ω-Automata
ω-Automata

• Automata that accept (or reject) words of infinite length.

• Languages of infinite words appear:
  – in verification, as encodings of non-terminating executions of a program.
  – in arithmetic, as encodings of sets of real numbers.
ω-Languages

• An ω-word is an infinite sequence of letters.
• The set of all ω-words is denoted by $\Sigma^\omega$.
• An ω-language is a set of ω-words, i.e., a subset of $\Sigma^\omega$.
• A language $L_1$ can be concatenated with an ω-language $L_2$ to yield the ω-language $L_1L_2$, but two ω-languages cannot be concatenated.
• The ω-iteration of a language $L \subseteq \Sigma^*$, denoted by $L^\omega$, is an ω-language.
• Observe: $\emptyset^\omega = \emptyset$. 
ω-Regular Expressions

• ω-regular expressions have syntax

\[ s ::= r^ω \mid rs_1 \mid s_1 + s_2 \]

where \( r \) is an (ordinary) regular expression.

• The ω-language \( L_ω(s) \) of an ω-regular expression \( s \) is inductively defined by

\[ L_ω(r^ω) = (L(r))^ω \quad L_ω(rs_1) = L(r)L_ω(s_1) \]
\[ L_ω(s_1 + s_2) = L_ω(s_1) \cup L_ω(s_2) \]

• A language is ω-regular if it is the language of some ω-regular expression.
Büchi Automata

• Invented by J.R. Büchi, swiss logician.
Büchi Automata

• Same syntax as DFAs and NFAs, but different acceptance condition.

• A run of a Büchi automaton on an $\omega$-word is an infinite sequence of states and transitions.

• A run is accepting if it visits the set of final states infinitely often.
  – Final states renamed to accepting states.

• A DBA or NBA $A$ accepts an $\omega$-word if it has an accepting run on it; the $\omega$-language $L_\omega(A)$ of $A$ is the set of $\omega$-words it accepts.
Some examples
From $\omega$-Regular Expressions to NBAs
From NBAs to $\omega$-Regular Expressions
From NBAs to $\omega$-Regular Expressions
From $\omega$-Regular Expressions to NBAs

• **Lemma**: Let $A$ be a NFA, and let $q, q'$ be states of $A$. The language $L_{q'}^q$ of words with runs leading from $q$ to $q'$ and visiting $q'$ exactly once is regular.

• Let $r_{q}^{q'}$ denote a regular expression for $L_{q'}^q$. 
From $\omega$-Regular Expressions to NBAs

• Example:

\[ r_0^1 = (a + b + c)^*(b + c) \]
\[ r_0^2 = (a + b + c)^*b \]
\[ r_1^1 = (b + c)^* \]
\[ r_2^1 = b + (a + c)(a + b + c)^*b \]
From $\omega$-Regular Expressions to NBAs

- Given a NBA $A$, we look at it as a NFA, and compute regular expressions $r^q_{q'}$.
- We show:

$$L_\omega(A) = L(\sum_{q \in F} r^q_{q_0} (r^q_q)^\omega)$$

- An $\omega$-word belongs to $L_\omega(A)$ iff it is accepted by a run that starts at $q_0$ and visits some accepting state $q$ infinitely often.
From $\omega$-Regular Expressions to NBAs

• Example:

\[ L_\omega(A) = r_0^1 (r_1^1)^\omega + r_0^2 (r_2^2)^\omega \]
DBAs are less expressive than NBAs

- **Prop.:** The $\omega$-language $(a + b)^*b^\omega$ is not recognized by any DBA.

- **Proof:** By contradiction. Assume some DBA recognizes $(a + b)^*b^\omega$.
  
  - DBA accepts $b^\omega$ \rightarrow DFA accepts $b^{n_0}$
  - DBA accepts $b^{n_0}a b^\omega$ \rightarrow DFA accepts $b^{n_0}a b^{n_1}$
  - DBA accepts $b^{n_0}a b^{n_1}ab^\omega$ \rightarrow DFA accepts $b^{n_0}a b^{n_1}a b^{n_2}$ etc.
  - By determinism, the DBA accepts $b^{n_0}a b^{n_1}a b^{n_2} ... a b^{n_i} ...$, which does not belong to $(a + b)^*b^\omega$. 
Generalized Büchi Automata

• Same power as Büchi automata, but more adequate for some constructions.
• Several sets of accepting states.
• A run is accepting if it visits each set of accepting states infinitely often.
From NGAs to NBAs

• Important fact:

All the sets $F_1, \ldots, F_n$ are visited infinitely often is equivalent to

$F_1$ is eventually visited and

every visit to $F_i$ is eventually followed by a visit to $F_i \oplus 1$
From NGAs to NBAs

NGA with 3 sets of accepting states

Equivalent NBA with 3 copies of the NGA
\textbf{NGAtoNBA}(A)

\textbf{Input:} NGA $A = (Q, \Sigma, q_0, \delta, \mathcal{F})$, where \(\mathcal{F} = \{F_1, \ldots, F_m\}\)

\textbf{Output:} NBA $A' = (Q', \Sigma, \delta', q'_0, F')$

1. \(Q', \delta', F' \leftarrow \emptyset; q'_0 \leftarrow [q_0, 0]\)
2. \(W \leftarrow \{[q_0, 0]\}\)
3. \textbf{while} \(W \neq \emptyset\) \textbf{do}
   4. \textbf{pick} \([q, i]\) \textbf{from} \(W\)
   5. \textbf{add} \([q, i]\) \textbf{to} \(Q'\)
   6. \textbf{if} \(q \in F_0 \text{ and } i = 0\) \textbf{then add} \([q, i]\) \textbf{to} \(F'\)
   7. \textbf{for all} \(a \in \Sigma, q' \in \delta(q, a)\) \textbf{do}
      8. \textbf{if} \(q \notin F_i\) \textbf{then}
         9. \textbf{if} \([q', i] \notin Q'\) \textbf{then add} \([q', i]\) \textbf{to} \(W\)
         10. \textbf{add} \(([q, i], a, [q', i])\) \textbf{to} \(\delta'\)
      11. \textbf{else} /* \(q \in F_i\) */
         12. \textbf{if} \([q', i \oplus 1] \notin Q'\) \textbf{then add} \([q', i \oplus 1]\) \textbf{to} \(W\)
         13. \textbf{add} \(([q, i], a, [q', i \oplus 1])\) \textbf{to} \(\delta'\)
   14. \textbf{return} \((Q', \Sigma, \delta', q'_0, F')\)
\[ \mathcal{F} = \{ \{q\}, \{r\} \} \]
DGAs have the same expressive power as DBAs, and so are not equivalent to NGAs.

- **Question:** Are there other classes of omega-automata with
  - the same expressive power as NBAs or NGAs, and
  - with equivalent deterministic and nondeterministic versions?

We are only willing to change the acceptance condition!
Co-Büchi automata

• A **nondeterministic co-Büchi automaton (NCA)** is syntactically identical to a NBA, but a run is accepting iff it only visits accepting states finitely often.
Which are the languages?
Determinizing co-Büchi automata

• Given a NCA $A$ we construct a DCA $B$ such that $L(A) = L(B)$.

• We proceed in three steps:
  – We assign to every $\omega$-word $w$ a directed acyclic graph $\text{dag}(w)$ that ``contains´´ all runs of $A$ on $w$.
  – We prove that $w$ is accepted by $A$ iff $\text{dag}(w)$ is infinite but contains only finitely many breakpoints.
  – We construct a DCA $B$ that accepts an $\omega$-word $w$ iff $\text{dag}(w)$ is infinite and contains finitely many breakpoints.
• Running example:
$\text{dag}(aba^\omega)$

$\text{dag}((ab)^\omega)$
• $A$ accepts $w$ iff some infinite path of $\text{dag}(w)$ only visits accepting states finitely often
Levels of a dag

Level 0  Level 1  Level 2  Level 3  Level 4
Breakpoints of a *dag*

- We defined inductively the set of levels that are breakpoints:
  - Level 0 is always a breakpoint
  - If level $l$ is a breakpoint, then the next level $l'$ such that *every path* between $l$ and $l'$ visits an accepting state is also a breakpoint.
Only two breakpoints

Infinitely many breakpoints
• **Lemma**: \( A \) accepts \( w \) iff \( \text{dag}(w) \) is infinite and has only finitely many breakpoints.

**Proof:**

If \( A \) accepts \( w \), then \( A \) has at least one run on \( w \), and so \( \text{dag}(w) \) is infinite. Moreover, the run visits accepting states only finitely often, and so after it stops visiting accepting states there are no further breakpoints.

If \( \text{dag}(w) \) is infinite, then it has an infinite path, and so \( A \) has at least one run on \( w \). Since \( \text{dag}(w) \) has finitely many breakpoints, then every infinite path visits accepting states only finitely often.
Constructing the DCA

• If we could tell if a level is a breakpoint by looking at it, we could take the set of breakpoints as states of the DCA.
• However, we also need some information about its `history`.
• Solution: add that information to the level!
• States: pairs \([P, O]\) where:
  – \(P\) is the set of states of a level, and
  – \(O \subseteq P\) is the set of states `that owe a visit to the accepting states`. Formally: \(q \in O\) if \(q\) is the
Constructing the DCA

- States: pairs \([P, O]\) where:
  - \(P\) is the set of states of a level, and
  - \(O \subseteq P\) is the set of states \"that owe a visit\" to the accepting states.

- Formally: \(q \in O\) if \(q\) is the endpoint of a path starting at the last breakpoint that has not yet visited any accepting state.
Constructing the DCA

- **States**: pairs \([P, O]\)
- **Initial state**: pair \([[\{q_0\}, \emptyset]]\) if \(q_0 \in F\), and \([[\{q_0\}, \{q_0\}]\) otherwise.
- **Transitions**: \(\delta([P, Q], a) = [P', O']\) where \(P' = \delta(P, a)\), and
  - \(O' = \delta(O, a) \setminus F\) if \(O \neq \emptyset\)
  (automaton updates set of owing states)
  - \(O' = \delta(P, a) \setminus F\) if \(O = \emptyset\)
  (automaton starts search for next breakpoint)
- **Accepting states**: pairs \([P, \emptyset]\) (no owing states)
NCAtoDCA(A)

Input: NCA A = (Q, Σ, δ, q₀, F)

Output: DCA B = (Q̃, Σ, ̃δ, ̃q₀, ̃F) with \( L_\omega(A) = \overline{B} \)

1. \( ̃Q, ̃δ, ̃F \leftarrow \emptyset; \) if \( q₀ \in F \) then \( ̃q₀ \leftarrow [q₀, \emptyset] \) else \( ̃q₀ \leftarrow [{q₀}, {q₀}] \)
2. \( W \leftarrow \{ ̃q₀ \} \)
3. while \( W \neq \emptyset \) do
4.     pick \([P, O]\) from \( W \); add \([P, O]\) to \( ̃Q \)
5.     if \( P = \emptyset \) then add \([P, O]\) to \( ̃F \)
6.     for all \( a \in Σ \) do
7.         \( P' = ̃δ(P, a) \)
8.         if \( O \neq \emptyset \) then \( O' \leftarrow ̃δ(O, a) \setminus F \) else \( O' \leftarrow ̃δ(P, a) \setminus F \)
9.         add \(([P, O], a, [P', O'])\) to \( ̃δ \)
10.    if \([P', O'] \notin ̃Q\) then add \([P', Q']\) to \( W \)

- **Complexity**: at most \( 3^n \) states
Running example
Recall ...

• **Question:** Are there other classes of omega-automata with
  – the same expressive power as NBAs or NGAs, and
  – with equivalent deterministic and nondeterministic versions?

Are co-Büchi automata a positive answer?
Unfortunately no ...

• **Lemma**: No DCA recognizes the language $\left( b^* a \right)^\omega$.

**Proof**: Assume the contrary. Then the same automaton seen as a DBA recognizes the complement $\left( a + b \right)^* b^\omega$. Contradiction.

So the quest goes on ...
Muller automata

• A nondeterministic Muller automaton (NMA) has a collection \( \{F_0, F_1, \ldots, F_{m-1}\} \) of sets of accepting states.

• A run is accepting if the set of states it visits infinitely often is equal to one of the sets in the collection.
From Büchi to Muller automata

• Let $A$ be a NBA with set $F$ of accepting states.
• A set of states of $A$ is good if it contains some state of $F$.
• Let $G$ be the set of all good sets of $A$.
• Let $A'$ be "the same automaton" as $A$, but with Muller condition $G$.
• Let $\rho$ be an arbitrary run of $A$ and $A'$. We have
  
  $\rho$ is accepting in $A$

  iff $\inf(\rho)$ contains some state of $F$

  iff $\inf(\rho)$ is a good set of $A$

  iff $\rho$ is accepting in $A'$
From Muller to Büchi automata

- Let $A$ be a NMA with condition $\{F_0, F_1, \ldots, F_{m-1}\}$.
- Let $A_0, \ldots, A_{m-1}$ be NMAs with the same structure as $A$ but Muller conditions $\{F_0\}, \{F_1\}, \ldots, \{F_{m-1}\}$ respectively.
- We have: $L(A) = L(A_0) \cup \ldots \cup L(A_{m-1})$
- We proceed in two steps:
  1. we construct for each NMA $A_i$ an NGA $A_i'$ such that $L(A_i) = L(A_i')$
  2. we construct an NGA $A'$ such that $L(A') = L(A'_0) \cup \ldots \cup L(A'_{m-1})$
Transitions leaving $F_i$ are duplicated and resent to the copy of $F_i$.

NGA with accepting condition \[
\{ \{ q'_1 \}, \ldots, \{ q'_m \} \}\]
NMA1 to NGA

**Input:** NMA $A = (Q, \Sigma, q_0, \delta, \{F\})$

**Output:** NGA $A = (Q', \Sigma, q'_0, \delta', \mathcal{F}')$

1. $Q', \delta', \mathcal{F'} \leftarrow \emptyset$
2. $q'_0 \leftarrow [q_0, 0]$
3. $W \leftarrow \{(q_0, 0)\}$
4. while $W \neq \emptyset$ do
5.     pick $[q, i]$ from $W$; add $[q, i]$ to $Q'$
6.     if $q \in F$ and $i = 1$ then add $\{[q, 1]\}$ to $\mathcal{F}'$
7.     for all $a \in \Sigma$, $q' \in \delta(q, a)$ do
8.         if $i = 0$ then
9.             add $([q, 0], a, [q', 0])$ to $\delta'$
10.        if $[q', 0] \notin Q'$ then add $[q', 0]$ to $W$
11.        if $q' \in F$ then
12.            add $([q, 0], a, [q', 1])$ to $\delta'$
13.            if $[q', 1] \notin Q'$ then add $[q', 1]$ to $W$
14.        else /* $i = 1$ */
15.            if $q' \in F$ then
16.                add $([q, 1], a, [q', 1])$ to $\delta'$
17.                if $[q', 1] \notin Q'$ then add $[q', 1]$ to $W$
18.        return $(Q', \Sigma, q'_0, \delta', \mathcal{F}')$
\( \mathcal{F} = \{ F_0, F_1 \} \)

\( F_0 = \{ q \} \)

\( F_1 = \{ r \} \)

\( \mathcal{F}'_0 = \{ [q, 1] \} \)

\( \mathcal{F}'_1 = \{ [r, 1] \} \)
Equivalence of NMAs and DMAs

• Theorem (Safra): Any NBA with \( n \) states can be effectively transformed into a DMA of size \( n^{O(n)} \).

  **Proof:** Omitted.

• DMA for \((a + b)^* b^\omega\):

![Diagram of a DMA](image)

- with accepting condition
  \[
  \{ \{q_1\} \}
  \]
Question: Are there other classes of omega-automata with
   – the same expressive power as NBAs or NGAs, and
   – with equivalent deterministic and nondeterministic versions?

Answer: Yes, Muller automata
Is the quest over?

- Recall the translation NBA $\Rightarrow$ NMA
- The NMA has the same structure as the NBA; its accepting condition are all the good sets of states.
- The translation has exponential complexity.

New question: Is there a class of $\omega$-automata with
- the same expressive power as NBAs,
- equivalent deterministic and nondeterministic versions, and
- polynomial conversions to and from Büchi automata?
Rabin automata

• The acceptance condition is a set of pairs
  \[ \{ \langle F_0, G_0 \rangle, \ldots, \langle F_{m-1}, G_{m-1} \rangle \} \]

• A run \( \rho \) is accepting if there is a pair
  \( \langle F_i, G_i \rangle \) such that \( \rho \) visits the set \( F_i \) infinitely often and the set \( G_i \) finitely often.

• Translations \( \text{NBA} \Rightarrow \text{NRA} \) and \( \text{NRA} \Rightarrow \text{NBA} \) are left as an exercise.

• **Theorem (Safra):** Any NBA with \( n \) states can be effectively transformed into a DRA with \( n^O(n) \) states and \( O(n) \) accepting pairs.