Mathematical Foundation of Computer Sciences III

Automata on Infinite Objects

Guoqiang Li

School of Software, Shanghai Jiao Tong University

Büchi Automata

Büchi Automata

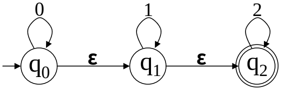
Accept ω-words, instead of finite words.

Acceptance condition is different.

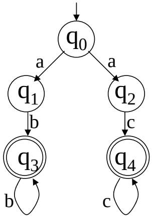
For a Büchi automaton $A = (S, \Sigma, \delta, q_0, F)$, ω-word $w = a_0 a_1 \ldots \in \Sigma^\omega$ is accepted if there exists $q_0 \xrightarrow{a_0} q_1 \xrightarrow{a_1} \ldots$ for $(q_i, a_i, q_{i+1}) \in \delta$ across F infinitely often. $L(A)$ is the set of accepted ω-words.

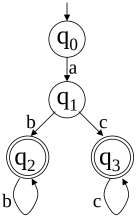
$L(A)$ is called regular (ω-language).

Examples of Büchi Automata



Accepts $\{0^*1^*2^\omega\}$



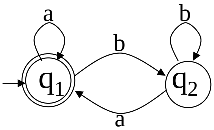


Accepts $\{ab^\omega, ac^\omega\}$

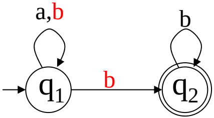
Deterministic VS. Non-deterministic

Accepted by deterministic Büchi automata?

- $(b^* a)^\omega$
  - a appears infinitely many
  - Yes!
- $(b^* a)^* b^\omega$
  - $\{a, b\}^\omega - (b^* a)^\omega$
  - No!



$(b^* a)^\omega$



$(b^* a)^* b^\omega$

Variation on Automata for ω-Words

Let $Inf(\sigma)$ be the set of states that a path σ across infinitely often. Let $\alpha \in \Sigma^\omega$

Büchi automata $A = (S, \Sigma, \delta, q_0, F)$

- α has a path σ such that $Inf(\sigma) \cap F \neq \emptyset$.

Muller automata $A = (S, \Sigma, \delta, q_0, \{F_1, \ldots, F_m\})$

- α has a path σ such that $Inf(\sigma) = F_i$.

Rabin automata $A = (S, \Sigma, \delta, q_0, \{(L_1, M_1), \ldots, (L_m, M_m)\})$

- α has a path σ such that $Inf(\sigma) \cap L_i = \emptyset$, $Inf(\sigma) \cap M_i \neq \emptyset$.

If nondeterministic, they are all equivalent.

Algorithm Reuse

Q: Which algorithms of finite automata can be reused in Büchi automata?

- Union?
- Concatenation?
- ω-Star?
- Intersection?
- Complement?
- Emptiness?
- Subset?

Operations of Büchi Automata

Two Büchi automata $A = (S, \Sigma, \delta, q_0, F)$, $B = (S', \Sigma, \delta', q_0', F')$, and a finite automaton $C = (S'', \Sigma, \delta'', q_0'', F'')$

Union: Büchi automaton that accepts $L(A) \cup L(B)$

$(S \cup S' \cup \{q\}, \Sigma, \delta \cup \delta' \cup \{(q, \varepsilon, q_0), (q, \varepsilon, q_0')\}, q, F \cup F')$

Concatenation: Concatenation between $L(C) \circ L(A)$

ω-Star: $L(A)^\omega$

Intersection of Büchi Automata

$A = (S, \Sigma, \delta, q_0, F), B = (S', \Sigma, \delta', q_0', F')$,

Büchi automaton that accepts $L(A) \cap L(B)$

$(S \times S' \times \{0, 1\}, \Sigma, \delta'', (q_0, q_0', 0), F \times S' \times \{0\} \cup S \times F' \times \{1\})$

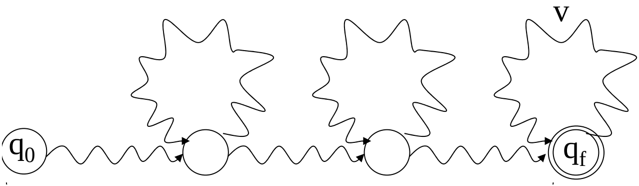
where $\delta'' = \{((s, s', i), a, (t, t', j)) \mid (s, a, t) \in \delta, (s', a, t') \in \delta'$,

- $j = 1$ if either $i = 0$ and $t \in F$, or $i = 1$ and $t' \notin F'$,
- $j = 0$ if either $i = 0$ and $t \notin F$, or $i = 1$ and $t' \in F'$

$\}$

Emptiness

For a Büchi automaton $A = (S, \Sigma, \delta, q_0, F)$, $L(A) \neq \emptyset$ implies $\exists u, v \in \Sigma^*$ such that $|u|, |v| \leq |S|$ and $uv^\omega \in L(A)$.



Emptiness

SCC!

Complement of Büchi Automata

For a Büchi automaton $A = (S, \Sigma, \delta, q_0, F)$,

Even if A is deterministic, A^c may be non-deterministic.

- $(b^* a)^\omega$

But, deterministic $\subset$ non-deterministic . . .

The Büchi automaton version of Myhill-Nerode Theorem discussion is required.

Several ways for Complement

Via Muller Automata

Via Alternating Automata

Miyano, S., Hayashi, T., Alternating Finite Automata on omega-Words. TCS 32, pp.321-330, 1984

Explicit representation by congruence classes (this also gives minimization)

Many many new techniques recently, which is still a hot topic nowadays.

Notations

For a Büchi automaton $A = (S, \Sigma, \delta, q_0, F)$,

$q \xrightarrow{u}_F q' \Leftrightarrow q \xrightarrow{u} q'$ across some $q_f \in F$

$u \sim_A v$ iff $\forall q, q' \in S$,

$$(q \xrightarrow{u} q' \Leftrightarrow q \xrightarrow{v} q') \wedge (q \xrightarrow{u}_F q' \Leftrightarrow q \xrightarrow{v}_F q')$$

Finite Congruence of Büchi Automata

$\sim_A$ is a finite congruence over Σ

$\sim_A$ classes U, V are regular

$U.V^\omega$ is regular ω-languages.

$L(A)$ as a Union of $U.V^\omega$

Lemma

For a Büchi automaton $A = (S, \Sigma, \delta, q_0, F)$,

$$L(A) = \bigcup_{U.V^\omega \cap L(A) \neq \emptyset} U.V^\omega$$

$L(A) \supseteq \bigcup_{U.V^\omega \cap L(A) \neq \emptyset} U.V^\omega$ is easy, since

$$U.V^\omega \cap L(A) \neq \emptyset \Rightarrow U.V^\omega \subseteq L(A)$$

$L(A) \subseteq \bigcup_{U.V^\omega \cap L(A) \neq \emptyset} U.V^\omega$ needs Ramsey's Theorem.

General Version

Let G be an infinite complete graph. Put a label from $\{1, 2, \ldots, k\}$ on each edge. Then, there exists an infinite complete sub-graph such that its each edge has the same label.

$L(A) \subseteq \bigcup_{U.V^\omega \cap L(A) \neq \emptyset} U.V^\omega$

For each $\alpha \in \Sigma^\omega$, there exist $\sim_A$ -classes U, V such that $\alpha \in U.V^\omega$.

Let $\alpha(m, n) = a_m a_{m+1} \dots a_{n-1}$ for $\alpha = a_1 a_2 a_3 \dots$

Regarding $\alpha(m, n) \in V$ as a label V, by Ramsey's Theorem, $\alpha(n_1, n_2), \alpha(n_2, n_3), \dots \in V$. If $\alpha(0, n_1) \in U$, then $\alpha \in U.V^\omega$.

Complement of $L(A)$

For a Büchi automaton $A = (S, \Sigma, \delta, q_0, F)$,

$$L(A) = \bigcup_{U.V^\omega \cap L(A) \neq \emptyset} U.V^\omega$$

Then,

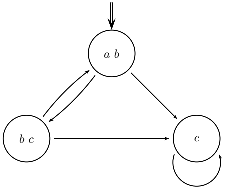
$$L^c(A) = \bigcup_{U.V^\omega \cap L(A) = \emptyset} U.V^\omega$$

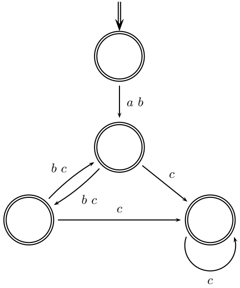
U, V are regular; thus, $L^c(A)$ is regular.

LTL Logic

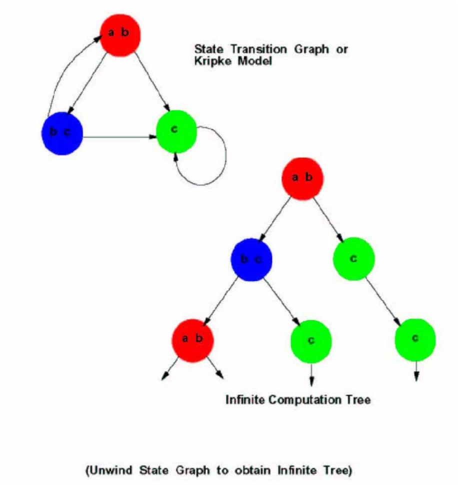
Models: Systems Vs. Machines

- Kripke structure: $M = (S, S_0, R, L)$
  - S, finite set of state
  - $S_0 \subseteq S$, initial state
  - $R \subseteq S \times S$, transition relations
  - $L : S \to 2^{AP}$, status label function
    (AP: atomic propositions)
- FA: $\mathcal{A} = (\Sigma, Q, Q_0, F, \delta)$
  - A, finite set of input alphabet
  - Q, finite set of control location
  - $Q_0 \subseteq Q$, initial control locations
  - $F \subseteq Q$, final control locations
  - $\delta \subseteq Q \times \Sigma \times Q$, transitions
- Büchi automata, pushdown automata,. . .

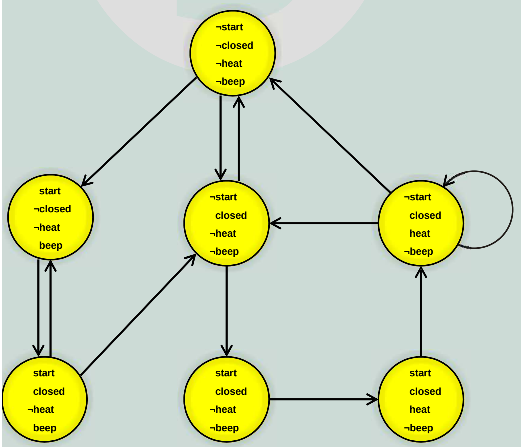




Models: Finite Systems Vs. Infinite Computation Tree

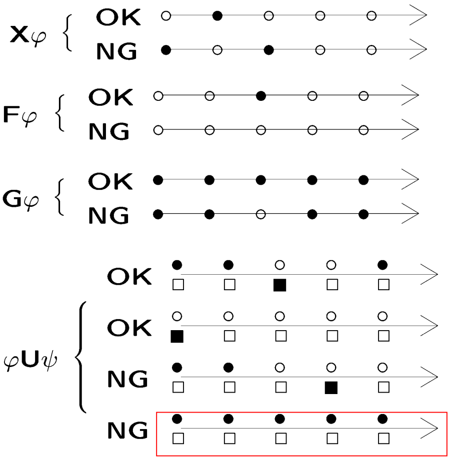


An Example: An Microwave Oven Example



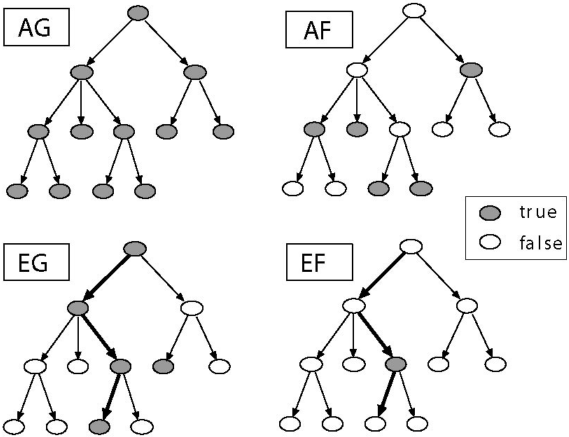
Logic-Based MC: Temporal Operators

- Next
- Finally
- Globally
- Until



$\bullet : \varphi$, $\blacksquare : \psi$, $\circ : \neg\varphi$, $\square : \neg\psi$

Logic-Based MC: Path Operators, *A*, *E*



- *AG*: safety, bad things will never happen.
- *AF*: liveness, good things will eventually happen.

State Formula & path formulas

Let AP be the set of atomic proposition names. The syntax of state formulas is given by the following rules:

- If $p \in AP$, then p is a state formula.
- If f and g are state formulas, then $\neg f$, $f \vee g$ and $f \wedge g$ are state formulas.
- If f is a path formula, then $E\,f$ and $A\,f$ are state formulas.
- If f is a state formula, then f is also a path formula.
- If f and g are path formulas, then $\neg f$, $f \vee g$, $f \wedge g$, $X\,f$, $F\,f$, $G\,f$ and $f\,U\,g$ are path formulas.

Formal Description

1.	$M, s \models p$	$\Leftrightarrow$	$p \in L(s)$.
2.	$M, s \models \neg f_1$	$\Leftrightarrow$	$M, s \not\models f_1$.
3.	$M, s \models f_1 \vee f_2$	$\Leftrightarrow$	$M, s \models f_1$ or $M, s \models f_2$.
4.	$M, s \models f_1 \wedge f_2$	$\Leftrightarrow$	$M, s \models f_1$ and $M, s \models f_2$.
5.	$M, s \models \mathbf{E}\ g_1$	$\Leftrightarrow$	there is a path π from s such that $M, \pi \models g_1$.
6.	$M, s \models \mathbf{A}\ g_1$	$\Leftrightarrow$	for every path π starting from s, $M, \pi \models g_1$.
7.	$M, \pi \models f_1$	$\Leftrightarrow$	s is the first state of π and $M, s \models f_1$.
8.	$M, \pi \models \neg g_1$	$\Leftrightarrow$	$M, \pi \not\models g_1$.
9.	$M, \pi \models g_1 \vee g_2$	$\Leftrightarrow$	$M, \pi \models g_1$ or $M, \pi \models g_2$.
10.	$M, \pi \models g_1 \wedge g_2$	$\Leftrightarrow$	$M, \pi \models g_1$ and $M, \pi \models g_2$.
11.	$M, \pi \models \mathbf{X}\, g_1$	$\Leftrightarrow$	$M, \pi^1 \models g_1$.
12.	$M, \pi \models \mathbf{F}\, g_1$	$\Leftrightarrow$	there exists a $k \geq 0$ such that $M, \pi^k \models g_1$.
13.	$M, \pi \models \mathbf{G}\, g_1$	$\Leftrightarrow$	for all $i \geq 0$, $M, \pi^i \models g_1$.
14.	$M, \pi \models g_1\, \mathbf{U}\, g_2$	$\Leftrightarrow$	there exists a $k \geq 0$ such that $M, \pi^k \models g_2$ and for all $0 \leq j < k$, $M, \pi^j \models g_1$.

Example Specification

- $EF(\texttt{Start} \land \neg \texttt{Ready})$
  - It is possible to get to a state where `Start` holds but `Ready` does not hold.
- $AG(\texttt{Req} \rightarrow AF\,\texttt{Ack})$
  - If a request occurs, then it will be eventually acknowledged.
- $AG(AF\,\texttt{DeviceEnabled})$
  - The proposition `DeviceEnabled` holds infinitely often on every computation path.
- $AG(EF\,\texttt{Restart})$
  - From any state it is possible to get to the `Restart`.

Example Specification

- $AG\ (\texttt{request} \rightarrow F\ \texttt{grant})$
  - each `request` will be finally `grant(ed)`.
- $AG(\neg(\neg\texttt{request}\ U\ \texttt{grant}))$
  - each `grant` follows some `request`.
- $AGF\ \texttt{request}$
  - `request` occurs infinitely often.

Subclass of CTL, LTL

- CTL: temporal operators must be immediately followed by path quantifiers.
  - e.g., $AF\varphi$, $EG\varphi$, $AXEG\varphi$, $EXA(\varphi U\psi)$
- LTL: path quantifiers are allowed only at the outermost position.
  - e.g., $AGF\varphi$, $EX(\varphi U\psi)$, $A(F\varphi \vee G\psi)$

Büchi Automata and Logic

Generalized Büchi Automata

The acceptance condition of a generalized Büchi automaton is a set of sets of states $\mathcal{F} \subseteq 2^S$, and the requirement is that some state of each of the sets $F_i \in \mathcal{F}$ appears infinitely often.

More formally, a generalized Büchi $A = (\Sigma, S, \delta, S_0, \mathcal{F})$ accepts a word w if there is a labeling ρ of w by states of A that satisfies the same first two conditions as given for Büchi automata, the third being replaced by:

- For each $F_i \in \mathcal{F}$, $inf(\rho) \cap F_i \neq \emptyset$.

Encoding Generalized Büchi automata

Given a generalized Büchi automaton $A = (\Sigma, S, \delta, S_0, F4)$, where $F = \{F_1, \ldots, F_k\}$, the Büchi automaton $A' = (\Sigma, S', \delta', S_0', F')$ defined as follows accepts the same language as A.

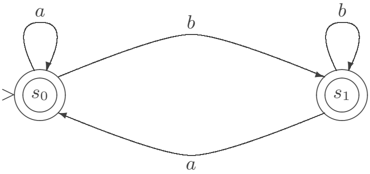
- $S' = S \times \{1, \ldots, k\}$.
- $S_0' = S_0 \times \{1\}$.
- δ' is defined by $(t, i) \in \delta'((s, j), a)$ if

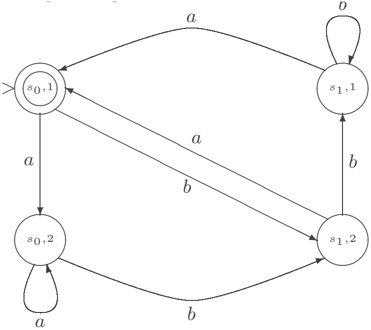
$$t \in \delta(s, a) \wedge \begin{cases} i = j & \text{if } s \notin F_j \\ i = (j \mod k) + 1 & \text{otherwise} \end{cases}$$

.

- $F' = F_1 \times \{1\}$.

An Example





Problem Statement

Given an LTL formula φ built from a set of atomic propositions AP, construct an automaton on infinite words over the alphabet 2^{AP} that accepts exactly the infinite sequences satisfying φ.

A Dialect of LTL Logic

- $true$, $false$, p, and $\neg p$, for all $p \in AP$;
- $\varphi_1 \wedge \varphi_2$ and $\varphi_1 \vee \varphi_2$, where φ_1 and φ_2 are LTL formulas;
- $X\varphi_1$, $\varphi_1 U \varphi_2$, and $\varphi_1 R \varphi_2$, where φ_1 and φ_2 are LTL formulas.

$\varphi_1 R \varphi_2$: it requires φ_2 always be true, a requirement that is released as soon as φ_1 becomes true.

The Way to Handle Negation

$$\sigma \not\models \varphi_1 U \varphi_2 \Leftrightarrow \sigma \models (\neg\varphi_1) R (\neg\varphi_2)$$

$$\sigma \not\models X\varphi \Leftrightarrow \sigma \models X\neg\varphi$$

Closure of a Formula

$$\varphi \in cl(\varphi)$$
$$\varphi_1 \wedge \varphi_2 \in cl(\varphi) \Rightarrow \varphi_1, \varphi_2 \in cl(\varphi)$$
$$\varphi_1 \vee \varphi_2 \in cl(\varphi) \Rightarrow \varphi_1, \varphi_2 \in cl(\varphi)$$
$$X\varphi_1 \in cl(\varphi) \Rightarrow \varphi_1 \in cl(\varphi)$$
$$\varphi_1 U \varphi_2 \in cl(\varphi) \Rightarrow \varphi_1, \varphi_2 \in cl(\varphi)$$
$$\varphi_1 R \varphi_2 \in cl(\varphi) \Rightarrow \varphi_1, \varphi_2 \in cl(\varphi)$$

Example

$$cl(F\neg p) = cl(true U \neg p) = \{F\neg p, \neg p, true\}$$

Hintikka Structure

A valid closure labeling $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ of a sequence $\sigma : \mathbb{N} \to 2^{AP}$ has to satisfy.

If a formula $\varphi_1 \in cl(\varphi)$ labels a position i, then the sequence $\sigma^i \models \varphi_1$.

Rules for Labeling Sequences

1. $false \notin \tau(i)$;
2. for $p \in AP$, if $p \in \tau(i)$ then $p \in \sigma(i)$, and if $\neg p \in \tau(i)$ then $p \notin \sigma(i)$;
3. if $\varphi_1 \wedge \varphi_2 \in \tau(i)$ then $\varphi_1 \in \tau(i)$ and $\varphi_2 \in \tau(i)$;
4. if $\varphi_1 \vee \varphi_2 \in \tau(i)$ then $\varphi_1 \in \tau(i)$ or $\varphi_2 \in \tau(i)$;
5. if $X\varphi_1 \in \tau(i)$ then $\varphi_1 \in \tau(i+1)$;
6. if $\varphi_1 U \varphi_2 \in \tau(i)$ then either $\varphi_2 \in \tau(i)$, or $\varphi_1 \in \tau(i)$ and $\varphi_1 U \varphi_2 \in \tau(i+1)$;
7. if $\varphi_1 R \varphi_2 \in \tau(i)$ then $\varphi_2 \in \tau(i)$, and either $\varphi_1 \in \tau(i)$ or $\varphi_1 R \varphi_2 \in \tau(i+1)$;
8. if $\varphi_1 U \varphi_2 \in \tau(i)$ then there exists a $j > i$ such that $\varphi_2 \in \tau(j)$.

Key Lemmas

Lemma.

Consider a formula φ defined over a set of propositions AP, a sequence $\sigma : \mathbb{N} \to 2^{AP}$, and a closure labeling $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ satisfying rules 1-8. For every formula $\varphi' \in cl(\varphi)$ and $i \geq 0$, one has that if $\varphi' \in \tau(i)$ then $\sigma^i \models \varphi'$.

Lemma.

Consider a formula φ defined over a set of propositions AP and a sequence $\sigma : \mathbb{N} \to 2^{AP}$. If $\sigma \models \varphi$, there exists a closure labeling $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ satisfying rules 1-8 and such that $\varphi \in \tau(0)$.

Theorem

Consider a formula φ defined over a set of propositions AP and a sequence $\sigma : \mathbb{N} \to 2^{AP}$. One then has that $\sigma \models \varphi$, iff there is a closure labeling $\tau : \mathbb{N} \to 2^{cl(\varphi)}$ satisfying rules 1-8 and such that $\varphi \in \tau(0)$.

Encoding to Büchi Automata Σ, S

Given a formula φ, a generalized Büchi automaton accepting exactly the sequences $\sigma : \mathbb{N} \to 2^{AP}$ satisfying φ can be defined as follows. The automaton is $A_\varphi = (S, \Sigma, \delta, S_0, \mathcal{F})$ where,

- $\Sigma = 2^{AP}$,
- $S \subseteq 2^{cl(\varphi)}$, and for each $s \in S$
  - $false \notin s$;
  - if $\varphi_1 \wedge \varphi_2 \in s$, then $\varphi_1 \in s$ and $\varphi_2 \in s$.
  - if $\varphi_1 \vee \varphi_2 \in s$, then $\varphi_1 \in s$ or $\varphi_2 \in s$.

Encoding to Büchi Automata δ, S_0

Given a formula φ, a generalized Büchi automaton accepting exactly the sequences $\sigma : \mathbb{N} \to 2^{AP}$ satisfying φ can be defined as follows. The automaton is $A_\varphi = (\Sigma, S, \delta, S_0, \mathcal{F})$ where,

- $t \in \delta(s, a)$ iff,
  - For all $p \in AP$, if $p \in s$ then $p \in a$.
  - For all $p \in AP$, if $\neg p \in s$ then $p \notin a$.
  - If $X\varphi \in s$, then $\varphi \in t$.
  - If $\varphi_1 U \varphi_2 \in s$ then either $\varphi_2 \in s$, or $\varphi_1 \in s$ and $\varphi_1 U \varphi_2 \in t$.
  - If $\varphi_1 R \varphi_2 \in s$ then $\varphi_2 \in s$ and either $\varphi_1 \in s$, or $\varphi_1 R \varphi_2 \in t$.
- $S_0 = \{s \in S \mid \varphi \in s\}$.

Encoding to Büchi Automata $\mathcal{F}$

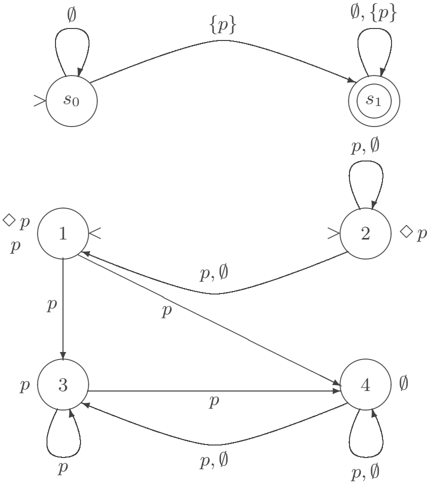
Given a formula φ, a generalized Büchi automaton accepting exactly the sequences $\sigma : \mathbb{N} \to 2^{AP}$ satisfying φ can be defined as follows. The automaton is $A_\varphi = (\Sigma, S, \delta, S_0, \mathcal{F})$ where,

- If the eventualities appearing in $cl(\varphi)$ are $e_1(\varphi_1), \ldots, e_m(\varphi_m)$, $\mathcal{F} = \{\Phi_1, \Phi_2 \ldots \Phi_m\}$, where

$$\Phi_i = \{s \in S \mid e_i(\varphi_i), \varphi_i \in s \vee e_i(\varphi_i) \notin s\}$$

for every eventuality formula $e(\varphi') = \varphi U \varphi'$

An Example $F\ p$



Optimizations

Omitting Redundant Transitions

Building the Automaton by Need

Identifying Equivalent States

Simplifying the Formula.

Early Detection of Inconsistencies.

Moving Propositions from States to Transitions.