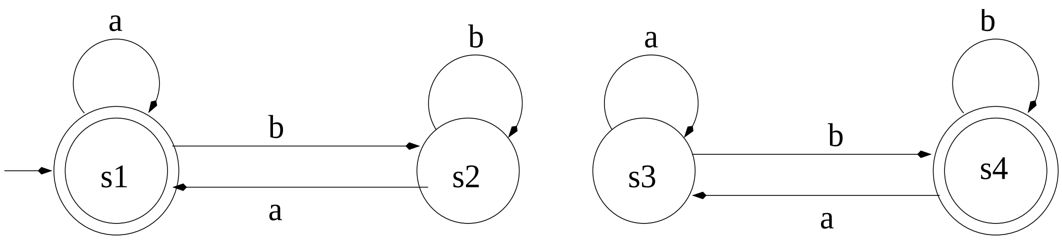

Figure 2: Büchi Automaton $\mathcal{A}_1$ & $\mathcal{A}_2$

In the Figure 2 Büchi Automaton $\mathcal{A}_1$ ($\mathcal{A}_2$) represents the set of ω-words in which $a(b)$ occurs infinitely often.

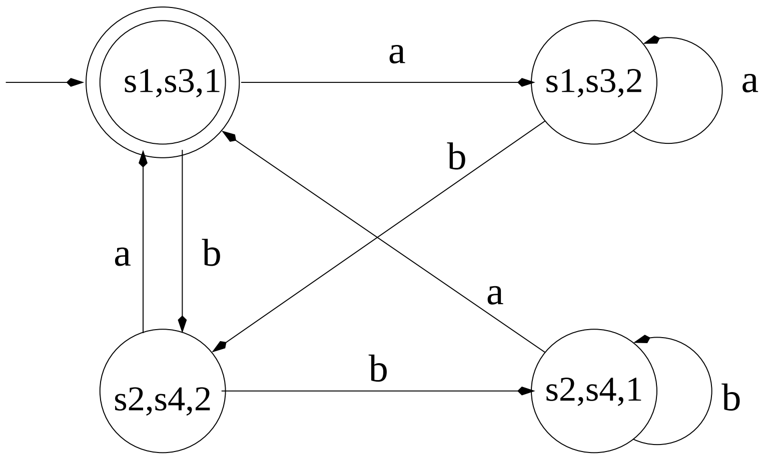

Figure 3: Büchi Automaton A $s.t$ $L(\mathcal{A}) = L(\mathcal{A}_1) \cap L(\mathcal{A}_2)$

Figure 3 is the automaton for intersection of $\mathcal{A}_1$&$\mathcal{A}_2$ (Figure 2). It accepts ω-words in which both a & b occur infinitely often.

The following results are from [1]:
(a)Any nonempty regular ω-language contains an ultimately periodic ω-word(i.e, an ω-word of the form $uvvv$..).
(b) The emptiness problem for Büchi automata is decidable.

3.4 Complementation of Büchi Automata

First we look at complementation of Deterministic Büchi Automata (DBA). Let $\mathcal{A}$ be a DBA. To construct $\mathcal{A}^c$ we make use of two DBA's $\mathcal{A}_1$ and $\mathcal{A}_2$, which are essentially modified copies of $\mathcal{A}$.

1. $\mathcal{A}_1$ is identical to $\mathcal{A}$, but none of its states are accepting. Also starting state of $\mathcal{A}_1$ is same as that of $\mathcal{A}$.

2. $\mathcal{A}_2$ is constructed from $\mathcal{A}$ by removing all its final states and corresponding transitions. The states that are left are all marked as accepting in $\mathcal{A}_2$.

3. Now we need to combine these two copies of $\mathcal{A}$. This is done as follows:
   For every transition from a state s to $s^{'}$ in $\mathcal{A}$, where $s^{'}$ is not an accepting state in $\mathcal{A}$, we put a transition (on same input symbol) from state labelled as s in $\mathcal{A}_1$ to state $s^{'}$ in $\mathcal{A}_2$.

4. The starting state for the combined automaton (call it $\mathcal{A}^{'}$) is the starting state of $\mathcal{A}_1$.

Following figures show the steps involved in construction:

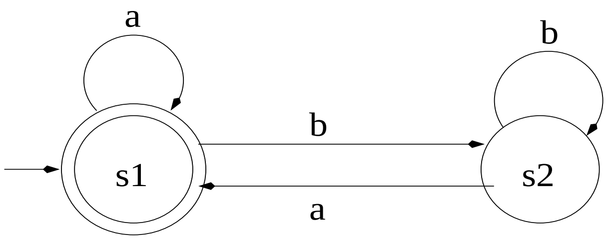

Figure 4: DBA $\mathcal{A}$ which is to be complemented

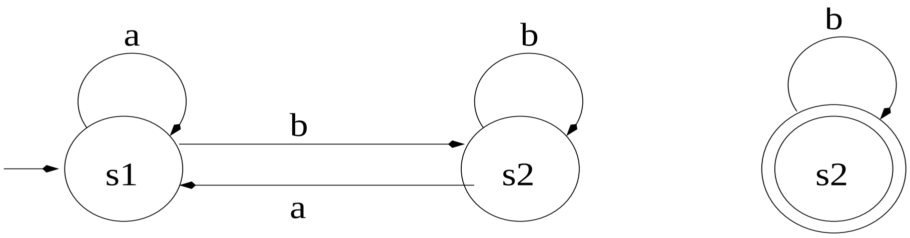

Figure 5: $\mathcal{A}_1$ & $\mathcal{A}_2$ as mentioned in the construction

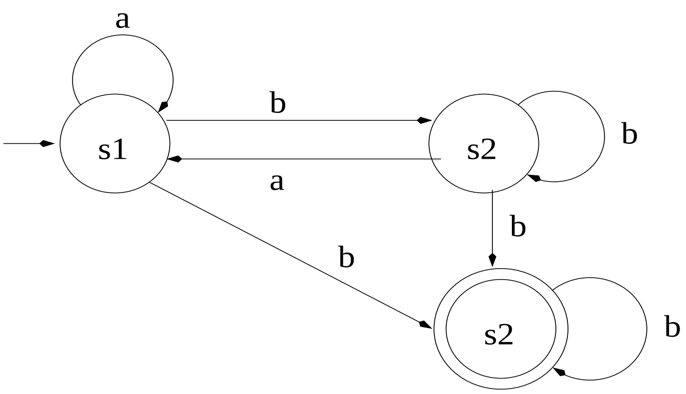

Figure 6: $\mathcal{A}^c$ obtained after combining $\mathcal{A}_1$ & $\mathcal{A}_2$ as above

Note that while $\mathcal{A}$ accepts ω-words in which a occurs infinitely often, $\mathcal{A}^c$ accepts ω-words with only finitely many occurrences of a.

We now intuitively describe why the above construction works.

Let the run of $\mathcal{A}$ on ω-word α be denoted as σ. Since $\mathcal{A}$ is a DBA, every ω-word has a run on $\mathcal{A}$.

- Let $\alpha \notin L(\mathcal{A})$. We need to show that $\alpha \in L(\mathcal{A}')$.Since $\alpha \notin L(\mathcal{A}) \rightarrow \exists k(\forall i(i > k \longrightarrow \sigma(i) \notin F))$, where k, i denotes positions in σ and F is the set of final states for $\mathcal{A}$. Since after position k no final state occurs implies that some non-final state occurs infinitely often. Clearly, the loop in which this state is visited should not contain any state in F. One way in which $\mathcal{A}'$ accepts this string is by moving in $\mathcal{A}_1$(upper copy) for first k input symbols. On $k+1$ th input symbol it can make a non-deterministic transition to state in $\mathcal{A}_2$ (note that this transition exists since $\sigma(k+1) \notin F$). The non-final state which was visited infinitely often in $\mathcal{A}$, will also be visited infinitely often in $\mathcal{A}_2$. Since all states in $\mathcal{A}_2$ are accepting α will be accepted.
- For $\alpha \in L(\mathcal{A})$, at least one of the states call it $s \in F$ occurs infinitely often in σ. If $\mathcal{A}'$ does not make a transition to the second copy when running α, α will clearly be not accepted. If it does make the transition then it won't be able to move beyond a certain point (when it won't find transition to state $s \in F$).

From the construction we can see that complemented automaton for DBA with n states will need will require at most $2 * n$ states. Note complemented automaton makes use of non-determinism. Next we look at complementation of Non-Deterministic Büchi Automata (NBA).

Construction for complementation of NBA is nontrivial and involves an interesting combinatorial argument. While the complementation of DBA (with n states) can be done using $2 * n$ states, complementation of an NBA may require up to $O(3^{n^2})$ states. An improved and essentially optimal bound of $2^{O(n \log n)}$ was given by Safra [5]. Since complementation of both DBA and NBA results in NBA in general, we can say that **BA are closed under complementation**.

Theorem 2 [1] *If $L \subseteq A^w$ is Büchi recognizable, so is $A^w - L$. Moreover, from a Büchi automaton recognizing L one can construct one recognizing $A^w - L$.*

The complete proof can be found in [1]. Below I list those parts of the proof that are helpful in explaining an example.

In the proof we try to represent both L and $A^w - L$ as finite union of sets $U.V^w$ where U and V are regular sets of special kind, namely classes of a certain congruence relation over A^* of finite index. A congruence is here an equivalence relation compatible with concatenation. Let $L = L(\mathcal{A})$ where $\mathcal{A} = (Q, q_0, \delta, F)$ is a BA. We write $s \xrightarrow{F}_w s'$ if there is a run of $\mathcal{A}$ on w from state s to s' such that at least one of the states in the run (including s and s') belongs to F. The set

$W^F_{ss'} = \{w \in A^* | s \xrightarrow{F}_w s'\}$ is regular. Now we define the equivalence relation $\sim_\mathcal{A}$ over A^* as follows:

$$u \sim_\mathcal{A} v \text{ iff } \forall s, s' \in Q(s \longrightarrow_u s' \Leftrightarrow s \longrightarrow_v s' \text{ and } s \xrightarrow{F}_u s' \Leftrightarrow s \xrightarrow{F}_v s')$$

The relation $\sim_{\mathcal{A}}$ is a congruence over A^*, which is of finite index by finiteness of Q. To show why $\sim_{\mathcal{A}}$ is of finite index, consider the pair of states $(s, s^{'})$. Clearly number of such pairs is n^2 where n is the no. of states in Q. Now given a finite word u and a pair $(s, s^{'})$ we have three possible (disjoint) cases:

- Case 1: $u \in W^F_{ss'}$ (and hence $u \in W_{ss'}$ too). This also implies that $u \notin A^* - W_{ss'}$ and $u \notin A^* - W^F_{ss'}$
- Case 2: $u \in W_{ss'}$ and $u \notin W^F_{ss'}$. This also implies that $u \notin A^* - W_{ss'}$ and $u \in A^* - W^F_{ss'}$
- Case 3: $u \notin W_{ss'}$ (and hence $u \notin W^F_{ss'}$ too). This also implies that $u \in A^* - W_{ss'}$ and $u \in A^* - W^F_{ss'}$

Let us denote the equivalence class containing u as $[u]$. We can write

$[u] = \bigcap_{(s,s') \in Q \times Q} C_{ss'}$, where

- $C_{ss'} = W^F_{ss'}$, if u satisfies case 1 for this $(s, s^{'})$
- $C_{ss'} = W_{ss'} \cap A^* - W^F_{ss'}$, if u satisfies case 2 for this $(s, s^{'})$
- $C_{ss'} = A^* - W_{ss'}$, if u satisfies case 3 for this $(s, s^{'})$

Note that each $[u]$ is a finite intersection of regular sets and hence is regular itself. Also note that number of equivalence classes is finite. This is because a given pair of states $(s, s^{'})$, we have 3 choices of $C_{ss'}$ and we have n^2 pairs of states, therefore we can have at most 3^{n^2} eq. classes.

A representation of $L(\mathcal{A})$ and $A^w - L(\mathcal{A})$ is terms of $\sim_{\mathcal{A}}$-classes will be provided by the following lemma:

Lemma 1 [1] *(a) Let $\mathcal{A}$ be a BA. For any $\sim_{\mathcal{A}}$-classes U, V: If $U.V^w \cap L(\mathcal{A}) \neq \phi$, then $U.V^w \subseteq L(\mathcal{A})$. (Hence: if $U.V^w \cap (A^w - L(\mathcal{A})) \neq \phi$, then $U.V^w \subseteq A^* - L(\mathcal{A})$).*
(b) Let $\mathcal{A}$ be an arbitrary congruence over A^* of finite index. For any ω-word $\alpha \in A^w$ there are $\sim_{\mathcal{A}}$-classes U, V such that $\alpha \in U.V^w$.

From the lemma 1 it follows that

$L(\mathcal{A}) = \bigcup\{U.V^w | U, V \sim -classes, U.V^w \cap L(\mathcal{A}) \neq \phi\}$

$A^w - L(\mathcal{A}) = \bigcup\{U.V^w | U, V \sim -classes, U.V^w \cap L(\mathcal{A}) = \phi\}$

We now give an example to illustrate the above ideas. The example below constructs the $\sim_{\mathcal{A}}$-classes for the given automaton and then expresses $L(\mathcal{A})$ and $A^w - L(\mathcal{A})$ in terms of these of these classes.

Consider NBA $\mathcal{A} = (Q, q_0, \delta, F)$ on alphabet $A = \{a, b\}$ as in Figure 7. $\mathcal{A}$ accepts ω-words in which a occurs finitely often. Now we can list the sets $W_{ss'}, W^F_{ss'}, A^* - W^F_{ss'}, A^* - W_{ss'}$ for pairs $(s, s^{'}) \in Q \times Q$.

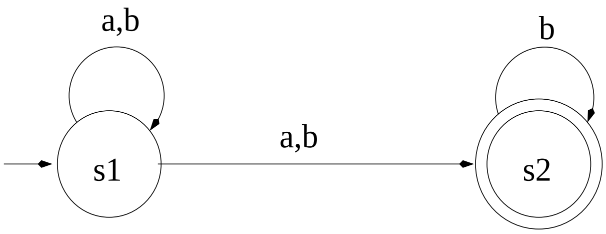

Figure 7: NBA over $\mathcal{A}$ $\{a,b\}$ which accepts ω-words with finite no. of $a's$

$(s,s^{'})$	$W_{ss'}$	$W^F_{ss'}$	$A^* - W_{ss'}$	$A^* - W^F_{ss'}$
(s_1,s_1)	$(a+b)^*$	ϕ	ϕ	$(a+b)^*$
(s_1,s_2)	$(a+b)^+$	$(a+b)^+$	ϵ	ϵ
(s_2,s_1)	ϕ	ϕ	$(a+b)^*$	$(a+b)^*$
(s_2,s_2)	b^*	b^*	$b^*a(a+b)^*$	$b^*a(a+b)^*$

Table of various regular sets for above automaton

Since equivalence class for a u, is given by

$[u] = \bigcap_{(s,s')\in Q\times Q} C_{ss'}$

Using above formula, we can write

1. $[a] = (W_{s_1s_1} \cap A^* - W^F_{s_1s_1}) \cap W^F_{s_1s_2} \cap A^* - W_{s_2s_1} \cap A^* - W_{s_2s_2}$. This intersection turns out to be the regular expression $b^*a(a+b)^*$
2. $[b] = (W_{s_1s_1} \cap A^* - W^F_{s_1s_1}) \cap W^F_{s_1s_2} \cap A^* - W_{s_2s_1} \cap W^F_{s_2s_2}$. This turns out to be the regular expression b^+
3. $[\epsilon] = (W_{s_1s_1} \cap A^* - W^F_{s_1s_1}) \cap A^* - W_{s_1s_2} \cap A^* - W_{s_2s_1} \cap W^F_{s_2s_2}$. This turns out to be the regular expression $\{\epsilon\}$

In this particular example it turns out that these are the only possible equivalence classes of $\sim_{\mathcal{A}}$. From the above equivalence classes we can form following 6 ω-regular expressions of form $U.V^w$. These are as follows $[a][a]^w, [b][a]^w, [\epsilon][a]^w, [b][b]^w, [a][b]^w, [\epsilon][b]^w$. Now we have to obtain $L(\mathcal{A}^c)$ as a finite union of $U.V^w$. For this we go back to Lemma 1 and observe following:

1. $[a][a]^w \cap L(\mathcal{A}) = \phi$, so $[a][a]^w \subseteq A^w - L(\mathcal{A})$
2. $[b][a]^w \cap L(\mathcal{A}) = \phi$, so $[b][a]^w \subseteq A^w - L(\mathcal{A})$
3. $[\epsilon][a]^w \cap L(\mathcal{A}) = \phi$, so $[\epsilon][a]^w \subseteq A^w - L(\mathcal{A})$
4. $[b][b]^w \cap L(\mathcal{A}) \neq \phi$, so $[b][b]^w \subseteq L(\mathcal{A})$
5. $[a][b]^w \cap L(\mathcal{A}) \neq \phi$, so $[a][b]^w \subseteq L(\mathcal{A})$
6. $[\epsilon][b]^w \cap L(\mathcal{A}) \neq \phi$, so $[\epsilon][b]^w \subseteq L(\mathcal{A})$

So finally using lemma 1 and $[a][a]^w, [b][a]^w, [\epsilon][a]^w, [b][b]^w, [a][b]^w, [\epsilon][b]^w$ we can write

$L(\mathcal{A}) = [b][b]^w \cup [a][b]^w \cup [\epsilon][b]^w$

$L(\mathcal{A}^c) = A^w - L(\mathcal{A}) = [a][a]^w \cup [b][a]^w \cup [\epsilon][a]^w$
This was the required representation for $L(\mathcal{A})$ & $L(\mathcal{A}^c)$ as ω-regular expressions.

Now let us consider the complexity of the complementation process. Given an NBA $\mathcal{A}$ with n states we can have at most 3^{n^2} different $\sim_{\mathcal{A}}$-classes. This leads to a size bound of $O(3^{n^2})$ for the complement automaton.

This finishes our discussion on BA and their closure properties. In the next sections we will consider S1S, McNaughton's Theorem and explore there relationships with BA.

4 S1S (Second Order Theory of one successor)

Automata on infinite words were introduced by Büchi. He was mainly interested in decidability of S1S, the monadic second order theory of one successor. A decision procedure for a logic decides, whether a given formula in this logic is valid or satisfiable. A formula is said to be valid if it is true in all possible models and satisfiable if there exists a model in which it is true.
Büchi showed the surprising fact that any condition on sequences written in S1S can be reformulated as a statement about the acceptance of sequences by an automaton.

We start by looking at how to express conditions on ω-sequences in S1S.

An ω-word $\alpha \in A^w$ is represented in the form $\underline{\alpha} = (\omega, 0, +1, <, (Q_a)_{a \in A})$ where $(\omega, 0, +1, <)$ is the structure of natural numbers with zero, successor function, and the usual ordering, and where $Q_a = \{i \in \omega | \alpha(i) = a\}$ for $a \in A$. In other words Q_a is the set of positions in α containing the letter a.

For example say $\alpha = abab.....$, for this α we have $Q_a = (0, 2, 4, ...) and\ Q_b = (1, 3, 5, ...)$.

We now define $\mathbf{S1S_A}$ or the second order theory of one successor over a given alphabet A

- A variable can be one of $x, y....$ It ranges over the set of non-negative numbers. These variables will be used for positions in ω-words.
- A set variable can be $X, Y, ...$ representing the sets of non-negative numbers.
- We have a function symbol $+1$ (successor) that can be applied to first order variables
- *Terms* are constructed from constant 0 and the variables $x, y...$ by application of $+1$ (successor).
- An atomic formula is of the form $t = t^{'}, t < t', t \in X, t \in Q_a$ (for $a \in A$) where $t, t^{'}$ are terms and X is a set variable.

- $S1S_A$ formulas (denote by φ) are the closure of atomic formulas under the usual Boolean connectives and quantification over variables. The meaning of the Boolean connectives and quantification is defined in the usual way.

Example 1: Language $L_1 \subseteq \{a, b, c\}^w$, containing ω-words where after any two consecutive occurrences of letter a, there is a letter b can be expressed as following $S1S_A$ formula:

$$\varphi_1 : \forall x \forall y (x \in Q_a \land y \in Q_a \land y = x + 1 \rightarrow \exists z (z = y + 1 \land z \in Q_b))$$

Example 2: This example illustrates the use of set variables. Language $L_2 \subseteq \{a, b\}^w$, containing ω-words where between any two succeeding occurrences of letter a there are an odd number of occurrences of b, can be expressed as:

$$\varphi_2 : \forall x \forall y (x \in Q_a \land y \in Q_a \land \neg \exists z (z > x \land z < y \land z \in Q_a))$$
$$\rightarrow \exists X (x \in X \land \forall z (z \in X \leftrightarrow \neg (z - 1 \in X)) \land (y \in X)).$$

We write $\varphi(X_1, ...X_n)$ to indicate that the variables $X_1, ..., X_n$ occur free in φ. Formulas without free variables are called sentences. Given $\alpha \in A^w$ and an $S1S_A$ sentence φ, we write $\underline{\alpha} \models \varphi$ if φ is satisfied in $\underline{\alpha}$ (recall $\underline{\alpha} = (\omega, 0, +1, <, (Q_a)_{a \in A})$). The ω-language recognized be an $S1S_A$ sentence φ is $L(\varphi) = \{\alpha \in A^w | \ \underline{\alpha} \models \varphi\}$.

For example, with $\alpha = aabaab...$ and φ_1 of example 1, we have $\underline{\alpha} \models \varphi_1$, and $L(\varphi_1)$ is the ω-language L_1 of the example.

4.1 Büchi's Theorem

Theorem 3 [1] *An ω-language is definable in S1S iff it is regular.*

We know that Büchi automata define the class of regular ω-languages. So the above theorem says that given a S1S sentence we must be able to find an equivalent Büchi automaton for it. And given a Büchi automaton we should be able to get an equivalent S1S sentence. Next we give examples to show how these conversions can be done.

4.1.1 Example for Büchi automaton to S1S sentence

Consider the Büchi automaton $\mathcal{A}$ in the Figure 8, $Q = \{0, 1, 2\}, q_0 = 0, F = \{0, 1\}$. We now need to construct an S1S sentence equivalent to the above automaton $\mathcal{A}$. For this, we use 3 set variables Y_0, Y_1, Y_2, where Y_i contains the set of positions where a run of $\mathcal{A}$ on an ω-word assumes state i. We now have 3 constraints that any successful run of $\mathcal{A}$ must satisfy. These are:

- A position in a given run corresponds to exactly one state. More formally, $\varphi_1 : \neg \exists x (x \in Y_0 \land x \in Y_1) \ \land \ \neg \exists x (x \in Y_1 \land x \in Y_2) \ \land \ \neg \exists x (x \in Y_0 \land x \in Y_2)$
- Transition relation must be obeyed i.e, $\varphi_2 : \forall x ((x \in Y_0 \land x \in Q_a \land x + 1 \in Y_1) \lor (x \in Y_1 \land x \in Q_b \land x + 1 \in Y_2) \lor$ $(x \in Y_2 \land x \in Q_c \land x + 1 \in Y_0) \lor (x \in Y_1 \land x \in Q_b \land x + 1 \in Y_0))$

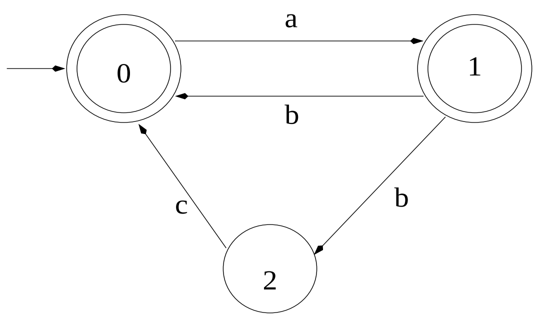

Figure 8: Büchi Automaton on alphabet $A = \{a, b, c\}$

- Büchi Acceptance condition i.e,
  $\varphi_3 : (\forall x \exists y (x < y \wedge y \in Y_0)) \vee (\forall x \exists y (x < y \wedge y \in Y_1))$

If we combine the above 3 constraints we get the required S1S sentence as:
$\varphi : \exists Y_0 \exists Y_1 \exists Y_2 (\varphi_1 \wedge \varphi_2 \wedge \varphi_3)$

4.1.2 S1S sentence to Büchi automaton (BA)

We assume that S1S formulae are interpreted over ω-words over the alphabet $\{0, 1\}^n$. (i.e, the letters of an arbitrary alphabet has been encoded using 0's and 1's). So each position of an ω-word now contains a vector of length n. In order to facilitate conversion of S1S formulae to BA, S1S formulae are first reduced to a simpler formalism $S1S_0$.
We will use X_i as the set of positions whose corresponding vectors have ith component as 1.
In $S1S_0$ only second-order variables X_i occur and atomic formulae are of form $X_i \subseteq X_j$ and $Succ(X_i, X_j)$ ("X_i, X_j are singletons $\{x\}, \{y\}$ where $x + 1 = y$").

A formula in S1S can be systematically reduced to one in $S1S_0$ as shown in [1]. For example, consider the S1S formula $\varphi : \forall x \exists y (x < y \wedge x + 1 = y)$. Steps for reduction to $S1S_0$ are as follows:

- Replace $x < y$ by
  $\varphi_1 : \forall W (x + 1 \in W \wedge \forall z (z \in W \rightarrow z + 1 \in W) \rightarrow (y \in W))$
- So our φ becomes as $\forall x \exists y (\varphi_1 \wedge x + 1 = y)$
- Since we are not allowed to have first order variables. We convert x, y, z to singleton sets $Sing(X), Sing(Y), Sing(Z)$.
- So finally φ becomes
  $\varphi : \forall X (Sing(X) \rightarrow \exists Y (Sing(Y) \wedge Succ(X, Y) \wedge \varphi_2))$
- Where φ_2 is obtained from φ_1 by replacing x, y, z inside it as follows:
  $\varphi_2 : \forall W (Y \subseteq W \wedge \forall Z (Sing(Z) \wedge Z \subseteq W \rightarrow \exists Z_1 (Sing(Z_1) \wedge Succ(Z, Z_1) \wedge Z_1 \subseteq W))$
  $\rightarrow (Y \subseteq W))$
- Note that "Sing X" can be defined in $S1S_0$ as follows
  $\exists Y (Y \subseteq X \wedge Y \neq X \wedge \neg \exists Z (Z \subseteq X \wedge Z \neq X \wedge Z \neq Y))$, where $X \neq Y$ can be expressed as $\neg (X \subseteq Y \wedge Y \subseteq X)$.

Once we have obtained $S1S_0$ equivalent of S1S formula, we can inductively construct BA for $S1S_0$ formula. Formally, it can be shown by induction over $S1S_0$ formulas, that for any $S1S_0$ formula there is a Büchi Automaton $\mathcal{A}$ over $\{0,1\}^n$ with $L(\mathcal{A}) = L(\varphi)$ ([1]).

The next few examples illustrate the conversion from $S1S_0$ formulae to BA:

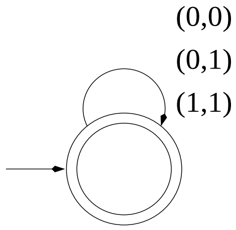

Figure 9: Büchi Automaton for $\varphi : X_1 \subseteq X_2$

Example 1: The Figure 9 shows BA for $X_1 \subseteq X_2$ on alphabet $\{0,1\}^2$. Any ω-word having $(1,0)$ in some position say k is not acceptable because then k will be in X_1 but not in X_2, thus violating $X_1 \subseteq X_2$.

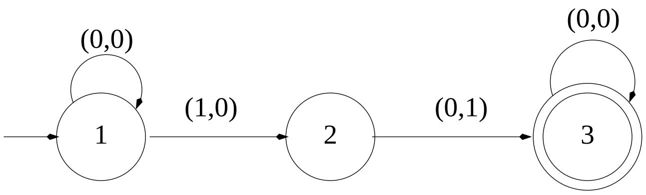

Figure 10: Büchi Automaton for $\varphi : Succ(X_1, X_2)$

Example 2: The Figure 10 shows BA for $Succ(X_1, X_2)$ on alphabet $\{0,1\}^2$. As long as we see (0,0) both X_1, X_2 remain empty. Say (1,0) occurs at position k, then at that position k gets added to X_1. If next input happens to be (0,1) then $k+1$ gets added to X_2. No more 1's should get added since X_1, X_2 are singletons. If all the above are satisfied then our $X_1 = \{k\}$ & $X_2 = \{k+1\}$, so ω-word will be accepted.

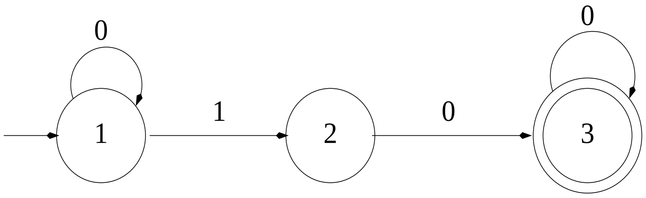

Figure 11: Büchi Automaton for $\varphi : \exists X_2 Succ(X_1, X_2)$

Example 3: The Figure 11 shows BA for $\exists X_2 Succ(X_1, X_2)$ on alphabet $\{0,1\}^2$. We have removed the component corresponding to X_2, which is the 2nd element of the input vector. A successful run of the resulting automaton "guesses" a second component for the given input

$\alpha \in \{0,1\}^w$ and for the resulting sequence from $\alpha \in (\{0,1\}^2)^w$ it is a successful run of the automaton for $Succ(X_1, X_2)$.

Example 4: For the formula $\varphi : Succ(X_1, X_2) \vee Succ(X_2, X_3)$. We obtain BA for $Succ(X_1, X_2)$ and $Succ(X_2, X_3)$ and take there union. Since BA are closed under all boolean properties this can be done for other boolean connectives too. Figure 12 shows the resulting BA over alphabet $\{0,1\}^3$.

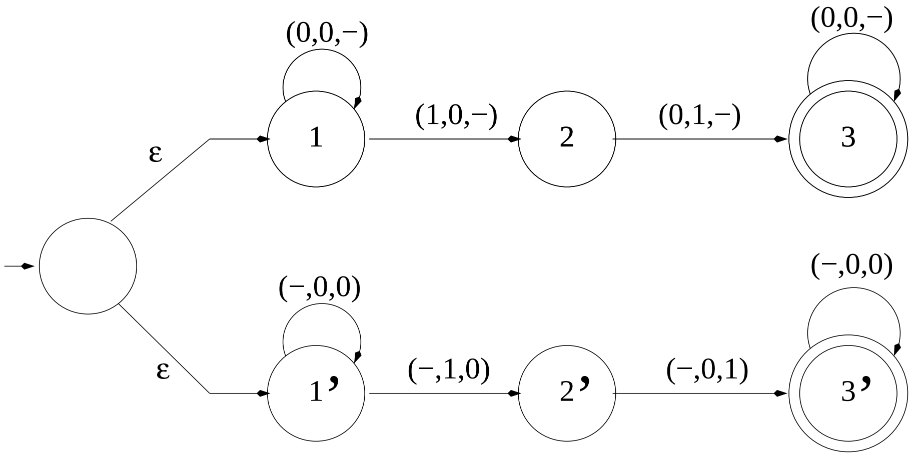

Figure 12: $\varphi : Succ(X_1, X_2) \vee Succ(X_2, X_3)$, - represents don't care

Example 5: $\varphi : \exists X_1 \exists X_2 \neg(X_1 \subseteq X_2)$. Here we use the construction to complement the DBA (two copies are used). Figure 13 shows the result.

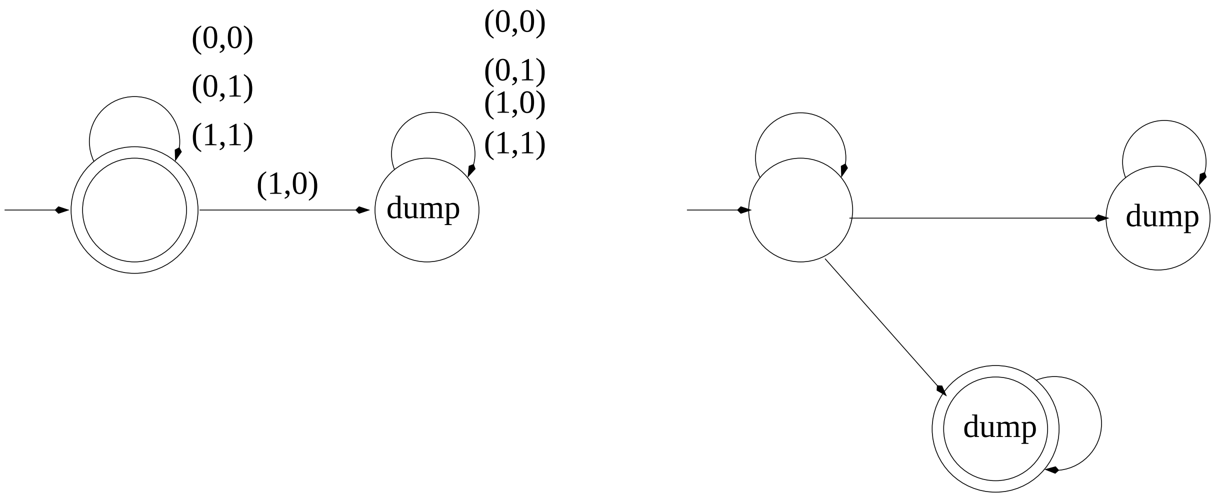

Figure 13: Figure on left is BA for $X_1 \subseteq X_2$, right has BA for $\varphi : \exists X_1 \exists X_2 \neg(X_1 \subseteq X_2)$

Theorem 4 [1] *Truth of sentences of S1S is decidable.*

We know that a sentence is a formula with no free variables. The resulting automaton (on S1S to BA conversion) for a sentence has unlabelled transitions and in this sense is "input-free" (as in Figure 13). However it does have structure and set of accepting states. This S1S sentence holds over $(\omega, 0, +1, <)$ iff the resulting "input-free" automata $\mathcal{A}$ has some successful run. The later is decidable because a successful run exists iff there is some state q in $\mathcal{A}$ which is reachable from the initial state and such that q is reachable from q via a nonempty path.

Example 1: Consider the sentence $\varphi : \exists X_1 \exists X_2 \neg(X_1 \subseteq X_2)$. Its BA is given in Figure 13. (right one). Note that it has unlabelled transitions. Also note that there is a reachable accepting state ("dump") and that can be visited in a loop. So φ is valid over domain of natural numbers.

Example 2: Consider the sentence $\varphi : \exists X_1 \exists X_2 (Succ(X_1, X_2) \wedge Succ(X_2, X_1))$. Intuitively we know that this formula is unsatisfiable. BA for $Succ(X_1, X_2)$ and $Succ(X_2, X_1)$ is given in Figure 14.
We use the construction for Büchi intersection to get BA for $(Succ(X_1, X_2) \wedge Succ(X_2, X_1))$. The resulting automaton is shown in Figure 15. Note that there's no final state reachable from start state and which can be visited infinitely often. So there does not exist any sequence which satisfies this formula.

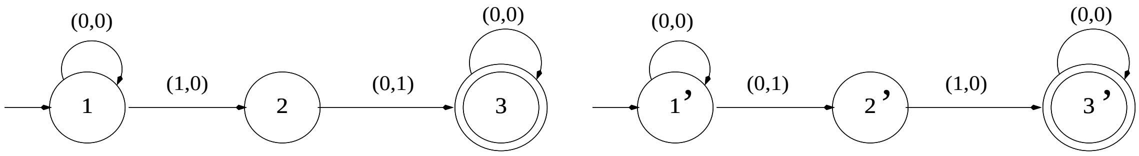

Figure 14: BA for $\mathcal{A}_1$ & $\mathcal{A}_2$ for $Succ(X_1, X_2)$ & $Succ(X_2, X_1)$

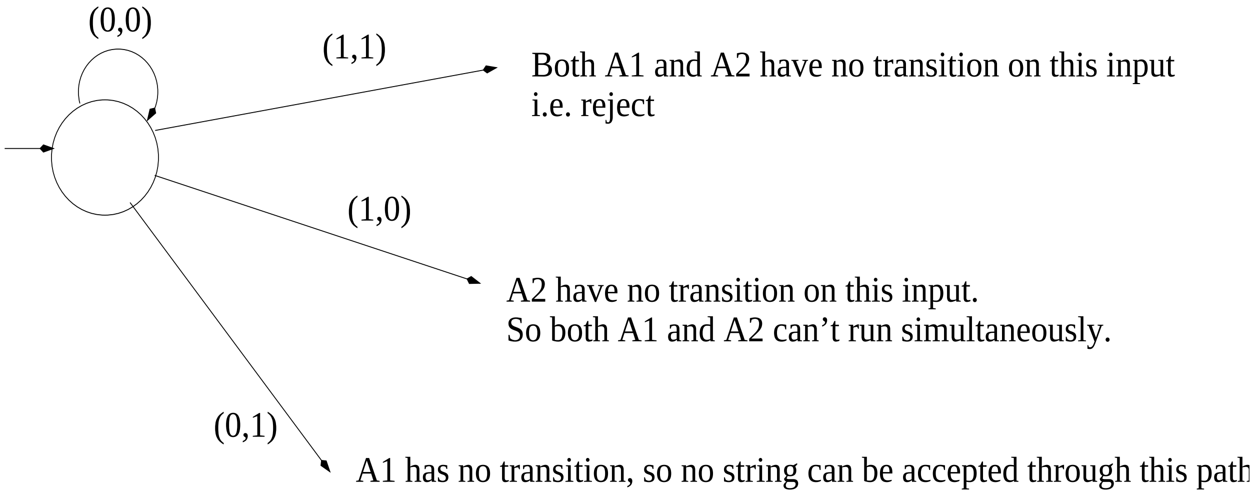

Figure 15: BA for $(Succ(X_1, X_2) \wedge Succ(X_2, X_1))$ i.e, intersection of $\mathcal{A}_1$ & $\mathcal{A}_2$ above

5 Determinism and McNaughton's Theorem

We will show by an example that Deterministic Büchi automaton (DBA) are not closed under complementation. That is we might not get a DBA on complementing another DBA. However, it turns out that by refining the notion of acceptance we can have deterministic automaton which characterize regular ω-languages. Muller Automaton is one such formalism (whose deterministic version is equivalent in expressiveness to Non Deterministic Büchi automaton (NBA)).

If a DFA $\mathcal{A}$, recognizing the set $W \subseteq A^*$ is used as a Büchi automaton instead, it accepts an ω-word α iff infinitely many prefixes of α lead $\mathcal{A}$ to a final state, i.e belong to W. Collecting these α we obtain the set

$$\vec{W} = \{\alpha \in A^w \mid \exists^w n \; \alpha(0, n) \in W\}$$

By the above mentioned conversion of DFA to DBA and vice versa, we can say that an ω-language $L \subseteq A^w$ is recognized by a DBA iff $L = \vec{W}$ for some regular $W \subseteq A^*$.

5.1 Expressive power of DBA's

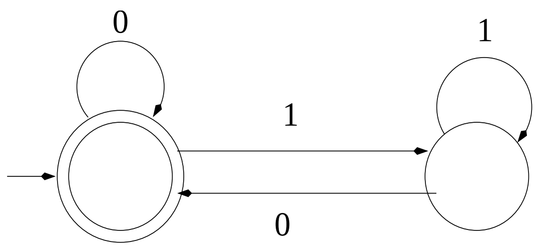

Fig 3

Figure 16: DBA recognizing $L = L(\mathcal{A}) = ((0+1)^*0)^w$

The DBA in Figure 16 accepts the language in which 0 occurs infinitely often. i.e, $L(\mathcal{A}) = ((0+1)^*0)^w$. We can show that the complement of the above language $L^c = (0+1)^*1^w$, cannot be expressed using a DBA. However, L^c can easily be expressed using a non deterministic Büchi automaton given in Figure 17:

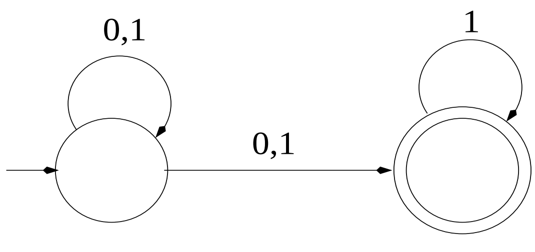

Fig 4

Figure 17: NBA recognizing $L^c = (0+1)^*1^w$

This shows that Non-deterministic Büchi automata (NBA) are more expressive that Deterministic Büchi automata. This also shows that closure under complementation fails for DBA. However Deterministic Muller automata(DMA), which is defined next capture the power of NBA's.

5.2 Deterministic Muller automata (DMA)

Definition: A *Deterministic Muller automaton* (DMA) over an alphabet A is of form $\mathcal{A} = (Q, q_0, \delta, \mathcal{F})$ with finite state set Q, starting state $q_0 \in Q$, transition function as $\delta : Q \times A \rightarrow Q$ and $\mathcal{F} \subseteq 2^Q$ as the set of final states. A run of σ of $\mathcal{A}$ is said to be successful if $In(\sigma) \in \mathcal{F}$ where $In(\sigma)$ is the set of states that occur infinitely many times in σ. $\mathcal{A}$ accepts an ω-word α iff there is a unique run σ of $\mathcal{A}$ on α that is successful. As an example consider a DMA in Figure 18:

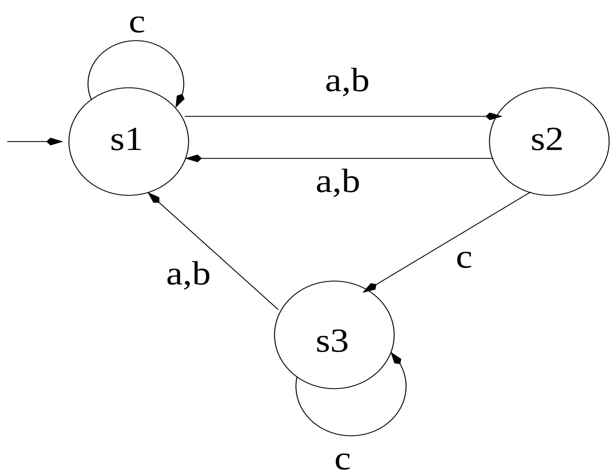

Figure 18: A Deterministic Muller automata (DMA)

Let us suppose $\mathcal{F} = \{\{s_1\}, \{s_1, s_2, s_3\}\}$ for the DMA in Figure 18. For $\alpha = acbacb...$ the Infinity set is $\{s_1, s_2, s_3\} \in \mathcal{F}$. So α is Muller acceptable by this Muller automaton. For $\alpha = ababab...$ the Infinity set is $\{s_1, s_2\} \notin \mathcal{F}$. So α is not Muller acceptable by this automaton.

An ω-language $L \subseteq A^w$ is called *Muller recognizable* iff it consists of all ω-words accepted by some DMA.

5.3 Non-deterministic Muller automata (NMA)

An NMA is defined in the same way as a DMA with δ replaced by $\delta \subseteq Q \times A \times Q$. Also, acceptance of α by an NMA means existence of a run σ on α with $In(\sigma) \in \mathcal{F}$.

We can show that NMA's equal NBA's in expressiveness as follows:

5.3.1 NMA to S1S

This conversion is similar to the one we did for BA to S1S. We will illustrate this by an example, consider the NMA $\mathcal{A}$ in the Figure 19:

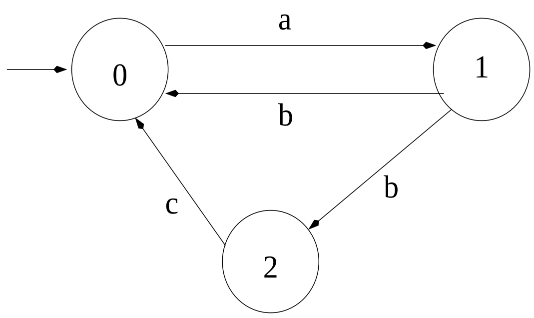

Figure 19: NMA on alphabet $A = \{a, b, c\}$

For the NMA $\mathcal{A}$ in Figure 19, $Q = \{0, 1, 2\}, q_0 = 0$. Let us suppose $\mathcal{F} = \{\{0\}, \{1, 2\}\}$. We now need an S1S sentence that evaluates to true only for ω-words accepted by $\mathcal{A}$. As before we use 3 set variables Y_0, Y_1, Y_2, where Y_i contains set of positions where run of $\mathcal{A}$ on an ω-word assumes state i. We now have 3 types of constraint that any successful run of $\mathcal{A}$ must satisfy. These are:

- A position in a given run corresponds to exactly one state. Expressed as
  $\varphi_1 : \neg\exists x(x \in Y_0 \wedge x \in Y_1) \ \wedge \ \neg\exists x(x \in Y_1 \wedge x \in Y_2) \ \wedge \ \neg\exists x(x \in Y_0 \wedge x \in Y_2)$
- Transition relation must be obeyed i.e,
  $\varphi_2 : \forall x((x \in Y_0 \wedge x \in Q_a \wedge x + 1 \in Y_1) \vee (x \in Y_1 \wedge x \in Q_b \wedge x + 1 \in Y_2) \vee$
  $(x \in Y_2 \wedge x \in Q_c \wedge x + 1 \in Y_0) \vee (x \in Y_1 \wedge x \in Q_b \wedge x + 1 \in Y_0))$
- Muller acceptance by *Infinity set* $= \{0\}$,
  We have to take care that only state 0 occurs infinitely often, so we write
  $\varphi_3 : ((\forall x\exists y(x < y \wedge y \in Y_0)) \wedge \neg(\forall x\exists y(x < y \wedge y \in Y_1)) \wedge \neg(\forall x\exists y(x < y \wedge y \in Y_2)))$
- Muller acceptance by *Infinity set* $= \{1, 2\}$,
  $\varphi_4 : (\neg(\forall x\exists y(x < y \wedge y \in Y_0)) \wedge (\forall x\exists y(x < y \wedge y \in Y_1)) \wedge (\forall x\exists y(x < y \wedge y \in Y_2)))$

If we combine the above 3 constraints we get the required S1S sentence as:
$\varphi : \exists Y_0 \exists Y_1 \exists Y_2(\varphi_1 \wedge \varphi_2 \wedge (\varphi_3 \vee \varphi_4))$

The above construction can be easily generalized to other NMA's. If we use $L(NMA)$ to denote the set of languages which can be expressed using NMA's and $L(S1S)$ to denote the set of languages that can be expressed in S1S, it follows that:

Result 1: $L(NMA) \subseteq L(S1S)$

To show that $L(S1S) \subseteq L(NMA)$, we now show that NBA's (which are as expressive as S1S) can be converted to NMA's

5.3.2 NBA to NMA

Let $\mathcal{A}_B = (Q, q_0, \delta, F)$ be an NBA. We can define the NMA with the same language as $\mathcal{A}_M = (Q, q_0, \delta, \mathcal{F})$, where $\mathcal{F}$ contains subsets of Q having non-empty intersection with F. We can easily prove that $L(\mathcal{A}_B) = L(\mathcal{A}_M)$.

If we use $L(NBA)$ to denote the set of languages which can be expressed using NBA, it follows that:

Result 2: $L(NBA) \subseteq L(NMA)$

From Result 1 and 2 and Büchi's Theorem (4.1) we can conclude that

$$\boxed{L(NBA) \equiv L(NMA) \equiv L(S1S)}$$

5.3.3 DBA to DMA

To convert a DBA to a DMA we use the same construction as that for NBA to NMA. So we have $L(DBA) \subseteq L(DMA)$.

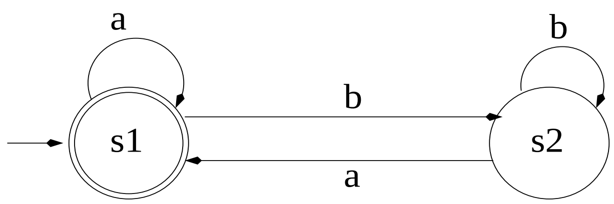

Figure 20: DBA which accepts ω-words with infinitely many $a's$

For the DBA in Figure 20, the equivalent DMA will have $\mathcal{F}$ as $\{\{s_1\}, \{s_1, s_2\}\}$.

5.4 Closure properties of DMA's

DMA's are closed under all boolean operations. Let $\mathcal{A} = (Q, q_0, \delta, \mathcal{F})$ respectively $\mathcal{A}^{'} = (Q^{'}, {q_0}^{'}, \delta^{'}, \mathcal{F}^{'})$ be two DMA's recognizing $L \subseteq A^w$ respectively $L^{'} \subseteq A^w$.

- Complementation: $(Q, q_0, \delta, 2^Q - \mathcal{F})$ recognizes $A^w - L(\mathcal{A})$.
- Intersection: The intersection automaton is simply the product automaton with the set of final states as $\mathcal{F} \times \mathcal{F}^{'}$. Here we define infinity set for a run on α to be a two tuple, the first (respectively second) component will have set of states that occur infinitely many times on run of $\mathcal{A}$ respectively $\mathcal{A}^{'}$ on α.
- Union: The union automaton is the product automaton where the collection of final state sets contain $\{(q_1, q_1^{'}), ..., (q_n, q_n^{'})\}$. iff $\{q_1, .., q_n\} \in \mathcal{F}$ or $\{q_1^{'}, .., q_n^{'}\} \in \mathcal{F}^{'}$.

Lemma 2 [1] *An ω-language $L \subseteq A^w$ is Muller recognizable iff L is boolean combination(over A^w) of sets $\vec{W}$ with regular $W \subseteq A^*$.*

To illustrate the above result consider the following example of an DMA converted to above mentioned union:
Let $\mathcal{F} = \{\{s_1\}, \{s_1, s_2, s_3\}\}$ for DMA $\mathcal{A}$ in Figure 21
An ω-word α is accepted by MA $\mathcal{A}$ iff for run of $\mathcal{A}$ on α, Infinity Set equals exactly $\{s_1\}$ which

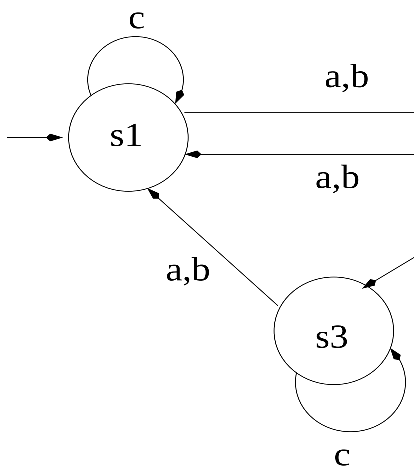

Figure 21: DMA $\mathcal{A}$

is expressed as:

$\alpha \in \overrightarrow{W}_{s_1} \cap (\sim \overrightarrow{W}_{s_2}) \cap (\sim \overrightarrow{W}_{s_3})$ or,

The Infinity set equals exactly $\{s_1, s_2, s_3\}$ which is expressed as:

$\alpha \in \overrightarrow{W}_{s_1} \cap \overrightarrow{W}_{s_2} \cap \overrightarrow{W}_{s_3}$.

So $L(\mathcal{A}) = (\overrightarrow{W}_{s_1} \cap (\sim \overrightarrow{W}_{s_2}) \cap (\sim \overrightarrow{W}_{s_3})) \cup (\overrightarrow{W}_{s_1} \cap \overrightarrow{W}_{s_2} \cap \overrightarrow{W}_{s_3})$.

5.5 McNaughton's Theorem

Theorem 5 [1] *An ω-language is regular (i.e, Büchi recognizable) iff it is Deterministic Muller recognizable.*

The proof in direction from right to left follows from above lemma and closure of Büchi recognizable languages under the boolean operation. For the converse, the following theorem is necessary.

Theorem 6 [1] *Every Büchi recognizable ω-language $L \subseteq A^w$ is a finite union of sets of form $\vec{W_i} \cap \sim \vec{W_j}$ where $\vec{W_i}$, $\vec{W_j} \subseteq A^*$ are regular.*

From McNaughton's Theorem and previous results we have

$$\boxed{L(DBA) \subset L(NBA) \equiv L(DMA) \equiv L(NMA) \equiv L(S1S)}$$

5.6 Rabin Automaton (RA)

It is a tuple $\mathcal{A} = (Q, q_0, \delta, \Omega)$, where Q, q_0, δ are the same as that in Muller automata, and the acceptance criterion is given by $\Omega = \{(L_1, U_1), ..., (L_n, U_n)\}$ where (L_i, U_i) with $L_i, U_i \subseteq Q$ are called accepting pairs.

A run σ of $\mathcal{A}$ is successful iff for some $i \in \{1, 2.., n\}$ we have, $In(\sigma) \cap L_i = \phi$ and $In(\sigma) \cap U_i \neq \phi$.

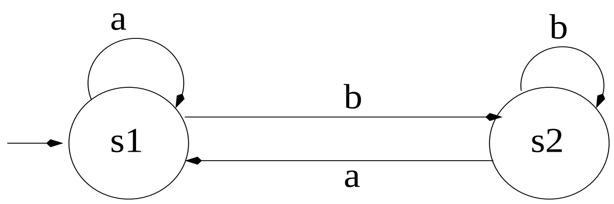

Figure 22: RA with $\Omega = \{(\{s_1\}, \{s_2\}), (\{s_2\}, \{s_1\})\}$

Example: Consider the RA in Figure 22 which accepts ω-words in which one of the a or b occurs finitely often.
The expressiveness of BA and RA can be shown to be equal from McNaughton's Theorem. We end our discussion on RA here. In the next section we give the conclusion of the report.

6 Conclusion

In this report we had surveyed some of the existing formalisms for automata on infinite objects. We began with Büchi Automata (BA), which was introduced by Büchi. We noted that BA are closed under all boolean operations. We looked at the constructions for complementation of DBA's and NBA's. We also looked upon how to specify properties of sequences in monadic second order theory of one successor (S1S). This was followed by Büchi's theorem, which established the equivalence between S1S and BA. We looked upon some of the constructions for BA to S1S and S1S to BA which are used in the proof of Büchi's theorem. Conversion of an S1S sentence to an equivalent BA gives us a way of establishing the truth of that sentence. This involves detecting a reachable final state (in the obtained BA) which can be visited infinitely often. Then we showed that the expressive power of DBA's is strictly less than that of NBA's. We discussed Muller automaton and its boolean properties. Equivalence between the expressive power of NMA, NBA and S1S was established. Expressive power of DMA's and NBA's was shown to be equal using fundamental McNaughton's Theorem.

Constructions for intersection and complementation of BA, BA to S1S, S1S to BA are quite interesting. We have seen that BA, MA and RA differ only in there acceptance criterion. Differences in the accepting criterion bring about significant differnces among these formalisms, for instance DMA's are as expressive as NMA's, while this is not true for DBA's. However, the equivalence between the expressive power of these apparently different formalisms like BA, S1S, MA established through Büchi's theorem and McNaughton's theorem is quite fascinating in itself.

Complexity of union and intersection for BA is of the order of $|Q_1| + |Q_2|$ and $|Q_1| * |Q_2|$ respectively where Q_1, Q_2 represents the state sets of two BA. The complementation of a NBA may require exponential (in n^2, where n is no. of states in NBA) number of states. This clearly puts a limit on practical usage of algorithms that involve complementing NBA's. On the other hand complementation of Muller automata amounts to complementing the set of final states. However, its complexity is also exponential in number of states in MA (because of 2^Q involved). Complexity of union and intersection of Muller automata is $|Q_1| * |Q_2|$. Complementation for

NBA's and Determinization for NMA's thus turn out to be computationally expensive problems.

References

[1] Thomas, W., Automata on infinite objects, *Handbook of Theoretical Computer Science*, Elsevier Science Publishers, 1990, pp. 135–191.

[2] Vardi, M. Y., An automata theoretic approach to linear time temporal logic, 1994.

[3] Büchi, J. R., On a decision method in restricted second order arithmentic, *Proc. 1960 International Congerss on Logic, Methodology, and Philosophy of Science*, 1962, pp. 1-11.

[4] Muller, D. E., Infinite sequences and finite machines, *Proc. 4th Annual IEEE Symposium on Switching Circuit Theory and Logical Design*, 1963, pp. 3-16.

[5] Safra, S., On the complexity of w-automata, *Proc. 29th Annual IEEE Symposium on Found−ations of Computer Science*, 1988, pp. 319-327.

[6] Emerson, E. A., The Role of Buchi's Automata in Computing Science, http://www.cs.utexas.edu/users/emerson/pubs/buchi1h.ps