# Software Engineering using Formal Methods

## Formal Modeling with Linear Temporal Logic

Wolfgang Ahrendt & Wojciech Mostowski & Richard Bubel

14th September 2011

# Syntax of Propositional Logic

**Signature**

A set of Propositional Variables $\mathcal{P}$ (with typical elements $p, q, r, \ldots$)

# Syntax of Propositional Logic

## Signature

A set of Propositional Variables $\mathcal{P}$ (with typical elements $p, q, r, \ldots$)

## Propositional Connectives

$\text{true},\ \text{false},\ \wedge,\ \vee,\ \neg,\ \rightarrow,\ \leftrightarrow$

# Syntax of Propositional Logic

## Signature

A set of Propositional Variables $\mathcal{P}$ (with typical elements $p, q, r, \ldots$)

## Propositional Connectives

$\text{true},\ \text{false},\ \wedge,\ \vee,\ \neg,\ \rightarrow,\ \leftrightarrow$

## Set of Propositional Formulas $For_0$

- Truth constants $\text{true}$, $\text{false}$ and variables $\mathcal{P}$ are formulas
- If $\phi$ and $\psi$ are formulas then

  $$\neg\phi, \quad (\phi \wedge \psi), \quad (\phi \vee \psi), \quad (\phi \rightarrow \psi), \quad (\phi \leftrightarrow \psi)$$

  are also formulas
- There are no other formulas (inductive definition)

# Remark on Concrete Syntax

| | Text book | SPIN |
|---|---|---|
| Negation | $\neg$ | ! |
| Conjunction | $\wedge$ | && |
| Disjunction | $\vee$ | \|\| |
| Implication | $\rightarrow$, $\supset$ | -> |
| Equivalence | $\leftrightarrow$ | <-> |

# Remark on Concrete Syntax

| | Text book | SPIN |
|---|---|---|
| Negation | $\neg$ | ! |
| Conjunction | $\wedge$ | && |
| Disjunction | $\vee$ | \|\| |
| Implication | $\rightarrow$, $\supset$ | -> |
| Equivalence | $\leftrightarrow$ | <-> |

We use mostly the textbook notation.
Except for tool specific slides.

# Examples

Let $\mathcal{P} = \{p, q, r\}$ be the set of propositional variables.

Are the following character sequences also propositional formulas?

- $(\mathrm{true} \rightarrow p)$

# Examples

Let $\mathcal{P} = \{p, q, r\}$ be the set of propositional variables.

Are the following character sequences also propositional formulas?

- $(\text{true} \rightarrow p)$ ✔

# Examples

Let $\mathcal{P} = \{p, q, r\}$ be the set of propositional variables.

Are the following character sequences also propositional formulas?

- $(\mathrm{true} \rightarrow p)$ ✔
- $((p(q \wedge r)) \vee p)$

# Examples

Let $\mathcal{P} = \{p, q, r\}$ be the set of propositional variables.

Are the following character sequences also propositional formulas?

- $(\mathrm{true} \rightarrow p)$ ✔
- $((p(q \wedge r)) \vee p)$ ✘

# Examples

Let $\mathcal{P} = \{p, q, r\}$ be the set of propositional variables.

Are the following character sequences also propositional formulas?

- $(\mathrm{true} \rightarrow p)$ ✔
- $((p(q \wedge r)) \vee p)$ ✘
- $(p \rightarrow (q\wedge))$

# Examples

Let $\mathcal{P} = \{p, q, r\}$ be the set of propositional variables.

Are the following character sequences also propositional formulas?

- $(\text{true} \rightarrow p)$ ✔
- $((p(q \wedge r)) \vee p)$ ✘
- $(p \rightarrow (q\wedge))$ ✘

# Examples

Let $\mathcal{P} = \{p, q, r\}$ be the set of propositional variables.

Are the following character sequences also propositional formulas?

- $(\text{true} \rightarrow p)$ ✔
- $((p(q \wedge r)) \vee p)$ ✘
- $(p \rightarrow (q\wedge))$ ✘
- $(\mathit{false} \wedge (p \rightarrow (q \wedge r)))$

# Examples

Let $\mathcal{P} = \{p, q, r\}$ be the set of propositional variables.

Are the following character sequences also propositional formulas?

- $(\text{true} \rightarrow p)$ ✔
- $((p(q \wedge r)) \vee p)$ ✘
- $(p \rightarrow (q\wedge))$ ✘
- $(\mathit{false} \wedge (p \rightarrow (q \wedge r)))$ ✔

# Semantics of Propositional Logic

**Interpretation $\mathcal{I}$**

Assigns a truth value to each propositional variable

$$\mathcal{I} : \mathcal{P} \rightarrow \{T, F\}$$

# Semantics of Propositional Logic

**Interpretation $\mathcal{I}$**

Assigns a truth value to each propositional variable

$$\mathcal{I} : \mathcal{P} \to \{T, F\}$$

**Example**

Let $\mathcal{P} = \{p, q\}$

$$(p \ \to \ (q \ \to \ p))$$

| | $p$ | $q$ |
|---|---|---|
| $\mathcal{I}_1$ | $F$ | $F$ |
| $\mathcal{I}_2$ | $T$ | $F$ |
| $\ldots$ | | |

# Semantics of Propositional Logic

**Interpretation $\mathcal{I}$**

Assigns a truth value to each propositional variable

$$\mathcal{I} : \mathcal{P} \to \{T, F\}$$

**Example**

Let $\mathcal{P} = \{p, q\}$

$$(p \ \to \ (q \ \to \ p))$$

| | $p$ | $q$ |
|---|---|---|
| $\mathcal{I}_1$ | $F$ | $F$ |
| $\mathcal{I}_2$ | $T$ | $F$ |
| | $\ldots$ | |

How to evaluate $(p \ \to \ (q \ \to \ p))$?

# Semantics of Propositional Logic

**Interpretation $\mathcal{I}$**

Assigns a truth value to each propositional variable

$$\mathcal{I} : \mathcal{P} \to \{T, F\}$$

**Valuation function**

$val_{\mathcal{I}}$: Continuation of $\mathcal{I}$ on $For_0$

$$val_{\mathcal{I}} : For_0 \ \to \ \{T, F\}$$

$val_{\mathcal{I}}(p_i) = \mathcal{I}(p_i)$
$val_{\mathcal{I}}(\text{true}) = T$
$val_{\mathcal{I}}(\text{false}) = F$

(cont'd next page)

# Semantics of Propositional Logic (Cont'd)

## Valuation function (Cont'd)

$$val_{\mathcal{I}}(\neg\phi) = \begin{cases} T & \text{if } val_{\mathcal{I}}(\phi) = F \\ F & otherwise \end{cases}$$

$$val_{\mathcal{I}}(\phi \wedge \psi) = \begin{cases} T & \text{if } val_{\mathcal{I}}(\phi) = T \textbf{ and } val_{\mathcal{I}}(\psi) = T \\ F & otherwise \end{cases}$$

$$val_{\mathcal{I}}(\phi \vee \psi) = \begin{cases} T & \text{if } val_{\mathcal{I}}(\phi) = T \textbf{ or } val_{\mathcal{I}}(\psi) = T \\ F & otherwise \end{cases}$$

$$val_{\mathcal{I}}(\phi \rightarrow \psi) = \begin{cases} T & \text{if } val_{\mathcal{I}}(\phi) = F \textbf{ or } val_{\mathcal{I}}(\psi) = T \\ F & otherwise \end{cases}$$

$$val_{\mathcal{I}}(\phi \leftrightarrow \psi) = \begin{cases} T & \text{if } val_{\mathcal{I}}(\phi) = val_{\mathcal{I}}(\psi) \\ F & otherwise \end{cases}$$

# Examples - Cont'd

**Example**

Let $\mathcal{P} = \{p, q\}$

$$(p \rightarrow (q \rightarrow p))$$

| | $p$ | $q$ |
|---|---|---|
| $\mathcal{I}_1$ | $F$ | $F$ |
| $\mathcal{I}_2$ | $T$ | $F$ |
| | $\ldots$ | |

How to evaluate $(p \rightarrow (q \rightarrow p))$ in $\mathcal{I}_2$?

# Examples - Cont'd

**Example**

Let $\mathcal{P} = \{p, q\}$

$$(p \rightarrow (q \rightarrow p))$$

| | $p$ | $q$ |
|---|---|---|
| $\mathcal{I}_1$ | $F$ | $F$ |
| $\mathcal{I}_2$ | $T$ | $F$ |
| | $\ldots$ | |

How to evaluate $(p \rightarrow (q \rightarrow p))$ in $\mathcal{I}_2$?

**Valuation in $\mathcal{I}_2$**

$val_{\mathcal{I}_2}(\ q \rightarrow p\ ) \quad =$

# Examples - Cont'd

**Example**

Let $\mathcal{P} = \{p, q\}$

$$(p \rightarrow (q \rightarrow p))$$

| | $p$ | $q$ |
|---|---|---|
| $\mathcal{I}_1$ | $F$ | $F$ |
| $\mathcal{I}_2$ | $T$ | $F$ |

$\ldots$

How to evaluate $(p \rightarrow (q \rightarrow p))$ in $\mathcal{I}_2$?

**Valuation in $\mathcal{I}_2$**

$val_{\mathcal{I}_2}(\ q \rightarrow p\ ) \quad = \quad T$

# Examples - Cont'd

**Example**

Let $\mathcal{P} = \{p, q\}$

$$(p \rightarrow (q \rightarrow p))$$

| | $p$ | $q$ |
|---|---|---|
| $\mathcal{I}_1$ | $F$ | $F$ |
| $\mathcal{I}_2$ | $T$ | $F$ |

$\ldots$

How to evaluate $(p \rightarrow (q \rightarrow p))$ in $\mathcal{I}_2$?

**Valuation in $\mathcal{I}_2$**

$val_{\mathcal{I}_2}(\ q \rightarrow p\ ) \quad = \quad T$

$val_{\mathcal{I}_2}(\ p \rightarrow (q \rightarrow p)\ ) \quad =$

# Examples - Cont'd

**Example**

Let $\mathcal{P} = \{p, q\}$

$$(p \rightarrow (q \rightarrow p))$$

| | $p$ | $q$ |
|---|---|---|
| $\mathcal{I}_1$ | $F$ | $F$ |
| $\mathcal{I}_2$ | $T$ | $F$ |
| | $\ldots$ | |

How to evaluate $(p \rightarrow (q \rightarrow p))$ in $\mathcal{I}_2$?

**Valuation in $\mathcal{I}_2$**

$val_{\mathcal{I}_2}(\ q \rightarrow p\ ) \quad = \quad T$

$val_{\mathcal{I}_2}(\ p \rightarrow (q \rightarrow p)\ ) \quad = \quad T$

# Semantic Notions of Propositional Logic

Let $\phi \in For_0$, $\Gamma \subset For_0$

**Definition (Satisfying Interpretation, Consequence Relation)**

$\mathcal{I}$ satisfies $\phi$ (write: $\mathcal{I} \models \phi$) iff $val_{\mathcal{I}}(\phi) = T$

$\phi$ follows from $\Gamma$ (write: $\Gamma \models \phi$) iff for all interpretations $\mathcal{I}$:

$$\text{If } \mathcal{I} \models \psi \text{ for all } \psi \in \Gamma \text{ then also } \mathcal{I} \models \phi$$

# Semantic Notions of Propositional Logic

Let $\phi \in For_0$, $\Gamma \subset For_0$

**Definition (Satisfying Interpretation, Consequence Relation)**

$\mathcal{I}$ satisfies $\phi$ (write: $\mathcal{I} \models \phi$) iff $val_{\mathcal{I}}(\phi) = T$

$\phi$ follows from $\Gamma$ (write: $\Gamma \models \phi$) iff for all interpretations $\mathcal{I}$:

$$\text{If } \mathcal{I} \models \psi \text{ for all } \psi \in \Gamma \text{ then also } \mathcal{I} \models \phi$$

**Definition (Satisfiability, Validity)**

A formula is satisfiable if it is valid in some interpretation.
If *every* interpretation satisfies $\phi$ (write: $\models \phi$) then $\phi$ is called valid.

# Examples

**Formula (same as before)**

$$p \rightarrow (q \rightarrow p)$$

# Examples

**Formula (same as before)**

$$p \rightarrow (q \rightarrow p)$$

Is this formula valid?

$$\models p \rightarrow (q \rightarrow p) \ ?$$

# Examples

$$p \wedge ((\neg p) \vee q)$$

Satisfiable?

# Examples

$$p \wedge ((\neg p) \vee q)$$

Satisfiable? ✔

# Examples

$$p \wedge ((\neg p) \vee q)$$

Satisfiable? ✔

Satisfying Interpretation?

# Examples

$$p \wedge ((\neg p) \vee q)$$

Satisfiable? ✔

Satisfying Interpretation? $\mathcal{I}(p) = T, \mathcal{I}(q) = T$

# Examples

$$p \wedge ((\neg p) \vee q)$$

Satisfiable? ✔

Satisfying Interpretation? $\mathcal{I}(p) = T, \mathcal{I}(q) = T$

Other Satisfying Interpretations?

# Examples

$$p \wedge ((\neg p) \vee q)$$

| | |
|---|---|
| Satisfiable? | ✔ |
| Satisfying Interpretation? | $\mathcal{I}(p) = T, \mathcal{I}(q) = T$ |
| Other Satisfying Interpretations? | ✘ |

# Examples

$$p \wedge ((\neg p) \vee q)$$

| | |
|---|---|
| Satisfiable? | ✔ |
| Satisfying Interpretation? | $\mathcal{I}(p) = T, \mathcal{I}(q) = T$ |
| Other Satisfying Interpretations? | ✘ |

Therefore, also not valid!

# Examples

$$p \wedge ((\neg p) \vee q)$$

Satisfiable? ✔

Satisfying Interpretation? $\mathcal{I}(p) = T, \mathcal{I}(q) = T$

Other Satisfying Interpretations? ✘

Therefore, also not valid!

$$p \wedge ((\neg p) \vee q) \models q \vee r$$

Does it hold?

# Examples

$$p \wedge ((\neg p) \vee q)$$

Satisfiable? ✔

Satisfying Interpretation? $\mathcal{I}(p) = T, \mathcal{I}(q) = T$

Other Satisfying Interpretations? ✘

Therefore, also not valid!

$$p \wedge ((\neg p) \vee q) \models q \vee r$$

Does it hold? Yes. Why?

# Formalisation

```
1 byte n;
2 active proctype [2] p() {
3   n = 0;
4   n = n + 1
5 }
```

How can we model the above PROMELA program P?

# Formalisation

```
1 byte n;
2 active proctype [2] p() {
3   n = 0;
4   n = n + 1
5 }
```

How can we model the above PROMELA program P?

A model of P is a propositional formula $\phi_{\mathrm{P}}$, which is true if and only if (iff) it describes a possible state of P.

# Formalisation

```
1 byte n;
2 active proctype [2] p() {
3   n = 0;
4   n = n + 1
5 }
```

$\mathcal{P}$ : $N0, N1, N2, \ldots, N255$
$PC0\_3, PC0\_4, PC0\_5, PC1\_3, PC1\_4, PC1\_5$

Which interpretations do we need to 'exclude'?

$\phi_{\mathrm{P}} :=$

$($

# Formalisation

```
1 byte n;
2 active proctype [2] p() {
3   n = 0;
4   n = n + 1
5 }
```

$\mathcal{P}$ : $N0, N1, N2, \ldots, N255$
$PC0\_3, PC0\_4, PC0\_5, PC1\_3, PC1\_4, PC1\_5$

Which interpretations do we need to 'exclude'?

- The variable `n` has exactly one value at one time.

$$\phi_{\mathrm{P}} := \left( ((N0 \wedge \neg N1 \wedge \ldots \wedge \neg N255) \vee \ldots \vee (\neg N0 \wedge \ldots \wedge \neg N254 \wedge N255)) \right.$$

# Formalisation

```
1 byte n;
2 active proctype [2] p() {
3   n = 0;
4   n = n + 1
5 }
```

$\mathcal{P}$ : $N0, N1, N2, \ldots, N255$
$PC0\_3, PC0\_4, PC0\_5, PC1\_3, PC1\_4, PC1\_5$

Which interpretations do we need to 'exclude'?

- The variable `n` has exactly one value at one time.
- A process cannot be at two positions at the same time.

$$\phi_{\mathrm{P}} := \Big( ((N0 \wedge \neg N1 \wedge \ldots \wedge \neg N255) \vee \ldots \vee (\neg N0 \wedge \ldots \wedge \neg N254 \wedge N255)) \wedge ((PC0\_3 \wedge \neg PC0\_4 \wedge \neg PC0\_5) \vee \ldots)$$

# Formalisation

```
1 byte n;
2 active proctype [2] p() {
3   n = 0;
4   n = n + 1
5 }
```

$\mathcal{P}$ : $N0, N1, N2, \ldots, N255$
$PC0\_3, PC0\_4, PC0\_5, PC1\_3, PC1\_4, PC1\_5$

Which interpretations do we need to 'exclude'?

- The variable `n` has exactly one value at one time.
- A process cannot be at two positions at the same time.
- If neither process 0 nor process 1 are at position 5, then `n` is zero.
- ...

$$\phi_{\mathrm{P}} := \left( \begin{array}{l} ((N0 \wedge \neg N1 \wedge \ldots \wedge \neg N255) \vee \ldots \vee (\neg N0 \wedge \ldots \wedge \neg N254 \wedge N255)) \\ \wedge \ ((PC0\_3 \wedge \neg PC0\_4 \wedge \neg PC0\_5) \vee \ldots) \\ \ldots \end{array} \right)$$

# Is propositional logic enough?

As seen we can express for a program $P$ a formula $\Phi_P$ characterizing all ever reachable states.

The consequence relation

$$\Phi_p \models \Psi$$

# Is propositional logic enough?

As seen we can express for a program $P$ a formula $\Phi_P$ characterizing all ever reachable states.

The consequence relation

$$\Phi_p \models \Psi$$

which holds if all interpretation $I$ satisfying $\Phi_P$ also satisfy $\Psi$.

Intuitively: If the consequence relation holds then any possible state reachable by some run of program $P$ satisfies $\Psi$.

# Is propositional logic enough?

As seen we can express for a program $P$ a formula $\Phi_P$ characterizing all ever reachable states.

The consequence relation

$$\Phi_p \models \Psi$$

which holds if all interpretation $I$ satisfying $\Phi_P$ also satisfy $\Psi$.

Intuitively: If the consequence relation holds then any possible state reachable by some run of program $P$ satisfies $\Psi$.

How to express properties like: In any run of a program $P$

- $n$ will become greater than 0 eventually?
- $n$ changes its value infinitely often

# Is propositional logic enough?

As seen we can express for a program $P$ a formula $\Phi_P$ characterizing all ever reachable states.

The consequence relation

$$\Phi_p \models \Psi$$

which holds if all interpretation $I$ satisfying $\Phi_P$ also satisfy $\Psi$.

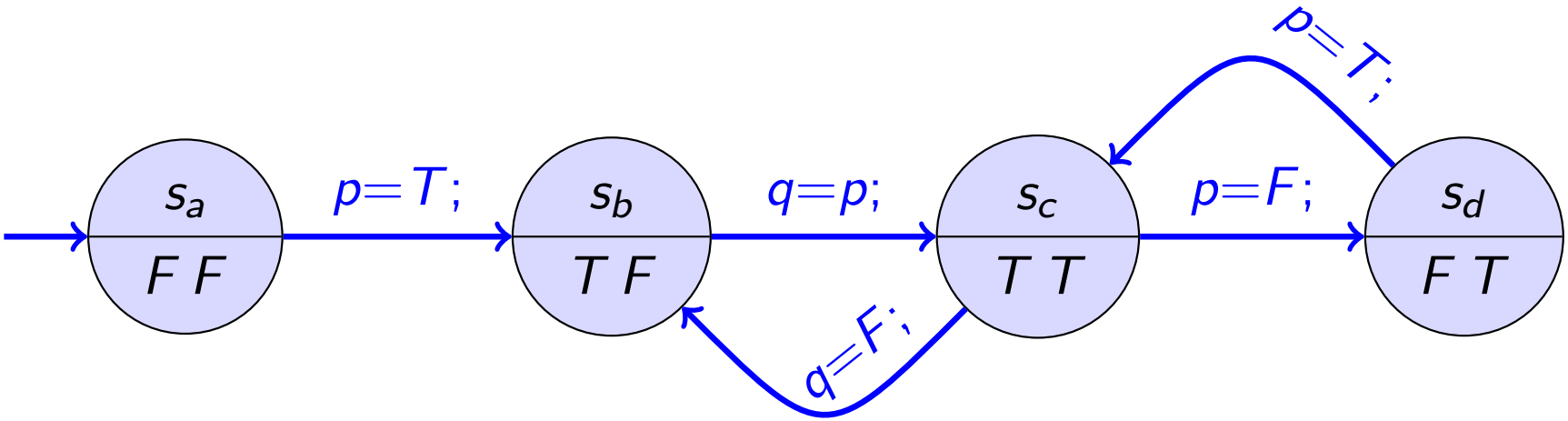
Intuitively: If the consequence relation holds then any possible state reachable by some run of program $P$ satisfies $\Psi$.

How to express properties like: In any run of a program $P$

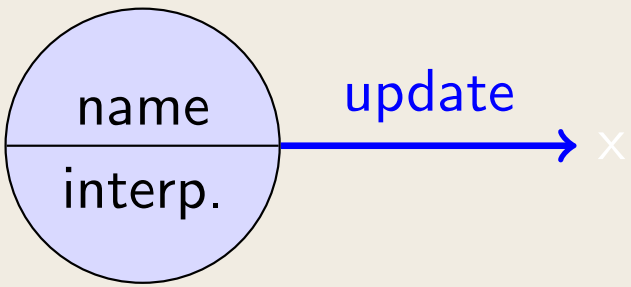
- $n$ will become greater than 0 eventually?
- $n$ changes its value infinitely often

$\Rightarrow$ Need a more expressive logic: Linear Temporal Logic

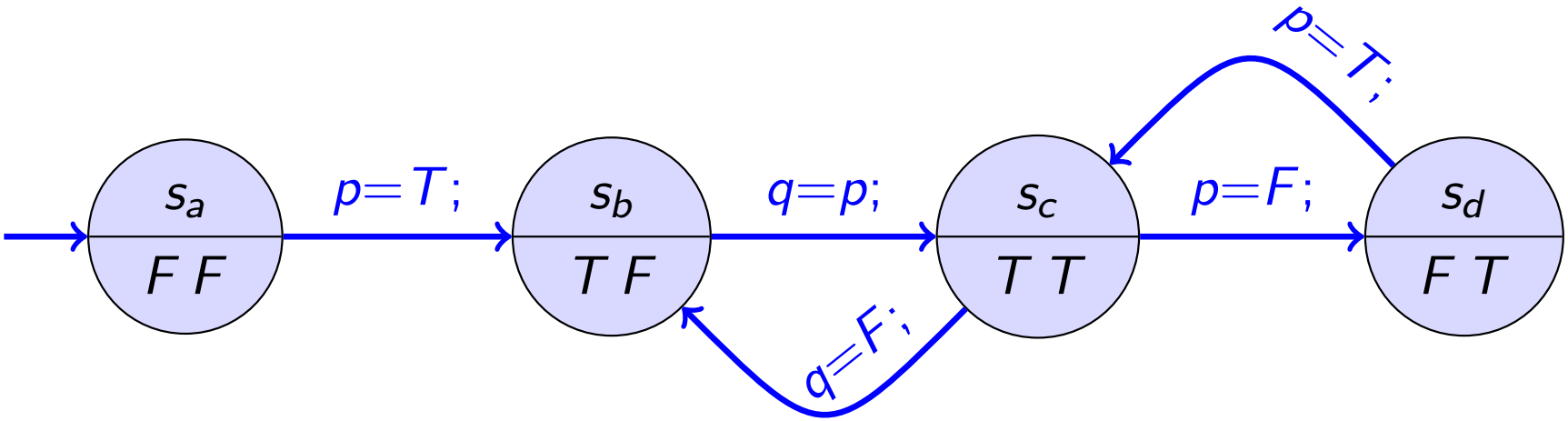
# Transition systems (aka. Kripke Structures)




## Notation



# Transition systems (aka. Kripke Structures)




- Each state has its own propositional interpretation!
- Computations, or *runs*, are infinite paths through states
- Infinitely many different runs
- How to express (for example) that either $p$ or $q$ changes its value infinitely often in each run?

# Temporal Logic

An extension of propositional logic that allows to specify properties of sets of runs

# Temporal Logic— Syntax

An extension of propositional logic that allows to specify properties of sets of runs

## Syntax

Based on propositional signature and syntax.

Extension with three connectives:

**Always** If $\phi$ is a formula then so is $\Box\phi$

**Sometimes** If $\phi$ is a formula then so is $\Diamond\phi$

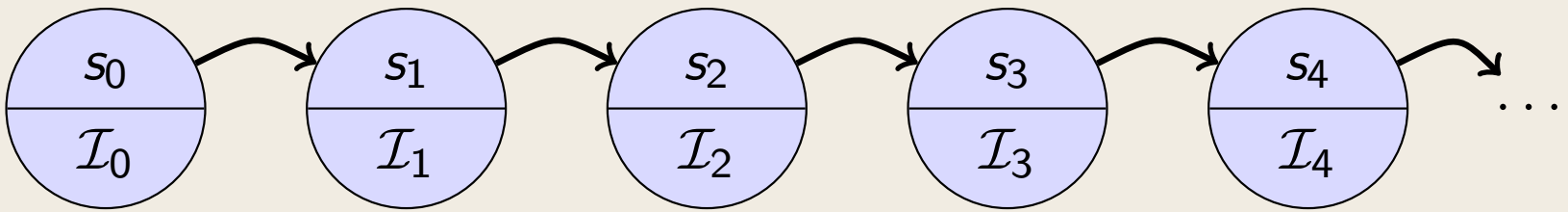
**Until** If $\phi$ and $\psi$ are formulas then so is $\phi\mathcal{U}\psi$

## Concrete Syntax

| | text book | SPIN |
|---|---|---|
| Always | $\Box$ | [] |
| Sometimes | $\Diamond$ | <> |
| Until | $\mathcal{U}$ | U |

# Semantics of Temporal Logic

**A run $\sigma$ is an infinite chain of states**

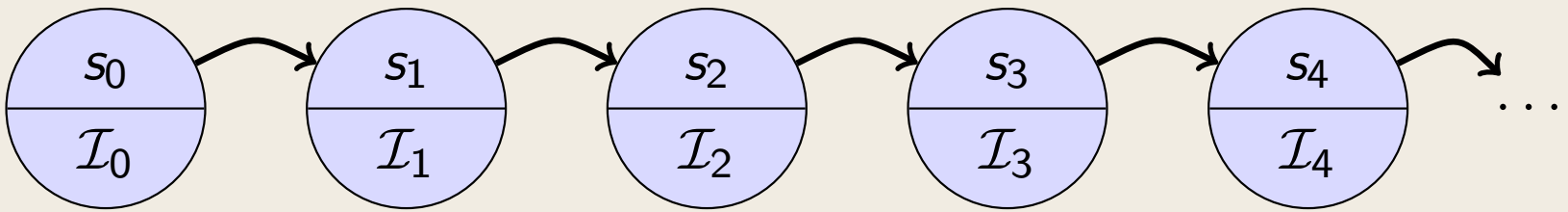



$\mathcal{I}_j$ propositional interpretation of variables in $j$-th state

Write more compactly $s_0\, s_1\, s_2\, s_3 \ldots$

# Semantics of Temporal Logic

**A run $\sigma$ is an infinite chain of states**




$\mathcal{I}_j$ propositional interpretation of variables in $j$-th state
Write more compactly $s_0\, s_1\, s_2\, s_3 \ldots$

If $\sigma = s_0\, s_1 \ldots$, then $\sigma|_i$ denotes the suffix $s_i\, s_{i+1} \ldots$ of $\sigma$.

# Semantics of Temporal Logic (Cont'd)

**Definition (Validity Relation)**

Validity of temporal formula depends on runs $\sigma = s_0\, s_1 \ldots$ for which the formula may, or may not, hold:

$\sigma \models p$ iff $\mathcal{I}_0(p) = T$, for $p \in \mathcal{P}$.

# Semantics of Temporal Logic (Cont'd)

**Definition (Validity Relation)**

Validity of temporal formula depends on runs $\sigma = s_0\, s_1 \ldots$ for which the formula may, or may not, hold:

$\sigma \models p$ iff $\mathcal{I}_0(p) = T$, for $p \in \mathcal{P}$.

$\sigma \models \neg\phi$ iff not $\sigma \models \phi$ (write $\sigma \not\models \phi$)

# Semantics of Temporal Logic (Cont'd)

**Definition (Validity Relation)**

Validity of temporal formula depends on runs $\sigma = s_0\, s_1 \ldots$ for which the formula may, or may not, hold:

$\sigma \models p$ iff $\mathcal{I}_0(p) = T$, for $p \in \mathcal{P}$.

$\sigma \models \neg\phi$ iff not $\sigma \models \phi$ (write $\sigma \not\models \phi$)

$\sigma \models \phi \wedge \psi$ iff $\sigma \models \phi$ and $\sigma \models \psi$

# Semantics of Temporal Logic (Cont'd)

**Definition (Validity Relation)**

Validity of temporal formula depends on runs $\sigma = s_0\, s_1 \ldots$ for which the formula may, or may not, hold:

$\sigma \models p$ iff $\mathcal{I}_0(p) = T$, for $p \in \mathcal{P}$.

$\sigma \models \neg\phi$ iff not $\sigma \models \phi$ (write $\sigma \not\models \phi$)

$\sigma \models \phi \wedge \psi$ iff $\sigma \models \phi$ and $\sigma \models \psi$

$\sigma \models \phi \vee \psi$ iff $\sigma \models \phi$ or $\sigma \models \psi$

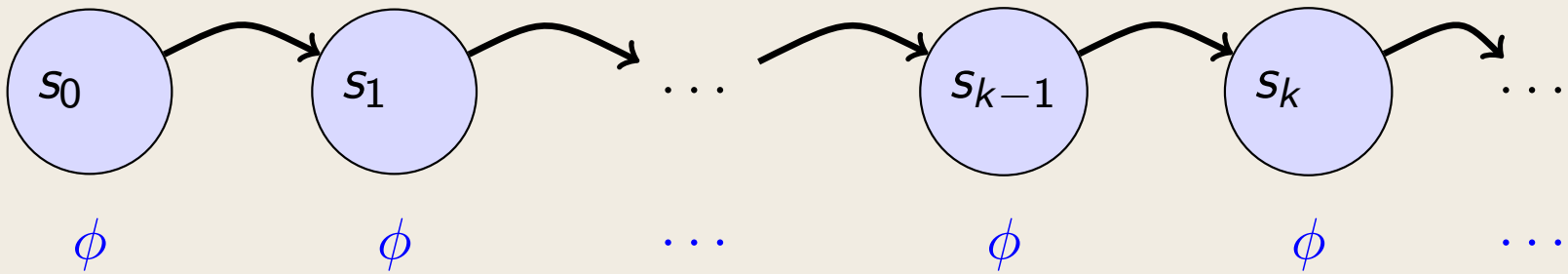
$\sigma \models \phi \rightarrow \psi$ iff $\sigma \not\models \phi$ or $\sigma \models \psi$

# Semantics of Temporal Logic Cont'd


**Definition (Validity Relation for Temporal Connectives)**

Given a run $\sigma = s_0\, s_1 \ldots$

# Semantics of Temporal Logic Cont'd

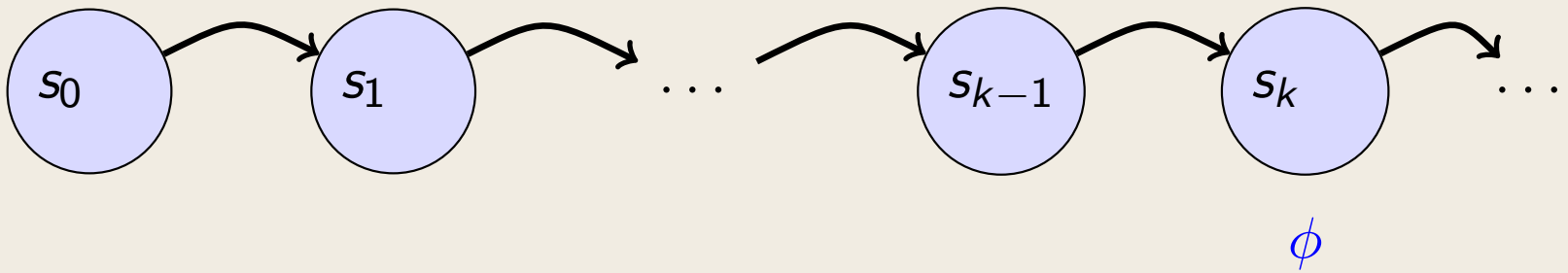



**Definition (Validity Relation for Temporal Connectives)**

Given a run $\sigma = s_0\, s_1 \ldots$

$\sigma \models \Box\phi$ iff $\sigma|_k \models \phi$ for all $k \geq 0$

# Semantics of Temporal Logic Cont'd



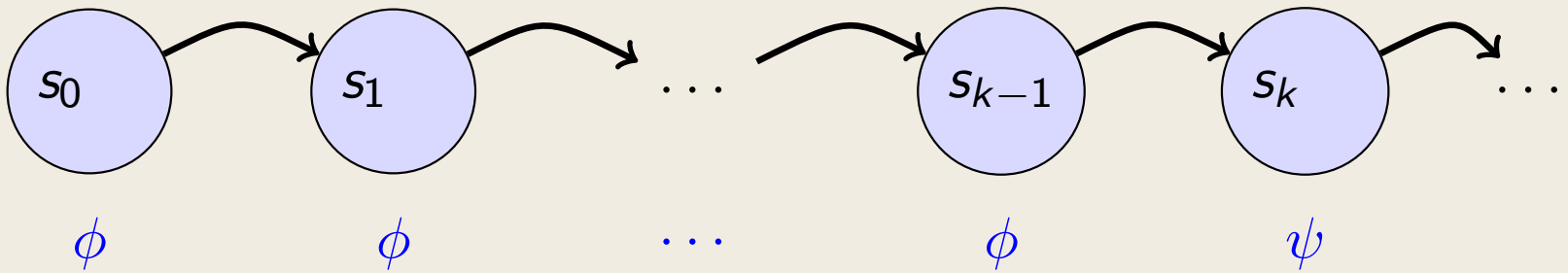

**Definition (Validity Relation for Temporal Connectives)**

Given a run $\sigma = s_0\, s_1 \ldots$

$\sigma \models \Box\phi$ iff $\sigma|_k \models \phi$ for all $k \geq 0$

$\sigma \models \Diamond\phi$ iff $\sigma|_k \models \phi$ for some $k \geq 0$

# Semantics of Temporal Logic Cont'd




**Definition (Validity Relation for Temporal Connectives)**

Given a run $\sigma = s_0\, s_1 \ldots$

$\sigma \models \Box\phi$ iff $\sigma|_k \models \phi$ for all $k \geq 0$

$\sigma \models \Diamond\phi$ iff $\sigma|_k \models \phi$ for some $k \geq 0$

$\sigma \models \phi\,\mathcal{U}\,\psi$ iff $\sigma|_k \models \psi$ for some $k \geq 0$, and $\sigma|_j \models \phi$ for all $0 \leq j < k$

# Safety and Liveness Properties

## Safety Properties

Always-formulas called safety property: something bad never happens

Let `mutex` be variable that is true when two process do not access a critical resource at the same time

$\Box$ `mutex` expresses that simultaneous access never happens

# Safety and Liveness Properties

## Safety Properties

Always-formulas called safety property: something bad never happens

Let `mutex` be variable that is true when two process do not access a critical resource at the same time

$\Box$ `mutex` expresses that simultaneous access never happens

## Liveness Properties

Sometimes-formulas called liveness property: something good happens eventually

Let `s` be variable that is true when a process delivers a service

$\Diamond$ `s` expresses that service is eventually provided

# Complex Properties

**What does this mean?**

$$\Box\Diamond\phi$$

# Complex Properties

**Infinitely Often**

$$\Box\Diamond\phi$$

During a run the formula $\phi$ will become true infinitely often.

# Validity Temporal Logic

**Definition (Validity)**

$\phi$ is valid, write $\models \phi$, iff $\phi$ is valid in all runs $\sigma = s_0\, s_1 \ldots$.

Recall that each run $s_0\, s_1 \ldots$ essentially is an infinite sequence of interpretations $\mathcal{I}_0\, \mathcal{I}_1 \ldots$.

# Examples

$$\Diamond\Box\phi$$

Valid?

# Examples

$$\Diamond\Box\phi$$

Valid?

No, there is a run in where it is not valid:

# Examples

$$\Diamond\Box\phi$$

Valid?

No, there is a run in where it is not valid:
$(\neg\phi, \neg\phi, \neg\phi, \ldots)$

# Examples

$$\Diamond\Box\phi$$

Valid?

No, there is a run in where it is not valid:
$(\neg\phi, \neg\phi, \neg\phi, \ldots)$

Valid in some run?

# Examples

$$\Diamond\Box\phi$$

Valid?

No, there is a run in where it is not valid:
$(\neg\phi, \neg\phi, \neg\phi, \ldots)$

Valid in some run?

Yes: $(\phi, \phi, \phi, \ldots)$

# Examples

$$\Diamond\Box\phi$$

Valid?

No, there is a run in where it is not valid:
$(\neg\phi, \neg\phi, \neg\phi, \ldots)$

Valid in some run?

Yes: $(\phi, \phi, \phi, \ldots)$

$$\Box\phi \rightarrow \phi \qquad\qquad (\neg\Box\phi) \leftrightarrow (\Diamond\neg\phi)$$

Both are valid!

# Examples

$$\Diamond\Box\phi$$

Valid?

No, there is a run in where it is not valid:
$(\neg\phi, \neg\phi, \neg\phi, \ldots)$

Valid in some run?

Yes: $(\phi, \phi, \phi, \ldots)$

$$\Box\phi \rightarrow \phi \qquad\qquad (\neg\Box\phi) \leftrightarrow (\Diamond\neg\phi)$$

Both are valid!

- $\Box$ is reflexive
- $\Box$ and $\Diamond$ are dual connectives

# Transition Systems Revisited

**Definition (Transition System)**

A Transition System $\mathcal{T} = (S, \mathit{Ini}, \delta, \mathcal{I})$ is given by a set of states $S$, a non-empty subset $\mathit{Ini} \subseteq S$ of initial states, and a transition relation $\delta \subseteq S \times S$, and $\mathcal{I}$ labeling each state $s \in S$ with a propositional interpretation $\mathcal{I}_s$.

**Definition (Runs of Transition System)**

A run of $\mathcal{T}$ is a run $\sigma = s_0\, s_1 \ldots$, with $s_i \in S$, such that $s_0 \in \mathit{Ini}$ and $(s_i, s_{i+1}) \in \delta$ for all $i$.

# Semantics of Temporal Logic (Cont'd)

Validity of temporal formula is extended to transition systems in the following way:

**Definition (Validity Relation)**

Given a transition systems $\mathcal{T} = (S, \mathit{Ini}, \delta, \mathcal{I})$, a temporal formula $\phi$ is valid in $\mathcal{T}$ (write $\mathcal{T} \models \phi$) iff $\sigma \models \phi$ for all runs $\sigma$ of $\mathcal{T}$.

# $\omega$-Languages

Given a finite alphabet (vocabulary) $\Sigma$.

A word $w \in \Sigma^*$ is a finite sequence

$$w = a_o \cdots a_n$$

with $a_i \in \Sigma, i \in \{0, \ldots, n\}$.

$\mathcal{L} \subseteq \Sigma^*$ is called a language.

# $\omega$-Languages

Given a finite alphabet (vocabulary) $\Sigma$.

An $\omega$ word $w \in \Sigma^\omega$ is an infinite sequence

$$w = a_o \cdots a_k \cdots$$

with $a_i \in \Sigma, i \in \mathbb{N}$.

$\mathcal{L}^\omega \subseteq \Sigma^\omega$ is called an $\omega$-language.

# Büchi Automaton

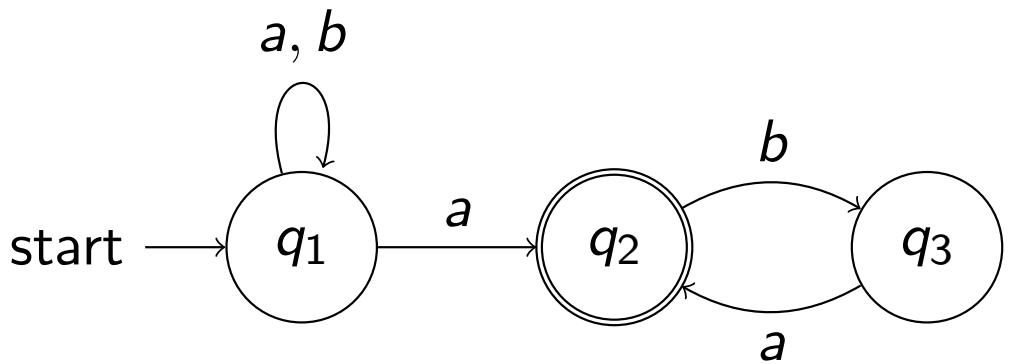
**Definition (Büchi Automaton)**

A (non-deterministic) Büchi automaton over an alphabet $\Sigma$ consists of a

- finite, non-empty set of locations $Q$
- a non-empty set of initial/start locations $I \subseteq Q$
- a set of accepting locations $F = \{F_1, \ldots, F_n\} \subseteq Q$
- a transition relation $\delta \subseteq Q \times \Sigma \times Q$

**Example**

$\Sigma = \{a, b\}, Q = \{q_1, q_2, q_3\}, I = \{q_1\}, F = \{q_2\}$



# Büchi Automaton – Acceptance

**Definition (Run and Accepted Run)**

An infinite word $w = a_o \ldots a_k \ldots \in \Sigma^\omega$ is a run of a Büchi automaton if

$$q_{i+1} \in \delta(q_i, a_i)$$

for some initial location $q_0 \in I$.
A Büchi automaton accepts a run $w \in \Sigma^\omega$, if some accepting location $f \in F$ is infinitely often visited (i.e., $\exists$ infinite set $R \subseteq \mathbb{N}$: $f \in \delta(q_i, a_i), \forall i \in R$)

# Büchi Automaton – Acceptance

**Definition (Run and Accepted Run)**

An infinite word $w = a_o \dots a_k \dots \in \Sigma^\omega$ is a run of a Büchi automaton if

$$q_{i+1} \in \delta(q_i, a_i)$$

for some initial location $q_0 \in I$.
A Büchi automaton accepts a run $w \in \Sigma^\omega$, if some accepting location $f \in F$ is infinitely often visited (i.e., $\exists$ infinite set $R \subseteq \mathbb{N}$: $f \in \delta(q_i, a_i), \forall i \in R$)

Let $\mathcal{B}$ be a Büchi automaton, then

$$\mathcal{L}^\omega(\mathcal{B}) = \{w \in \Sigma^\omega | w \in \Sigma^\omega \text{ is an accepted run of } \mathcal{B}\}$$

denotes the $\omega$-language recognised by $\mathcal{B}$.

# Büchi Automaton – Acceptance

**Definition (Run and Accepted Run)**

An infinite word $w = a_o \dots a_k \dots \in \Sigma^\omega$ is a run of a Büchi automaton if

$$q_{i+1} \in \delta(q_i, a_i)$$

for some initial location $q_0 \in I$.
A Büchi automaton accepts a run $w \in \Sigma^\omega$, if some accepting location $f \in F$ is infinitely often visited (i.e., $\exists$ infinite set $R \subseteq \mathbb{N}$: $f \in \delta(q_i, a_i), \forall i \in R$)

Let $\mathcal{B}$ be a Büchi automaton, then

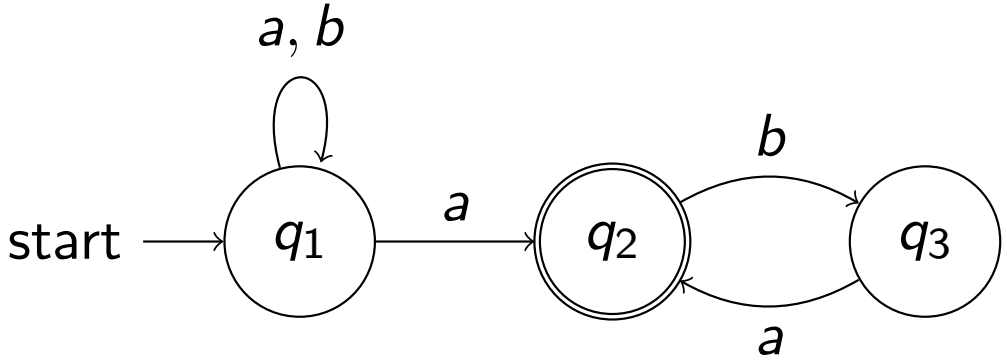
$$\mathcal{L}^\omega(\mathcal{B}) = \{w \in \Sigma^\omega | w \in \Sigma^\omega \text{ is an accepted run of } \mathcal{B}\}$$

denotes the $\omega$-language recognised by $\mathcal{B}$.

An $\omega$-language for which an accepting Büchi automaton exists is called $\omega$-regular language.

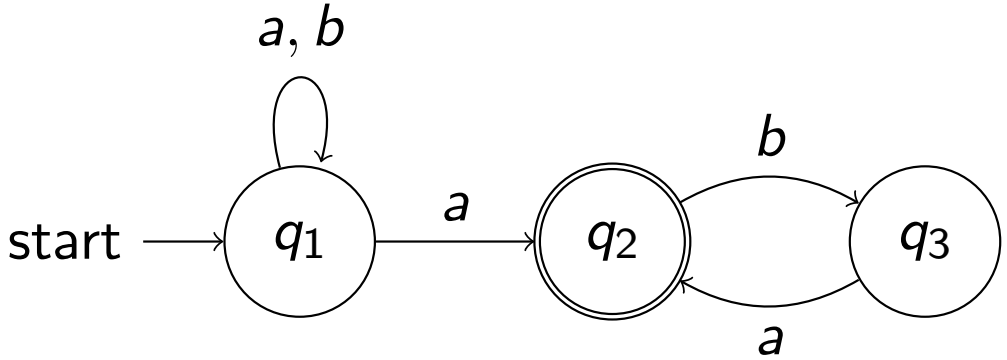
# Examples

Which language is accepted by the following Büchi automaton?



# Examples

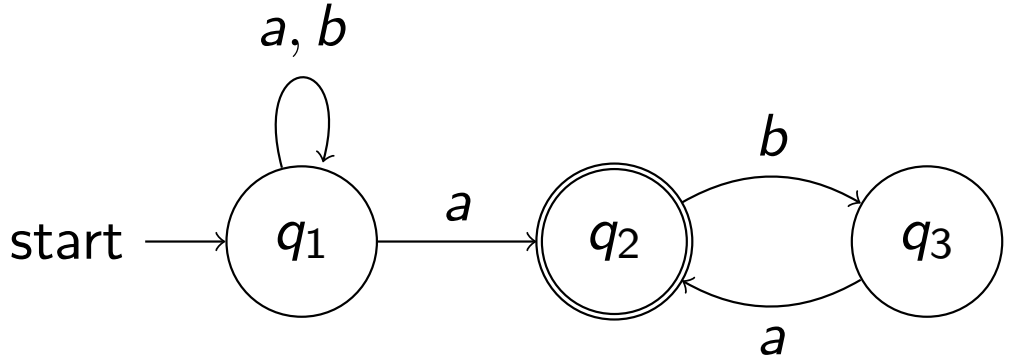
Which language is accepted by the following Büchi automaton?




Solution: $(a + b)^*(ab)^\omega$

# Examples

Which language is accepted by the following Büchi automaton?




Solution: $(a + b)^*(ab)^\omega$

$\omega$-regular expressions like standard regular expression

| | | |
|---|---|---|
| | $ab$ | $a$ **then** $b$ |
| | $a + b$ | $a$ **or** $b$ |
| | $a^*$ | arbitrarily, but finitely often $a$ |
| **new:** | $a^\omega$ | infinitely often $a$ |

# Decidability, Closure Properties

Many properties for regular finite automata hold also for Büchi automata.

**Theorem (Decidability)**

*It is decidable whether the accepted language $\mathcal{L}^{\omega}(\mathcal{B})$ of a Büchi automaton $\mathcal{B}$ is empty or not.*

# Decidability, Closure Properties

Many properties for regular finite automata hold also for Büchi automata.

**Theorem (Decidability)**

*It is decidable whether the accepted language $\mathcal{L}^\omega(\mathcal{B})$ of a Büchi automaton $\mathcal{B}$ is empty or not.*

**Theorem (Closure properties)**

*The set of $\omega$-regular languages is closed with respect to intersection, union and complement, i.e.,*

- *$\mathcal{L}_1, \mathcal{L}_2$ are $\omega$-reg. then $\mathcal{L}_1 \cap \mathcal{L}_2$ and $\mathcal{L}_1 \cup \mathcal{L}_2$ are $\omega$-reg.*
- *$\mathcal{L}$ is $\omega$-reg. then $\Sigma^\omega \backslash \mathcal{V}^\omega$ is $\omega$-reg.*

# Decidability, Closure Properties

Many properties for regular finite automata hold also for Büchi automata.

**Theorem (Decidability)**

*It is decidable whether the accepted language $\mathcal{L}^\omega(\mathcal{B})$ of a Büchi automaton $\mathcal{B}$ is empty or not.*

**Theorem (Closure properties)**

*The set of $\omega$-regular languages is closed with respect to intersection, union and complement, i.e.,*

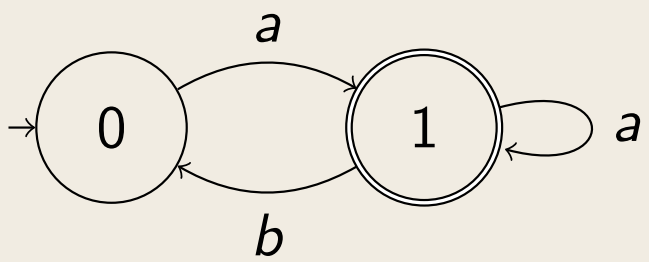
- *$\mathcal{L}_1, \mathcal{L}_2$ are $\omega$-reg. then $\mathcal{L}_1 \cap \mathcal{L}_2$ and $\mathcal{L}_1 \cup \mathcal{L}_2$ are $\omega$-reg.*
- *$\mathcal{L}$ is $\omega$-reg. then $\Sigma^\omega \backslash \mathcal{V}^\omega$ is $\omega$-reg.*

But in contrast to regular finite automata:

Non-deterministic Büchi automata are strictly more expressive than deterministic ones.

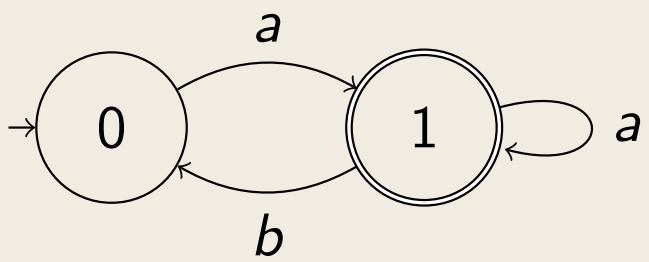
# Examples

## Language:



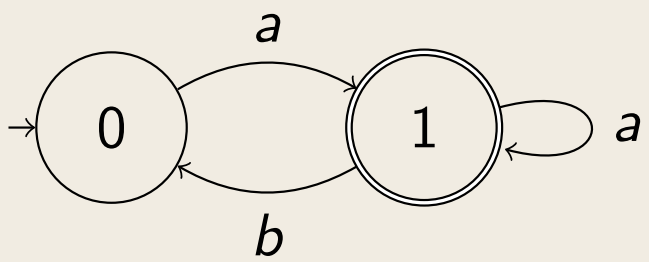
# Examples

**Language:** $a(a + ba)^\omega$

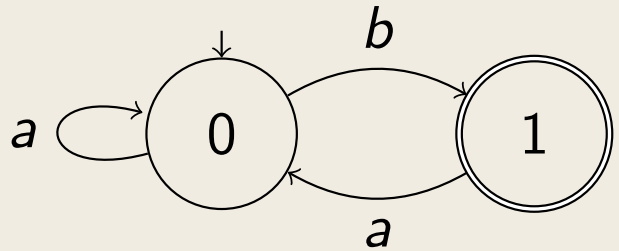


# Examples

**Language:** $a(a + ba)^\omega$

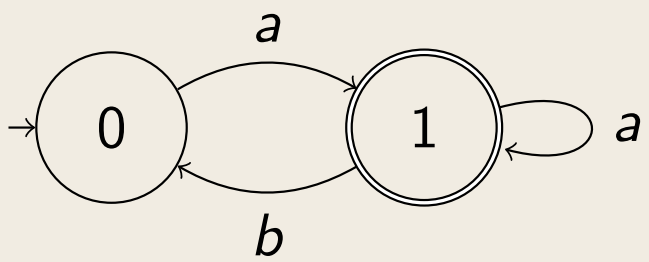



**Language:**

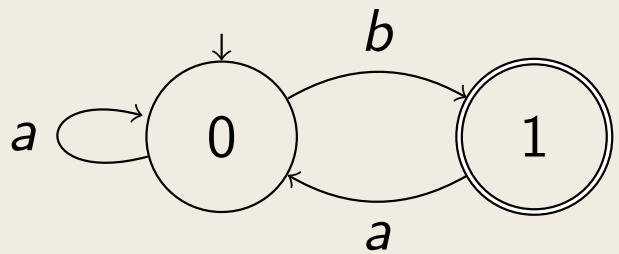


# Examples

**Language:** $a(a + ba)^\omega$




**Language:** $(a^* ba)^\omega$



# Linear Temporal Logic and Büchi Automata

LTL and Büchi Automata are connected.
Short reminder

**Definition (Validity Relation)**

Given a transition systems $\mathcal{T} = (S, Ini, \delta, \mathcal{I})$, a temporal formula $\phi$ is valid in $\mathcal{T}$ (write $\mathcal{T} \models \phi$) iff $\sigma \models \phi$ for all runs $\sigma$ of $\mathcal{T}$.

A run of the transition system is an infinite sequence of interpretations $I$.

# Linear Temporal Logic and Büchi Automata

LTL and Büchi Automata are connected.

Short reminder

**Definition (Validity Relation)**

Given a transition systems $\mathcal{T} = (S, Ini, \delta, \mathcal{I})$, a temporal formula $\phi$ is valid in $\mathcal{T}$ (write $\mathcal{T} \models \phi$) iff $\sigma \models \phi$ for all runs $\sigma$ of $\mathcal{T}$.

A run of the transition system is an infinite sequence of interpretations $I$.

Goal

Given an LTL formula $\phi$.

# Linear Temporal Logic and Büchi Automata

LTL and Büchi Automata are connected.

Short reminder

**Definition (Validity Relation)**

Given a transition systems $\mathcal{T} = (S, Ini, \delta, \mathcal{I})$, a temporal formula $\phi$ is valid in $\mathcal{T}$ (write $\mathcal{T} \models \phi$) iff $\sigma \models \phi$ for all runs $\sigma$ of $\mathcal{T}$.

A run of the transition system is an infinite sequence of interpretations $I$.

Goal

Given an LTL formula $\phi$.

Construct a Büchi automaton accepting exactly those runs (infinite sequences of interpretations) that satisfy $\phi$ .

# Encoding an LTL Formula as Büchi Automaton

$\mathcal{P}$ set of propositional variables, e.g. $\mathcal{P} = \{r, s\}$

Alphabet $\Sigma$

# Encoding an LTL Formula as Büchi Automaton

$\mathcal{P}$ set of propositional variables, e.g. $\mathcal{P} = \{r, s\}$

Alphabet $\Sigma$

Define $\Sigma$ as set of all interpretations over $\mathcal{P}$ (i.e. $\Sigma := 2^{\mathcal{P}}$)

# Encoding an LTL Formula as Büchi Automaton

$\mathcal{P}$ set of propositional variables, e.g. $\mathcal{P} = \{r, s\}$

Alphabet $\Sigma$

Define $\Sigma$ as set of all interpretations over $\mathcal{P}$ (i.e. $\Sigma := 2^{\mathcal{P}}$)

E.g. $\Sigma = \{\emptyset, \{r\}, \{s\}, \{r, s\}\}$

Each element of $\Sigma$ corresponds exactly to one propositional interpretation over $\mathcal{P}$, e.g.,

$$I_{\emptyset}(r) = F, I_{\emptyset}(s) = F, I_{\{r\}}(r) = T, I_{\{r\}}(s) = F, \ldots$$

# Encoding an LTL Formula as Büchi Automaton

$\mathcal{P}$ set of propositional variables, e.g. $\mathcal{P} = \{r, s\}$

Alphabet $\Sigma$

Define $\Sigma$ as set of all interpretations over $\mathcal{P}$ (i.e. $\Sigma := 2^{\mathcal{P}}$)

E.g. $\Sigma = \{\emptyset, \{r\}, \{s\}, \{r, s\}\}$

Each element of $\Sigma$ corresponds exactly to one propositional interpretation over $\mathcal{P}$, e.g.,

$$I_{\emptyset}(r) = F, I_{\emptyset}(s) = F, I_{\{r\}}(r) = T, I_{\{r\}}(s) = F, \ldots$$

Using this alphabet we can now construct a Büchi automaton for a formula $\phi$ over $\mathcal{P}$

# Examples – Büchi Automaton for LTL Formulas

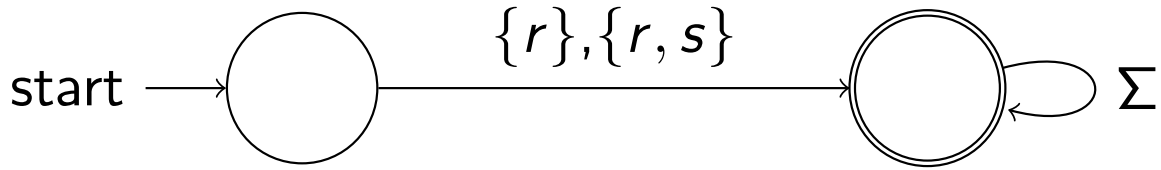
**Example (Büchi automaton for formula $r$)**

Give a Büchi automaton $\mathcal{B}$ accepting exactly those runs $\sigma$ satisfying $r$

# Examples – Büchi Automaton for LTL Formulas

**Example (Büchi automaton for formula $r$)**

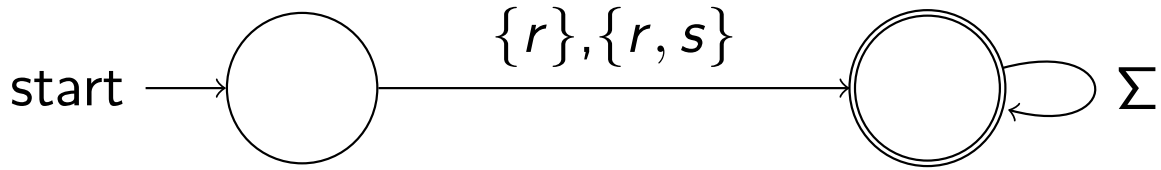
Give a Büchi automaton $\mathcal{B}$ accepting exactly those runs $\sigma$ satisfying $r$



# Examples – Büchi Automaton for LTL Formulas

**Example (Büchi automaton for formula $r$)**

Give a Büchi automaton $\mathcal{B}$ accepting exactly those runs $\sigma$ satisfying $r$

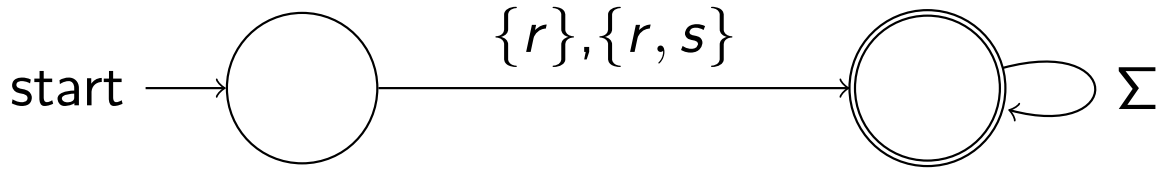



**Example (Büchi automaton for formula $\Box r$)**

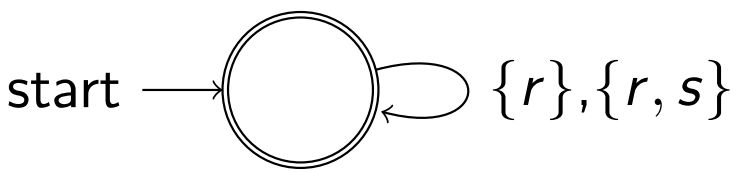
# Examples – Büchi Automaton for LTL Formulas

**Example (Büchi automaton for formula $r$)**

Give a Büchi automaton $\mathcal{B}$ accepting exactly those runs $\sigma$ satisfying $r$



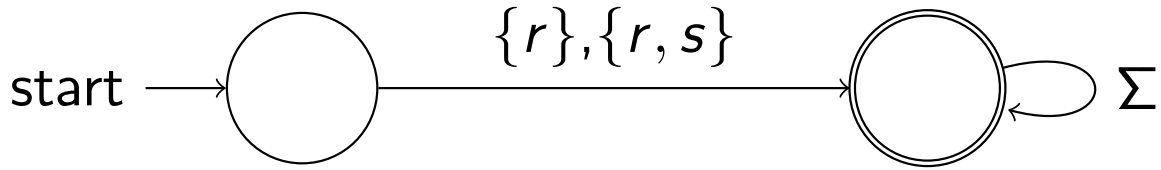

**Example (Büchi automaton for formula $\Box r$)**



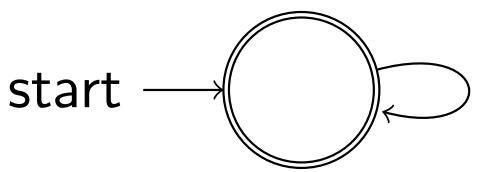
# Examples – Büchi Automaton for LTL Formulas

**Example (Büchi automaton for formula $r$)**

Give a Büchi automaton $\mathcal{B}$ accepting exactly those runs $\sigma$ satisfying $r$




**Example (Büchi automaton for formula $\Box r$)**



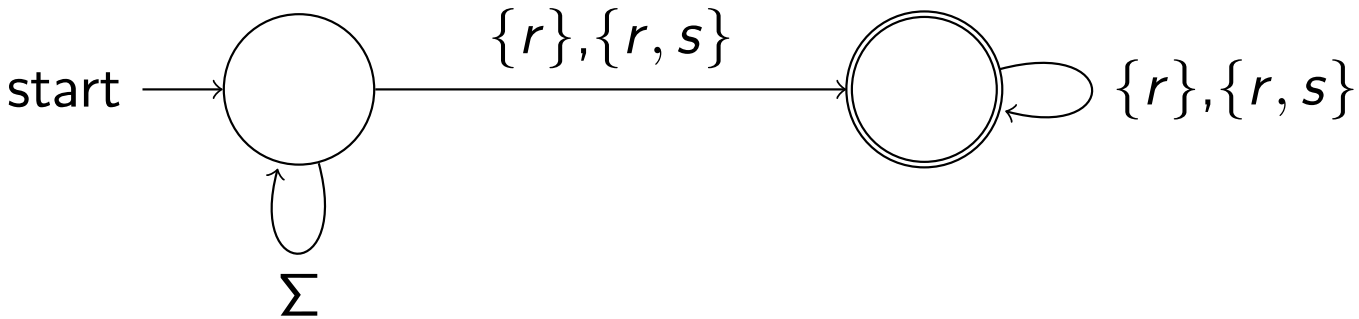

$(R := \{I | I \in \Sigma, r \in I\})$

# Examples – Büchi Automaton for LTL Formulas

**Example (Büchi automaton for formula $\Diamond\Box r$)**

# Examples – Büchi Automaton for LTL Formulas

**Example (Büchi automaton for formula $\Diamond\Box r$)**



# Model Checking

How can we now check if a formula is valid in all runs of a transition system?

Given a transition system $\mathcal{T}$ (e.g., derived from a PROMELA program).

Verification task: Is the LTL formula $\phi$ satisfied in all runs of $\mathcal{T}$, i.e.,

$$\mathcal{T} \models \phi \quad ?$$

# Model Checking

How can we now check if a formula is valid in all runs of a transition system?

Given a transition system $\mathcal{T}$ (e.g., derived from a PROMELA program).

Verification task: Is the LTL formula $\phi$ satisfied in all runs of $\mathcal{T}$, i.e.,

$$\mathcal{T} \models \phi \quad ?$$

Temporal model checking with SPIN: Topic of next lecture.

# Model Checking

How can we now check if a formula is valid in all runs of a transition system?

Given a transition system $\mathcal{T}$ (e.g., derived from a PROMELA program).

Verification task: Is the LTL formula $\phi$ satisfied in all runs of $\mathcal{T}$, i.e.,

$$\mathcal{T} \models \phi \quad ?$$

Temporal model checking with SPIN: Topic of next lecture.

Today: Basic principle behind model checking only.

# Model Checking – Overview

$$\mathcal{T} \models \phi \quad ?$$

1. Represent transition system $\mathcal{T}$ as Büchi automaton $\mathcal{B}_{\mathcal{T}}$ such that $\mathcal{B}_{\mathcal{T}}$ accepts exactly those words corresponding to runs through $\mathcal{T}$

# Model Checking – Overview

$$\mathcal{T} \models \phi \quad ?$$

1. Represent transition system $\mathcal{T}$ as Büchi automaton $\mathcal{B}_{\mathcal{T}}$ such that $\mathcal{B}_{\mathcal{T}}$ accepts exactly those words corresponding to runs through $\mathcal{T}$
2. Construct Büchi automaton $\mathcal{B}_{\neg\phi}$ for negation of formula $\phi$

# Model Checking – Overview

$$\mathcal{T} \models \phi \quad ?$$

1. Represent transition system $\mathcal{T}$ as Büchi automaton $\mathcal{B}_{\mathcal{T}}$ such that $\mathcal{B}_{\mathcal{T}}$ accepts exactly those words corresponding to runs through $\mathcal{T}$
2. Construct Büchi automaton $\mathcal{B}_{\neg\phi}$ for negation of formula $\phi$
3. If

$$\mathcal{L}^{\omega}(\mathcal{B}_{\mathcal{T}}) \cap \mathcal{L}^{\omega}(\mathcal{B}_{\neg\phi}) = \emptyset$$

then $\phi$ holds, otherwise we have a counterexample.

# Model Checking – Overview

$$\mathcal{T} \models \phi \quad ?$$

1. Represent transition system $\mathcal{T}$ as Büchi automaton $\mathcal{B}_\mathcal{T}$ such that $\mathcal{B}_\mathcal{T}$ accepts exactly those words corresponding to runs through $\mathcal{T}$
2. Construct Büchi automaton $\mathcal{B}_{\neg\phi}$ for negation of formula $\phi$
3. If
$$\mathcal{L}^\omega(\mathcal{B}_\mathcal{T}) \cap \mathcal{L}^\omega(\mathcal{B}_{\neg\phi}) = \emptyset$$
then $\phi$ holds, otherwise we have a counterexample.
To check $\mathcal{L}^\omega(\mathcal{B}_\mathcal{T}) \cap \mathcal{L}^\omega(\mathcal{B}_{\neg\phi})$ construct intersection automaton and search for cycle through accepting state.

# Representing a model as Büchi Automaton

First Step: Represent transition system $\mathcal{T}$ as Büchi automaton $\mathcal{B}_{\mathcal{T}}$ accepting exactly those words representing a run of $\mathcal{T}$.

Example

```
active proctype P () {
do
:: atomic {
     !wantQ;
     wantP = true
   }
   pcs = true;
   atomic {
     pcs = false;
     wantP = false
   }
od }
```

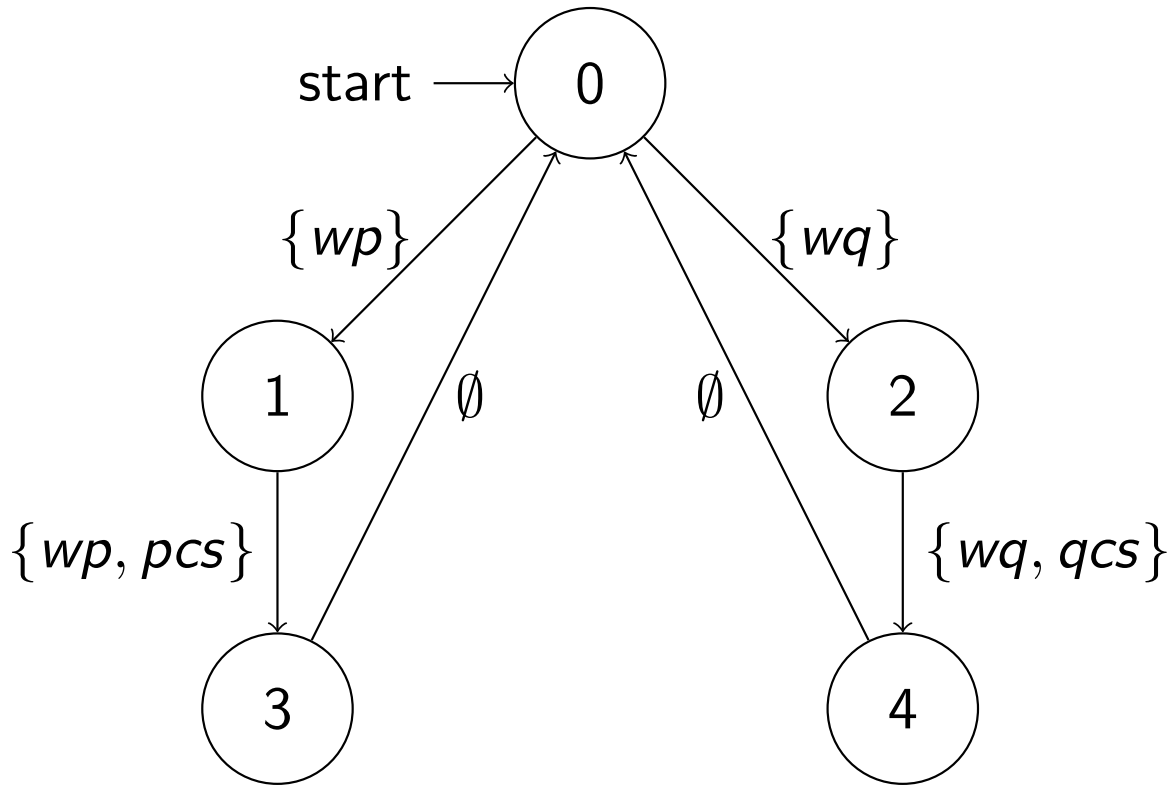
(the very first location skipped and second `atomic` only to keep the automaton small; similar for `Q`)

# Representing a model as Büchi Automaton

First Step: Represent transition system $\mathcal{T}$ as Büchi automaton $\mathcal{B}_{\mathcal{T}}$ accepting exactly those words representing a run of $\mathcal{T}$.

Example

```
active proctype P () {
do
:: atomic {
     !wantQ;
     wantP = true
   }
   pcs = true;
   atomic {
     pcs = false;
     wantP = false
   }
od }
```



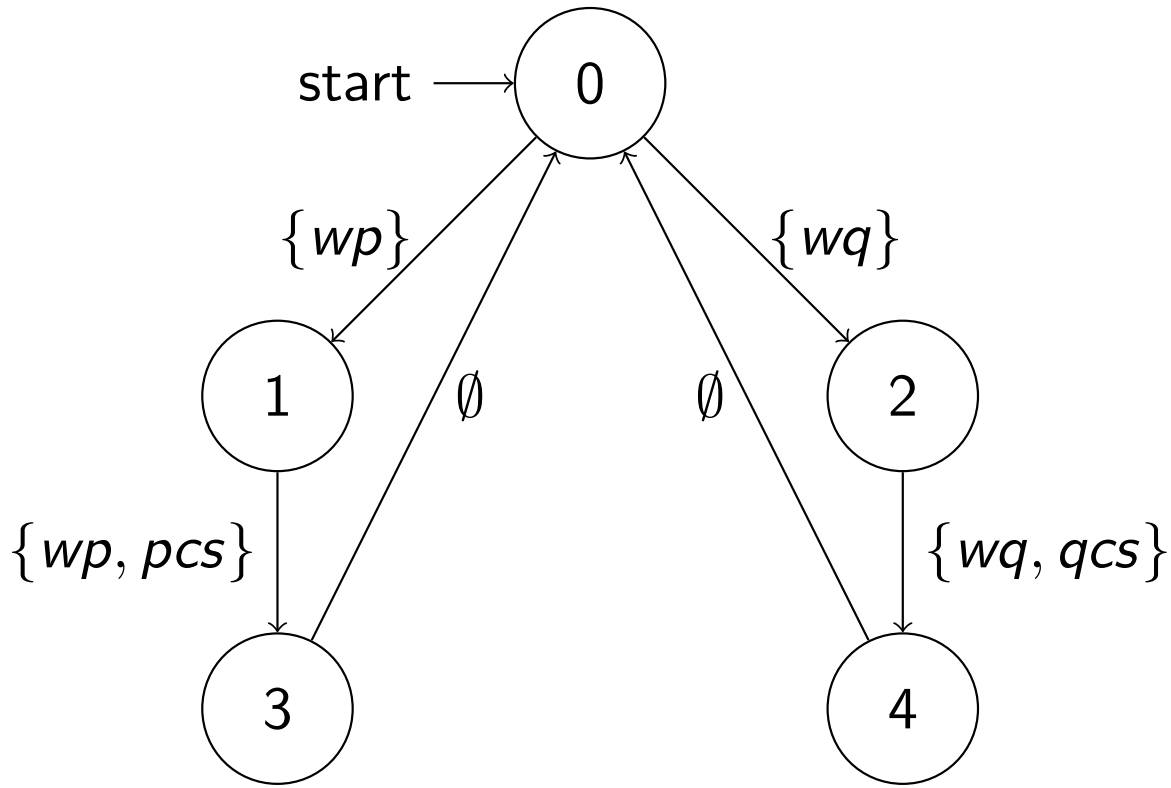

(the very first location skipped and second `atomic` only to keep the automaton small; similar for `Q`)

# Representing a model as Büchi Automaton

First Step: Represent transition system $\mathcal{T}$ as Büchi automaton $\mathcal{B}_{\mathcal{T}}$ accepting exactly those words representing a run of $\mathcal{T}$.

Example

```
active proctype P () {
do
:: atomic {
     !wantQ;
     wantP = true
   }
   pcs = true;
   atomic {
     pcs = false;
     wantP = false
   }
od }
```



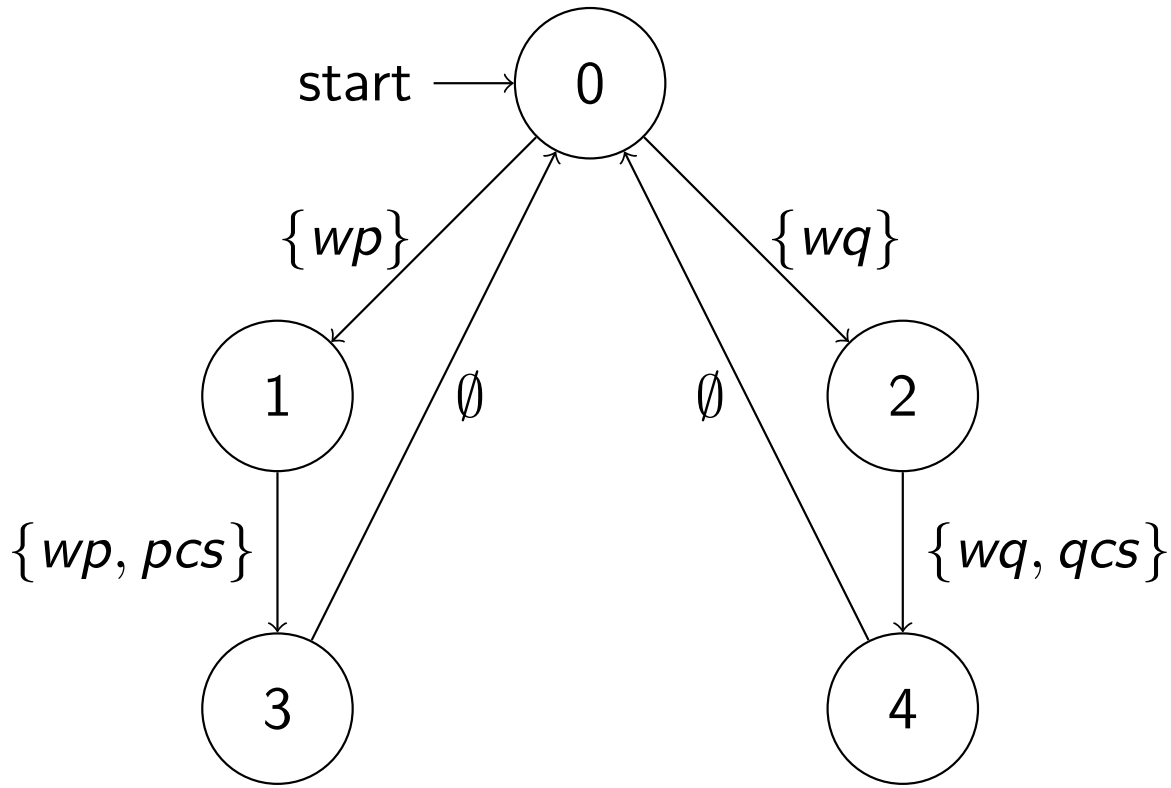

What are the accepting locations?

# Representing a model as Büchi Automaton

First Step: Represent transition system $\mathcal{T}$ as Büchi automaton $\mathcal{B}_{\mathcal{T}}$ accepting exactly those words representing a run of $\mathcal{T}$.

Example

```
active proctype P () {
do
:: atomic {
     !wantQ;
     wantP = true
   }
   pcs = true;
   atomic {
     pcs = false;
     wantP = false
   }
od }
```



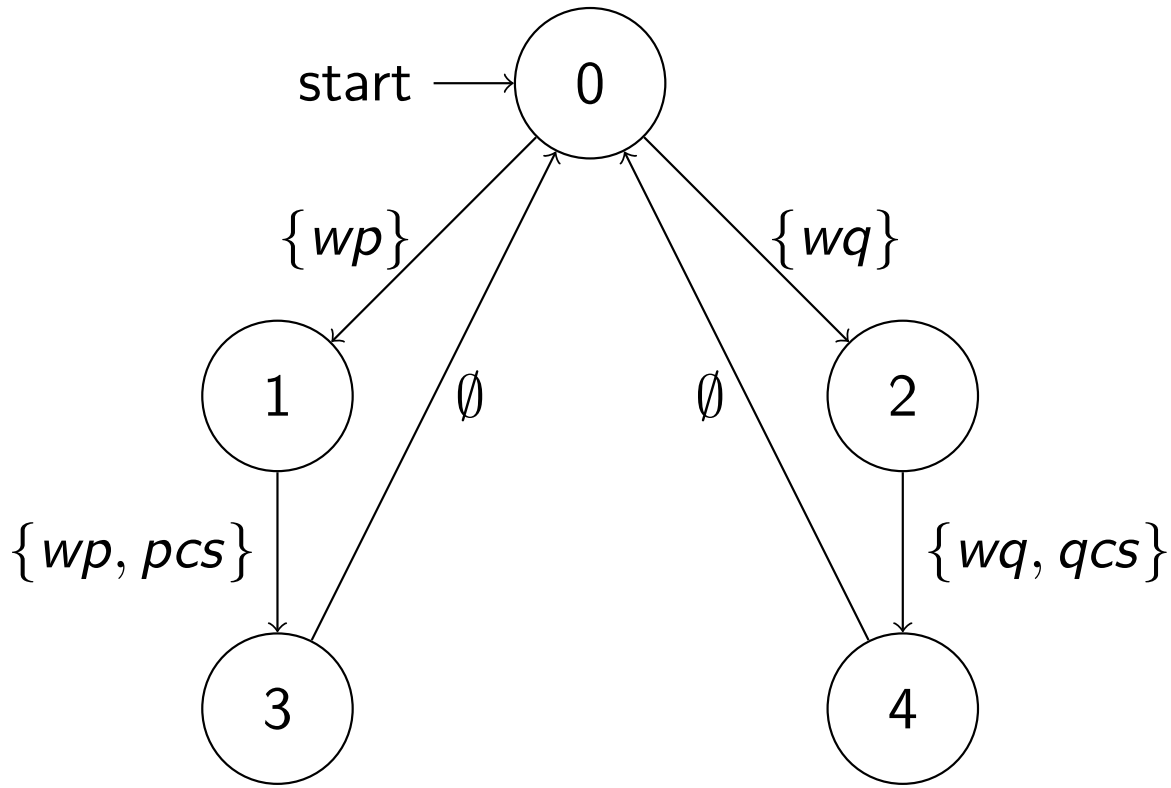

What are the accepting locations? All

# Representing a model as Büchi Automaton

First Step: Represent transition system $\mathcal{T}$ as Büchi automaton $\mathcal{B}_{\mathcal{T}}$ accepting exactly those words representing a run of $\mathcal{T}$.

Example

```
active proctype P () {
do
:: atomic {
     !wantQ;
     wantP = true
   }
   pcs = true;
   atomic {
     pcs = false;
     wantP = false
   }
od }
```




The property we want to check is $\phi = \Box\neg pcs$

# Büchi Automaton $B_{\neg\phi}$ for $\neg\phi$

$\mathcal{T} \models \phi$ holds iff. there is no accepting run for $\neg\phi$

$\neg\phi = \neg\Box\neg pcs = \Diamond pcs$

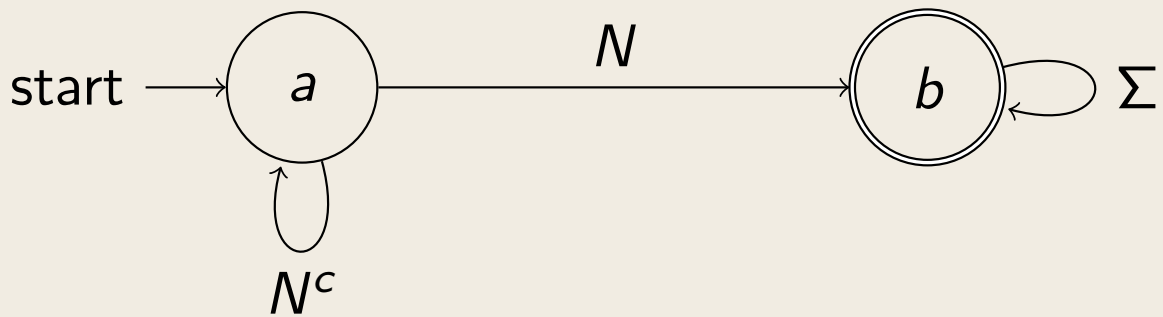
# Büchi Automaton $B_{\neg\phi}$ for $\neg\phi$

$\mathcal{T} \models \phi$ holds iff. there is no accepting run for $\neg\phi$

$\neg\phi = \neg\Box\neg pcs = \Diamond pcs$

## Büchi Automaton $\mathcal{B}_{\neg\phi}$

$$\mathcal{P} = \{wp, wq, pcs, qcs\}, \Sigma = 2^{\mathcal{P}}$$




$N = \{I | I \in \Sigma, pcs \in I\}, \quad N^c = \Sigma - N$

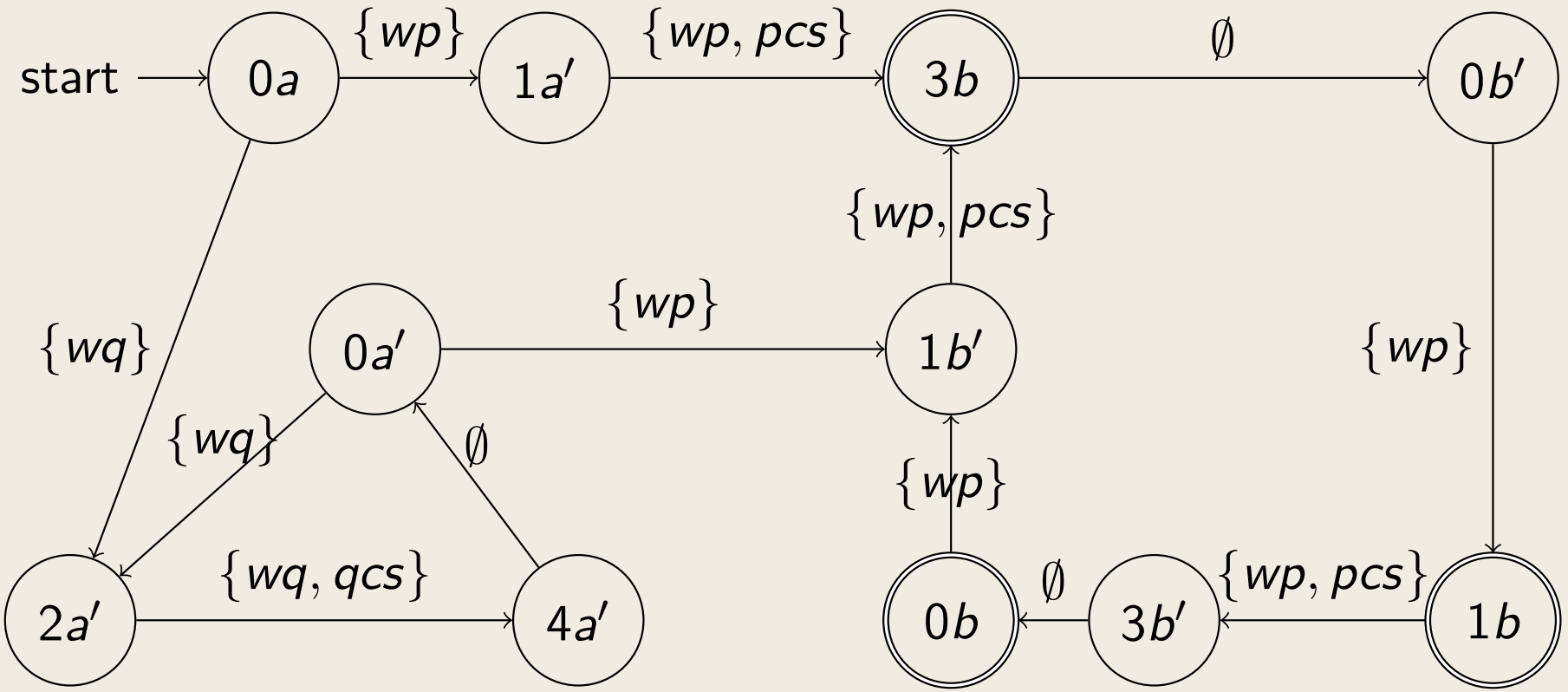
# Checking for the empty language

$$\mathcal{L}^{\omega}(\mathcal{B}_{\neg\phi}) \cap \mathcal{L}^{\omega}(\mathcal{B}_{\mathcal{T}}) = \emptyset \quad ?$$

# Checking for the empty language

$$\mathcal{L}^{\omega}(\mathcal{B}_{\neg\phi}) \cap \mathcal{L}^{\omega}(\mathcal{B}_{\mathcal{T}}) = \emptyset \quad ?$$

## Intersection Automaton

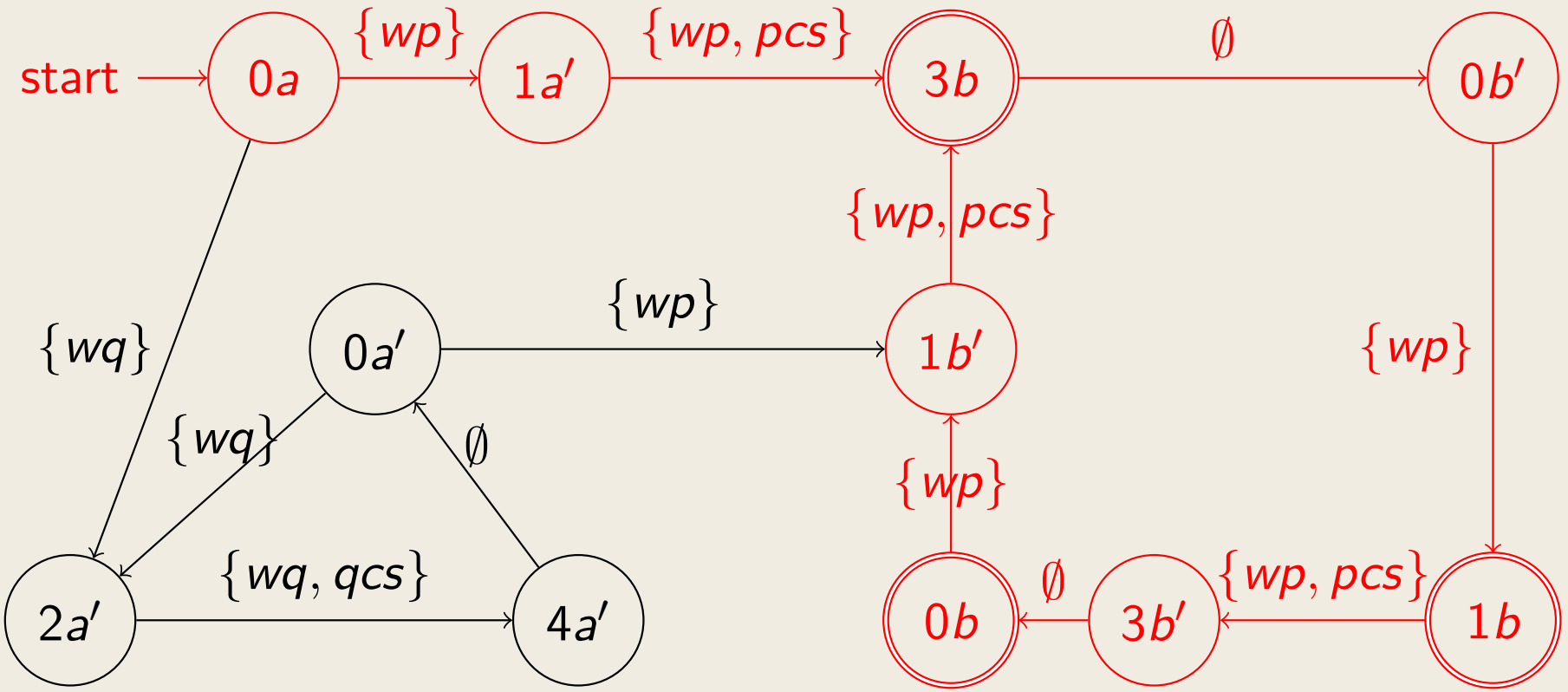


# Checking for the empty language

$$\mathcal{L}^{\omega}(\mathcal{B}_{\neg\phi}) \cap \mathcal{L}^{\omega}(\mathcal{B}_{\mathcal{T}}) \neq \emptyset$$

Counterexample

**Intersection Automaton**



# Literature for this Lecture

**Ben-Ari** Section 5.2.1
(PROMELA examples on the surface only)

**Baier and Katoen** Principles of Model Checking, ISBN: 026202649X, May, 2008, The MIT Press

# Intersection Automaton

–

# Construction

# Insersection Automaton (I)

Given: Two Büchi automata $\mathcal{B}_i = (Q_i, \delta_i, I_i, F_i)$

# Insersection Automaton (I)

Given: Two Büchi automata $\mathcal{B}_i = (Q_i, \delta_i, I_i, F_i)$

Wanted: A Büchi automaton

$$\mathcal{B}_{1\cap 2} = (Q_{1\cap 2}, \delta_{1\cap 2}, I_{1\cap 2}, F_{1\cap 2})$$

accepting a word $w$ iff. $w$ accepted by $\mathcal{B}_1$ and $\mathcal{B}_2$

# Insersection Automaton (I)

Given: Two Büchi automata $\mathcal{B}_i = (Q_i, \delta_i, I_i, F_i)$

Wanted: A Büchi automaton

$$\mathcal{B}_{1\cap 2} = (Q_{1\cap 2}, \delta_{1\cap 2}, I_{1\cap 2}, F_{1\cap 2})$$

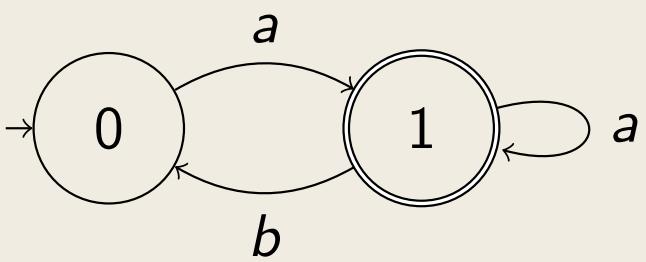
accepting a word $w$ iff. $w$ accepted by $\mathcal{B}_1$ and $\mathcal{B}_2$

Maybe just the productautomaton as for regular automata?

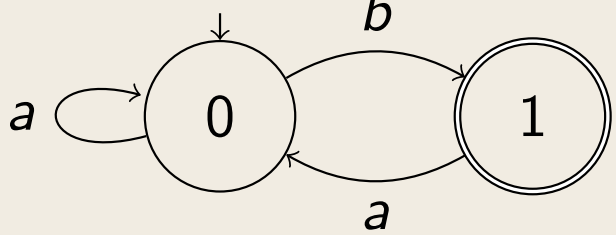
# Productautomata as Intersection Automata

$\Sigma = \{a, b\}$

$a(a + ba)^\omega$ :



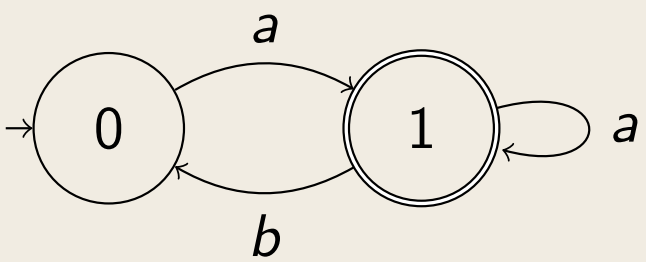

$(a^* ba)^\omega$ :



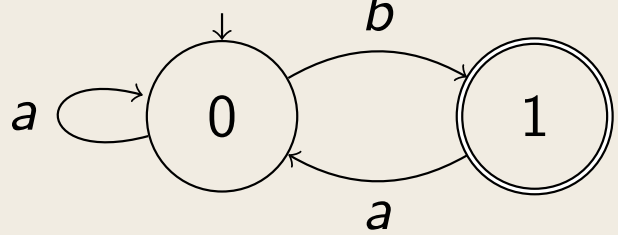
# Productautomata as Intersection Automata

$\Sigma = \{a, b\}$, $a(a + ba)^\omega \cap (a^* ba)^\omega = \emptyset$?

$a(a + ba)^\omega$ :



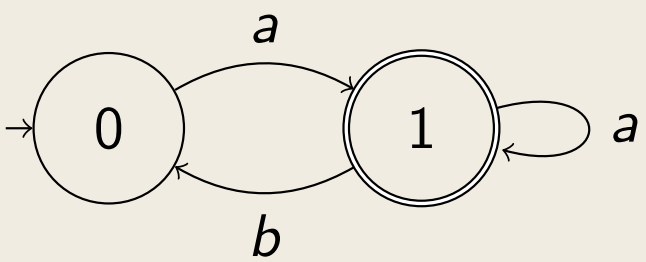

$(a^* ba)^\omega$ :



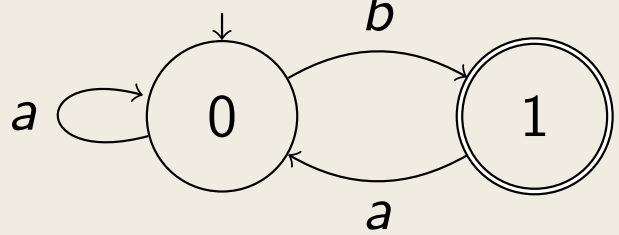
# Productautomata as Intersection Automata

$\Sigma = \{a, b\}$, $a(a + ba)^\omega \cap (a^* ba)^\omega = \emptyset$? No, e.g. $a(ba)^\omega$

$a(a + ba)^\omega$ :



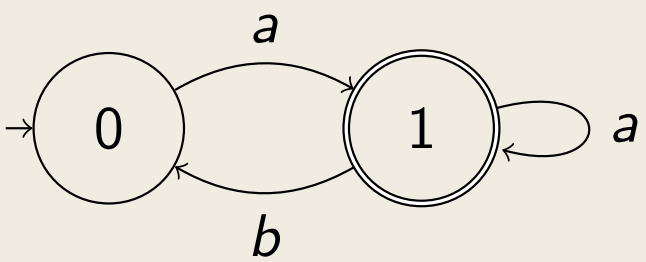

$(a^* ba)^\omega$ :



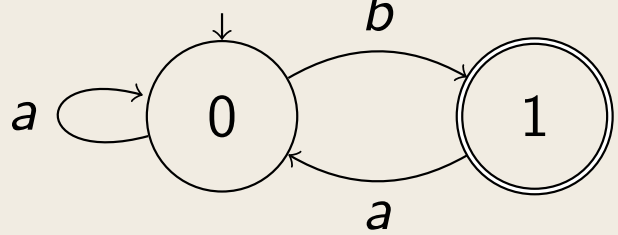
# Productautomata as Intersection Automata

$\Sigma = \{a, b\}$, $a(a + ba)^\omega \cap (a^*ba)^\omega = \emptyset$? No, e.g. $a(ba)^\omega$

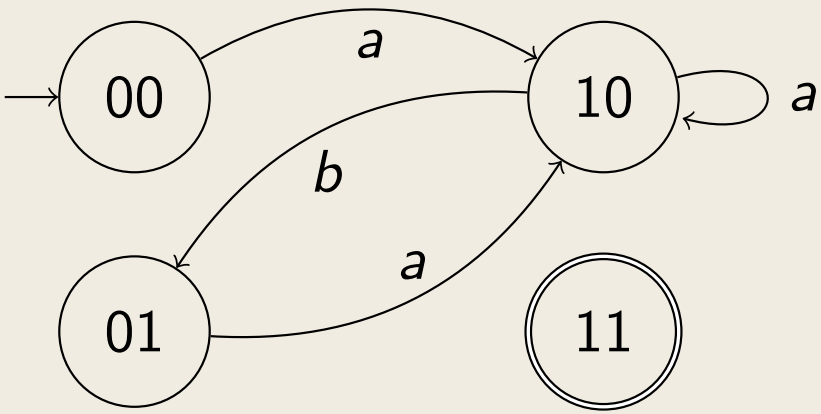
$a(a + ba)^\omega$ :




$(a^*ba)^\omega$ :



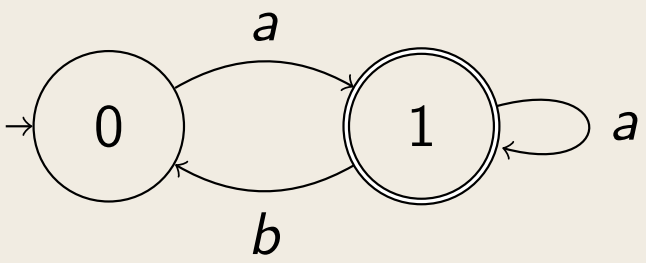

**Productautomaton:**



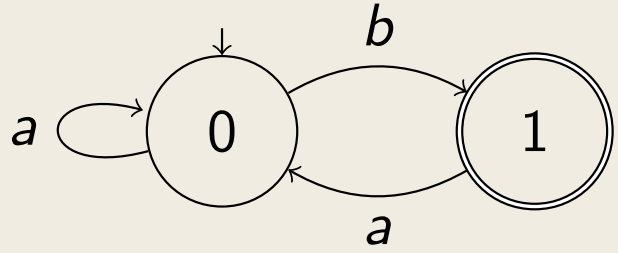
# First Try: Productautomata as Intersection Automata

$\Sigma = \{a, b\}$, $a(a + ba)^\omega \cap (a^* ba)^\omega = \emptyset$? No, e.g. $a(ba)^\omega$

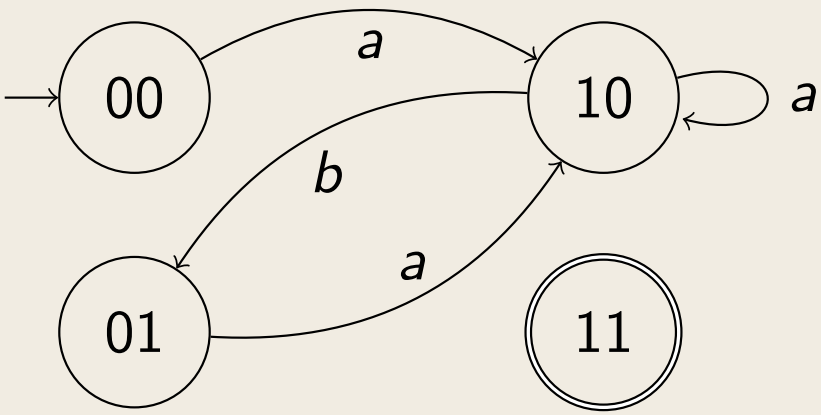
$a(a + ba)^\omega$ :




$(a^* ba)^\omega$ :

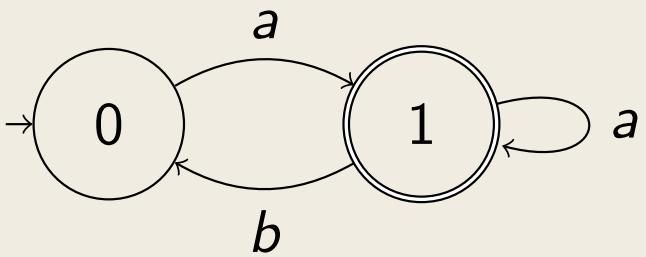



**Productautomaton: Oops, accepting location 11 never reached.**

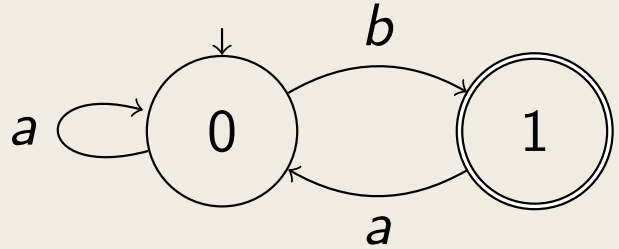


# Next Try: Constructing the Intersection Automata

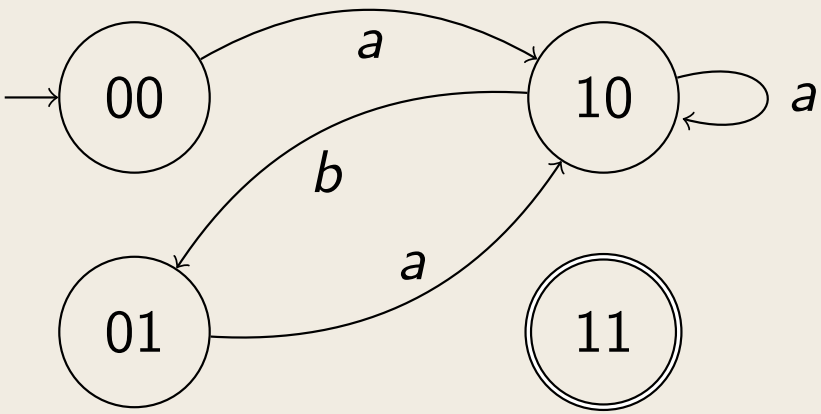
$a(a + ba)^\omega$ :




$(a^* ba)^\omega$ :



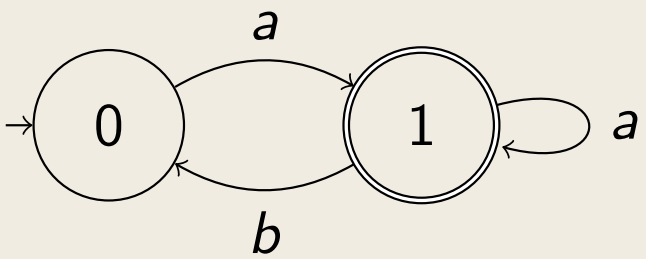

## Construction



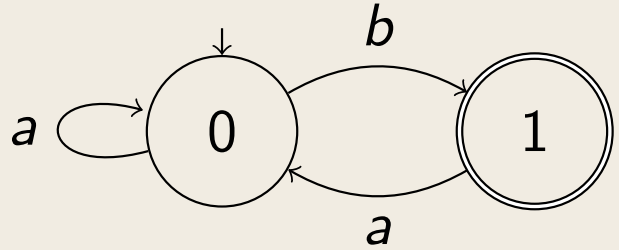

$Q_\cap = Q_1 \times Q_2$

# Next Try: Constructing the Intersection Automata

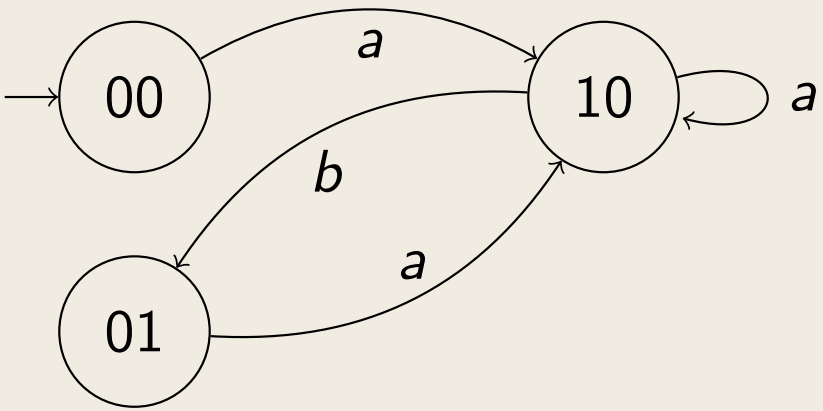
$a(a + ba)^\omega$ :




$(a^* ba)^\omega$ :



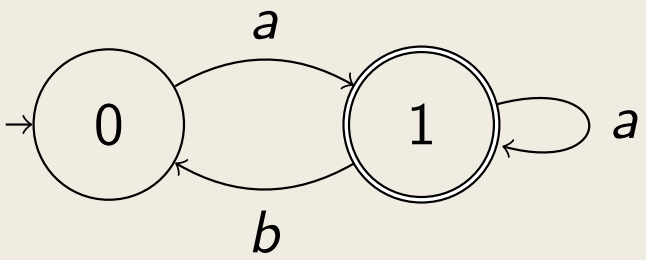

## Construction



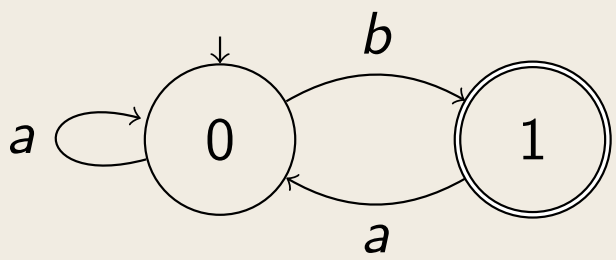

$Q_\cap = Q_1 \times Q_2$

# Next Try: Constructing the Intersection Automata

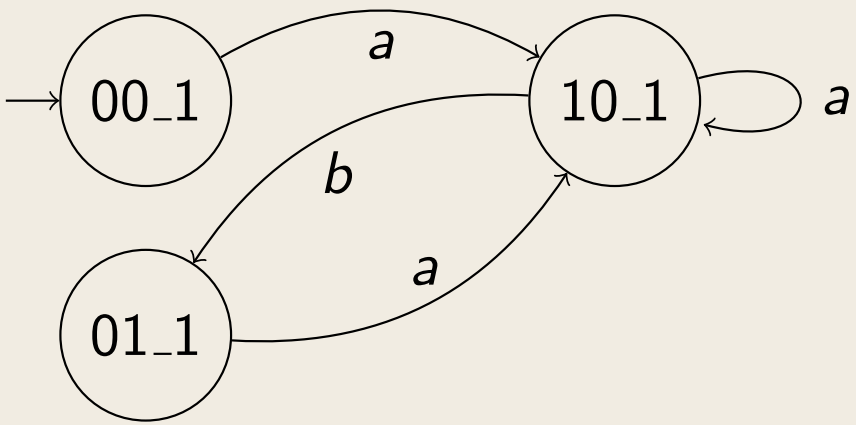
$a(a + ba)^\omega$ :

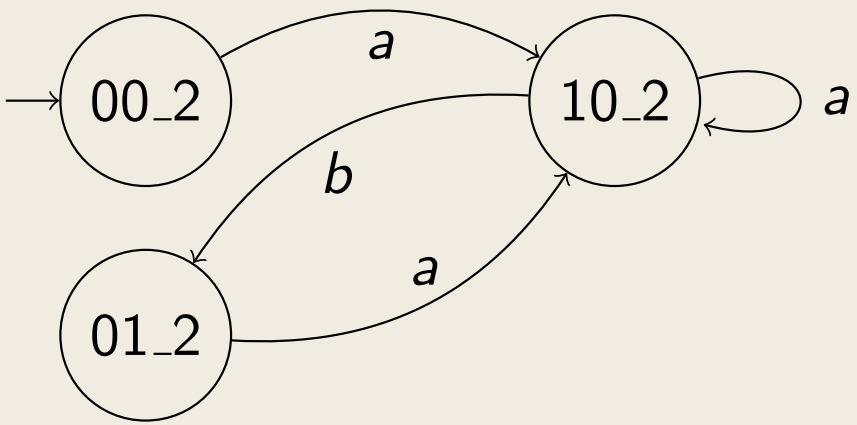



$(a^* ba)^\omega$ :




## Construction

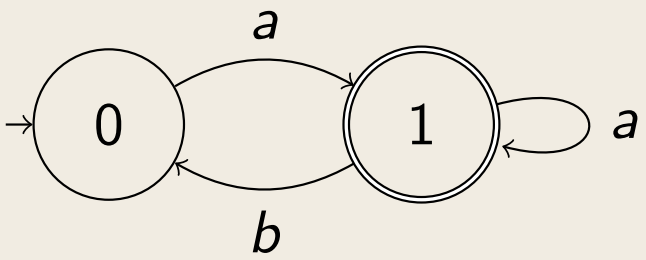


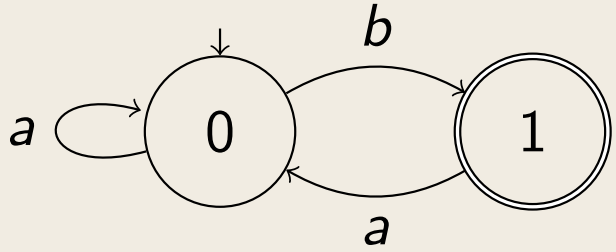

$Q_\cap = Q_1 \times Q_2 \times \{1, 2\}$

# Next Try: Constructing the Intersection Automata

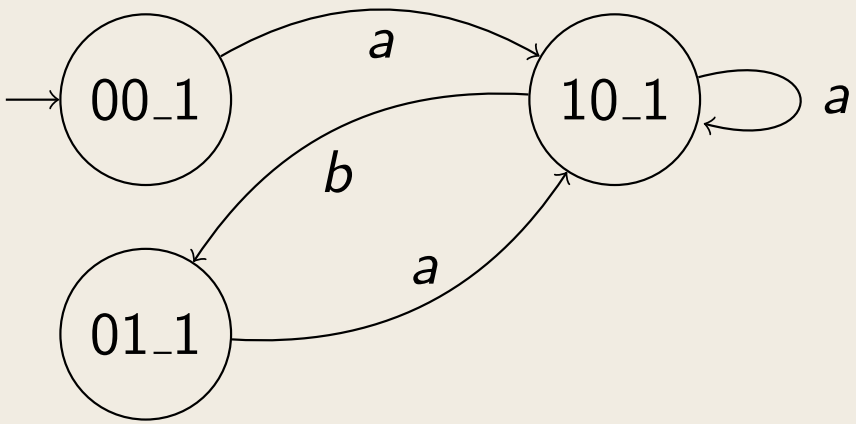
$a(a + ba)^\omega$ :

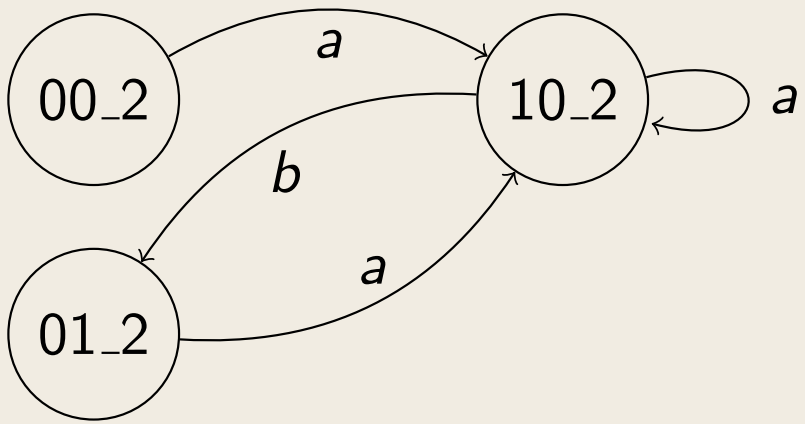



$(a^* ba)^\omega$ :




## Construction

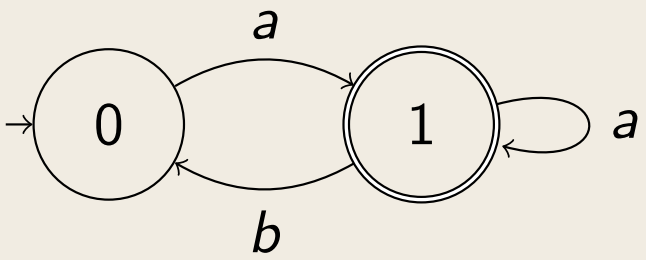


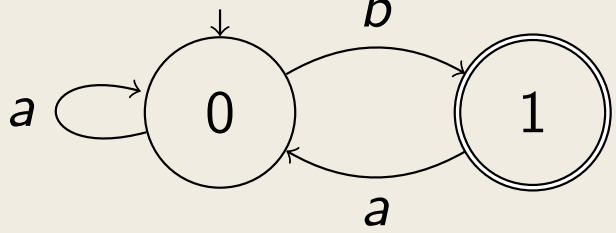

$Q_\cap = Q_1 \times Q_2 \times \{1, 2\}, I_\cap = I_1 \times I_2 \times \{1\}$

# Next Try: Constructing the Intersection Automata

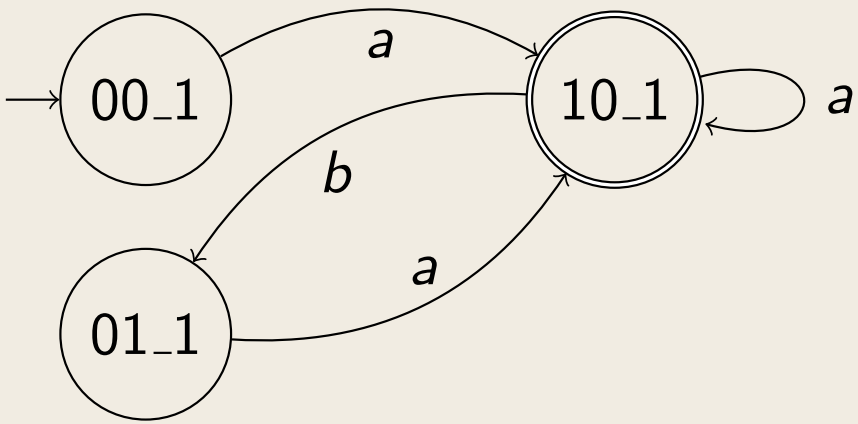
$a(a + ba)^\omega$ :

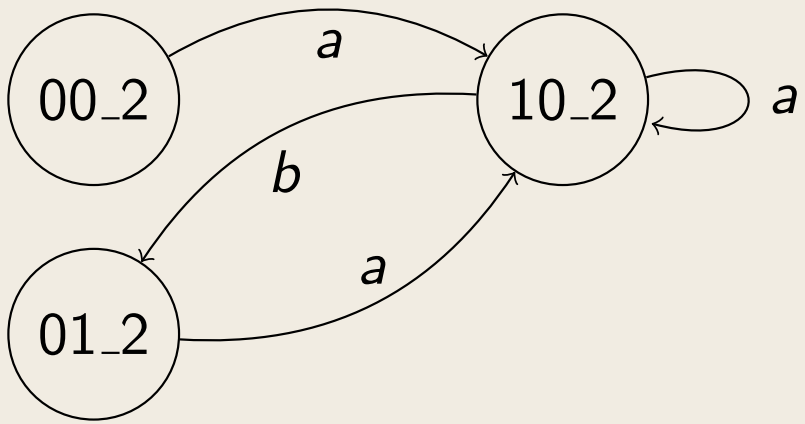



$(a^* ba)^\omega$ :




## Construction

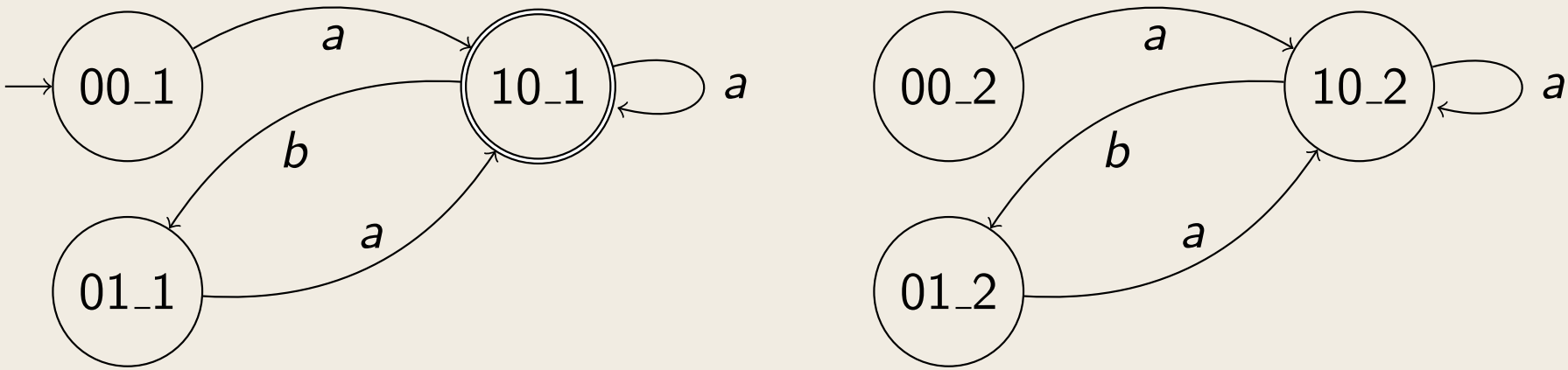



$Q_\cap = Q_1 \times Q_2 \times \{1, 2\}, I_\cap = I_1 \times I_2 \times \{1\}, F = F_1 \times Q_2 \times \{1\}$

# Next Try: Constructing the Intersection Automata

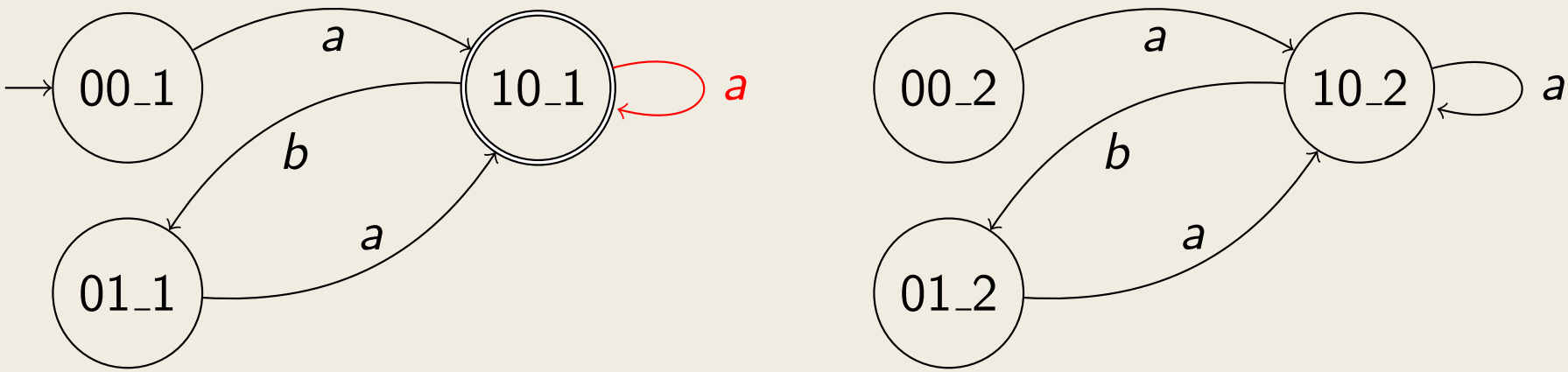
## Construction




$Q_\cap = Q_1 \times Q_2 \times \{1,2\}, I_\cap = I_1 \times I_2 \times \{1\}, F = F_1 \times Q_2 \times \{1\}$

# Next Try: Constructing the Intersection Automata

## Construction



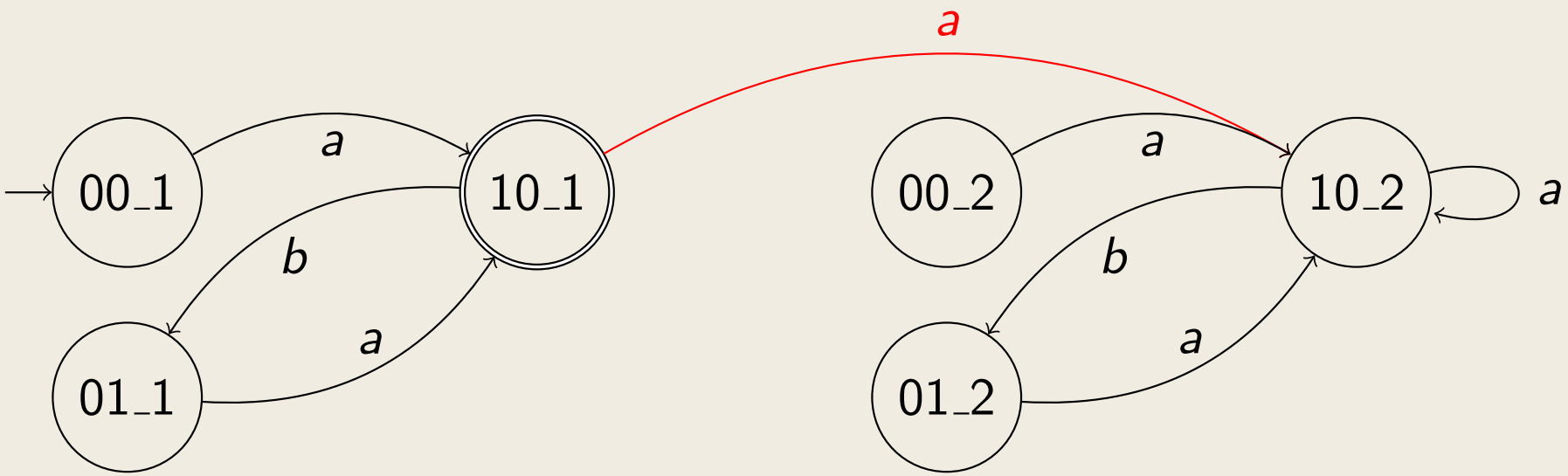

$Q_\cap = Q_1 \times Q_2 \times \{1,2\}, I_\cap = I_1 \times I_2 \times \{1\}, F = F_1 \times Q_2 \times \{1\}$

$s_1 \in Q_1, s_2 \in Q_2, \alpha \in \Sigma$ :

if $s_1 \in F_1$ : $\quad \delta_\cap((s_1, s_2, 1), \alpha) = \{(s_1', s_2', 2) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

# Next Try: Constructing the Intersection Automata

## Construction



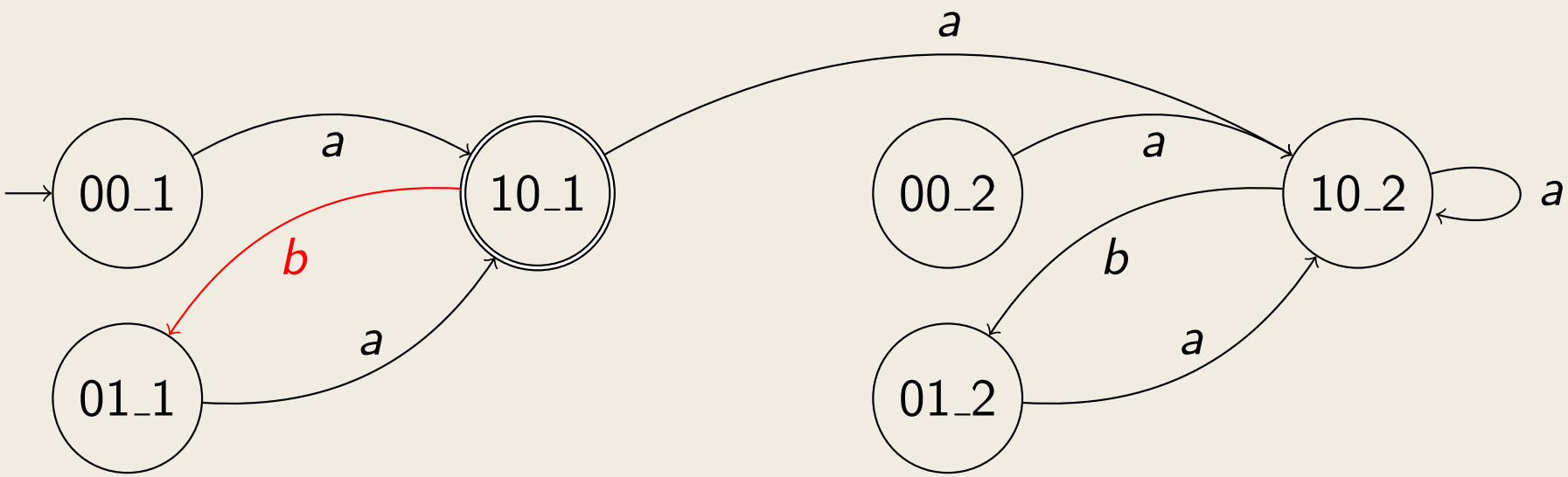

$Q_\cap = Q_1 \times Q_2 \times \{1,2\}, I_\cap = I_1 \times I_2 \times \{1\}, F = F_1 \times Q_2 \times \{1\}$

$s_1 \in Q_1, s_2 \in Q_2, \alpha \in \Sigma$ :

if $s_1 \in F_1$ : $\quad \delta_\cap((s_1, s_2, 1), \alpha) = \{(s_1', s_2', 2) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

# Next Try: Constructing the Intersection Automata

## Construction



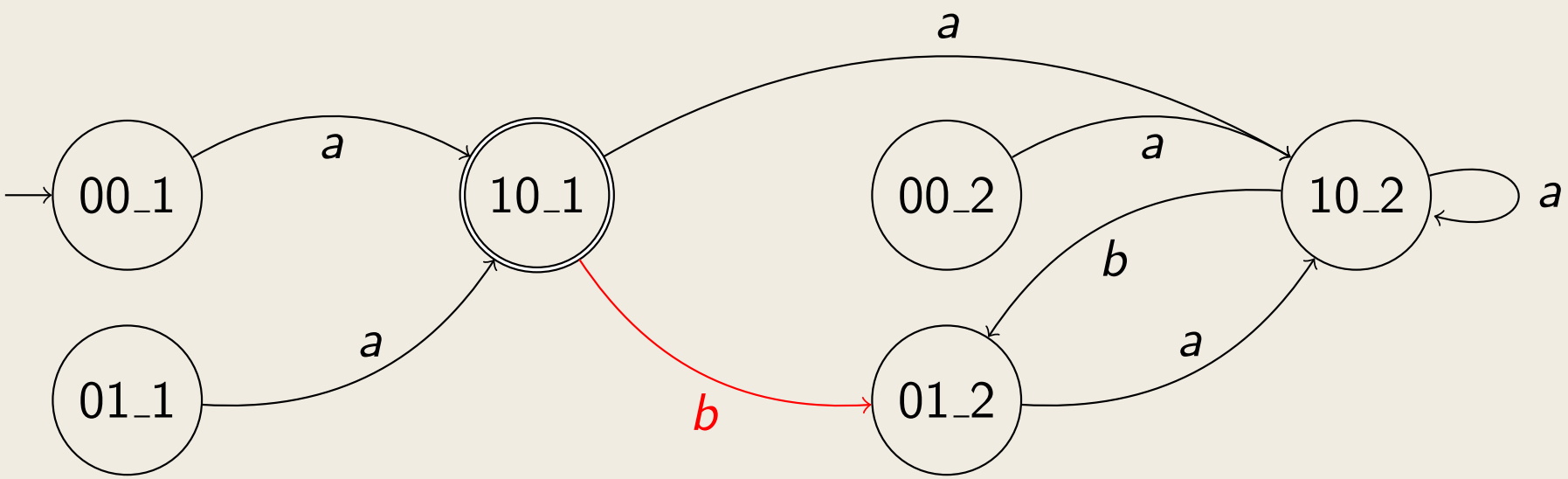

$Q_\cap = Q_1 \times Q_2 \times \{1,2\}, I_\cap = I_1 \times I_2 \times \{1\}, F = F_1 \times Q_2 \times \{1\}$

$s_1 \in Q_1, s_2 \in Q_2, \alpha \in \Sigma$ :

if $s_1 \in F_1$ : $\quad \delta_\cap((s_1, s_2, 1), \alpha) = \{(s_1', s_2', 2) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

# Next Try: Constructing the Intersection Automata

## Construction



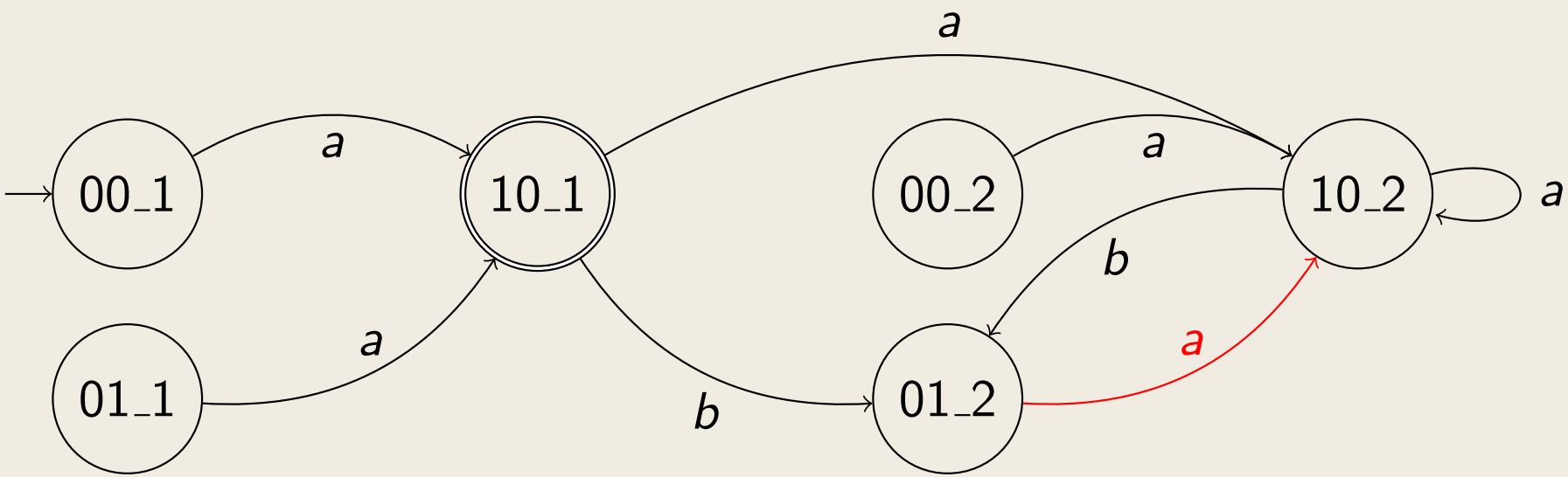

$Q_\cap = Q_1 \times Q_2 \times \{1,2\}, I_\cap = I_1 \times I_2 \times \{1\}, F = F_1 \times Q_2 \times \{1\}$

$s_1 \in Q_1, s_2 \in Q_2, \alpha \in \Sigma$ :

if $s_1 \in F_1$ : $\quad \delta_\cap((s_1, s_2, 1), \alpha) = \{(s_1', s_2', 2) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

# Next Try: Constructing the Intersection Automata

## Construction




$Q_\cap = Q_1 \times Q_2 \times \{1,2\}, I_\cap = I_1 \times I_2 \times \{1\}, F = F_1 \times Q_2 \times \{1\}$

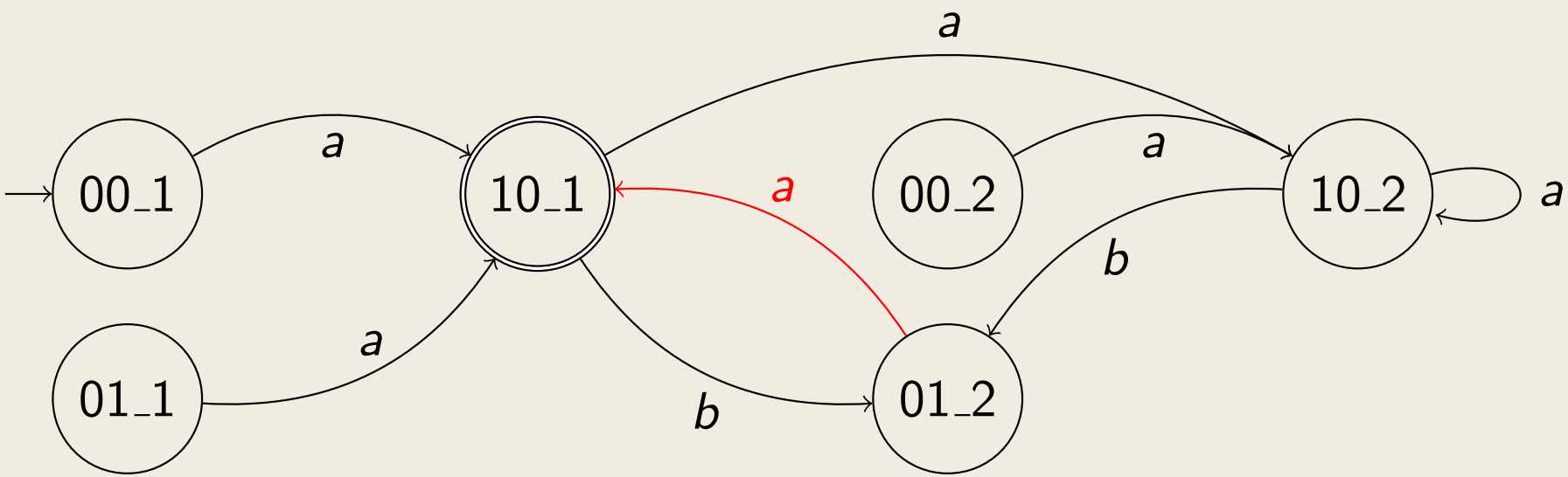
$s_1 \in Q_1, s_2 \in Q_2, \alpha \in \Sigma$ :

if $s_1 \in F_1$ : $\quad \delta_\cap((s_1, s_2, 1), \alpha) = \{(s_1', s_2', 2) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

if $s_2 \in F_2$ : $\quad \delta_\cap((s_1, s_2, 2), \alpha) = \{(s_1', s_2', 1) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

# Next Try: Constructing the Intersection Automata

## Construction




$Q_\cap = Q_1 \times Q_2 \times \{1,2\}, I_\cap = I_1 \times I_2 \times \{1\}, F = F_1 \times Q_2 \times \{1\}$

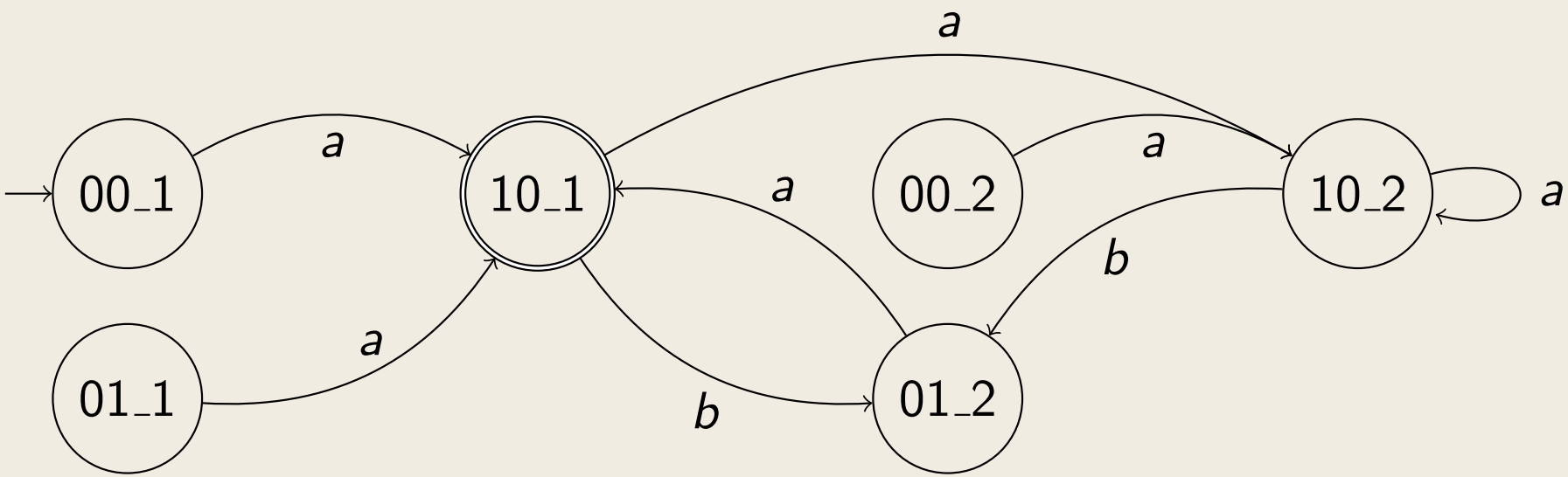
$s_1 \in Q_1, s_2 \in Q_2, \alpha \in \Sigma$ :

if $s_1 \in F_1$ : $\quad \delta_\cap((s_1, s_2, 1), \alpha) = \{(s_1', s_2', 2) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

if $s_2 \in F_2$ : $\quad \delta_\cap((s_1, s_2, 2), \alpha) = \{(s_1', s_2', 1) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

# Next Try: Constructing the Intersection Automata

## Construction




$Q_\cap = Q_1 \times Q_2 \times \{1,2\}, I_\cap = I_1 \times I_2 \times \{1\}, F = F_1 \times Q_2 \times \{1\}$

$s_1 \in Q_1, s_2 \in Q_2, \alpha \in \Sigma$ :

if $s_1 \in F_1$ : $\quad \delta_\cap((s_1, s_2, 1), \alpha) = \{(s_1', s_2', 2) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

if $s_2 \in F_2$ : $\quad \delta_\cap((s_1, s_2, 2), \alpha) = \{(s_1', s_2', 1) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

else: $\quad \delta_\cap((s_1, s_2, i), \alpha) = \{(s_1', s_2', i) | s_1' \in \delta_1(s_1, \alpha), s_2' \in \delta_2(s_2, \alpha)\}$

# Construction of a Büchi Automaton $\mathcal{B}_\phi$ for an LTL-Formula $\phi$

# Generalised Büchi Automaton

A generalised Büchi automaton is defined as:

$$\mathcal{B}^g = (Q, \delta, I, \mathbb{F})$$

$Q, \delta, I$ as for normal Büchi automata.

$\mathbb{F} = \{\mathcal{F}_1, \ldots, \mathcal{F}_n\}$, where $\mathcal{F}_i = \{q_{i1}, \ldots, q_{im_i}\}, q_{ik} \in Q$

# Generalised Büchi Automaton

A generalised Büchi automaton is defined as:

$$\mathcal{B}^g = (Q, \delta, I, \mathbb{F})$$

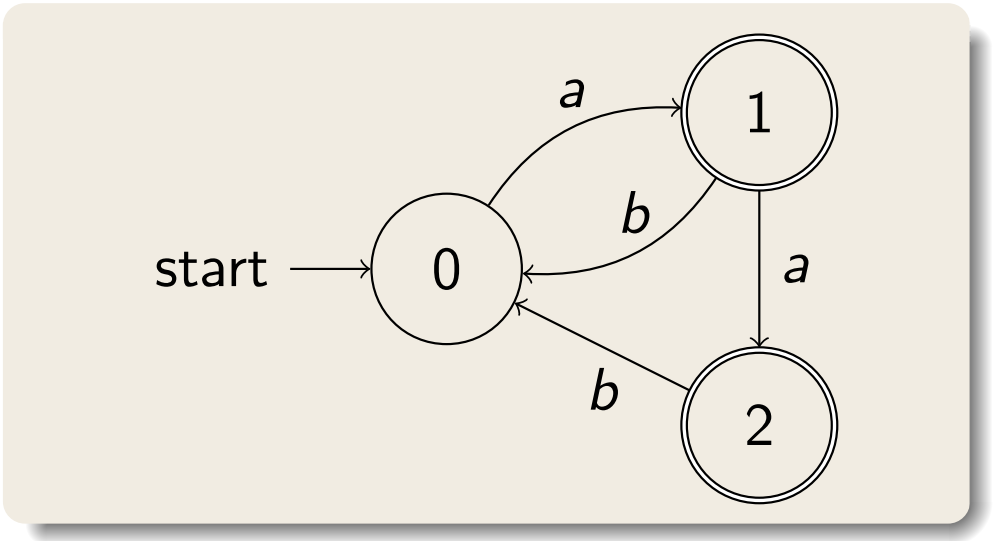
$Q, \delta, I$ as for normal Büchi automata.

$\mathbb{F} = \{\mathcal{F}_1, \ldots, \mathcal{F}_n\}$, where $\mathcal{F}_i = \{q_{i1}, \ldots, q_{im_i}\}, q_{ik} \in Q$

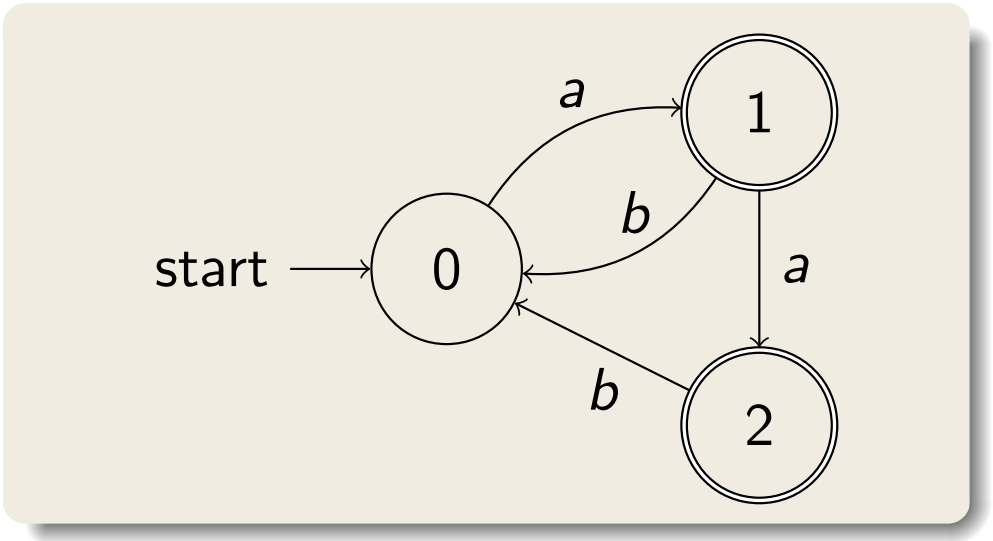
**Definition (Acceptance for generalised Büchi automata)**

A generalised Büchi automata accepts an $\omega$-word $w \in \Sigma^\omega$ iff. for every $i \in \{1, \ldots, n\}$ some $q_{ik} \in \mathcal{F}_i$ are visited infinitely often.

# Example – Generalised Büchi Automaton

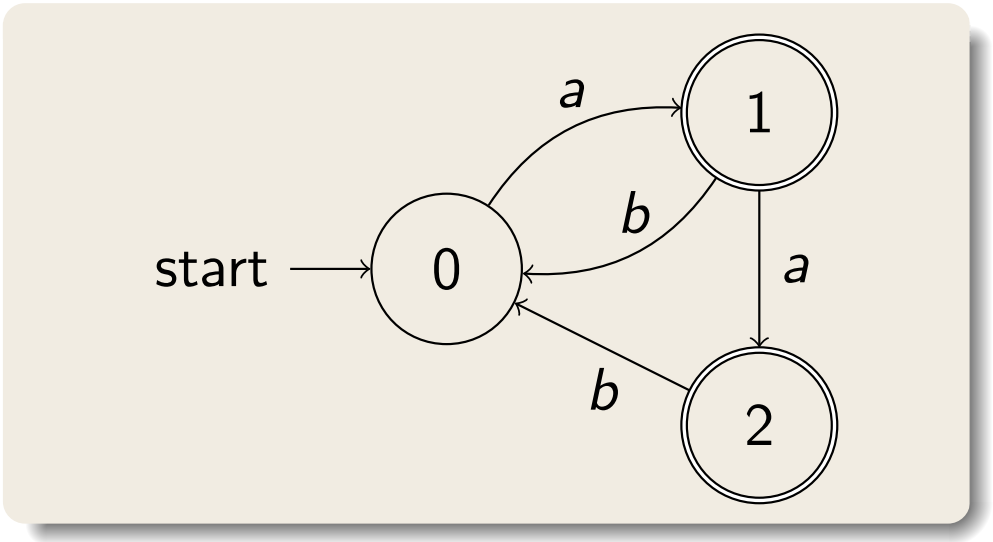


# Example – Generalised Büchi Automaton




$\mathcal{B}^{normal}$ with $\mathcal{F} = \{1, 2\}$, $\mathcal{B}^{general}$ with $\mathbb{F} = \{\overbrace{\{1\}}^{\mathcal{F}_1}, \overbrace{\{2\}}^{\mathcal{F}_2}\}$

# Example – Generalised Büchi Automaton

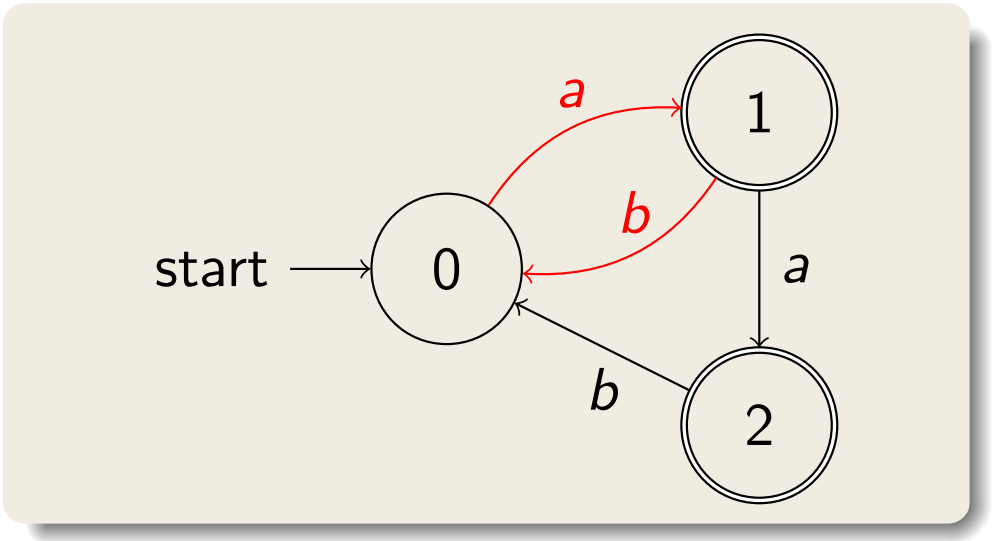



$\mathcal{B}^{normal}$ with $\mathcal{F} = \{1, 2\}$, $\mathcal{B}^{general}$ with $\mathbb{F} = \{\overbrace{\{1\}}^{\mathcal{F}_1}, \overbrace{\{2\}}^{\mathcal{F}_2}\}$

Which $\omega$-word is accepted by which automata?

| $\omega$-word | $\mathcal{B}^{normal}$ | $\mathcal{B}^{general}$ |
|---|---|---|

# Example – Generalised Büchi Automaton

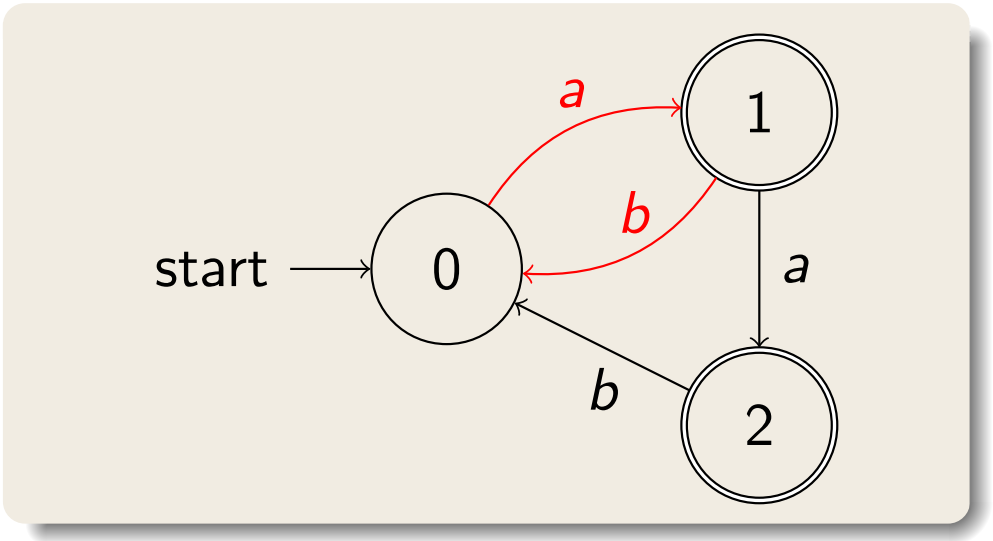



$\mathcal{B}^{normal}$ with $\mathcal{F} = \{1, 2\}$, $\mathcal{B}^{general}$ with $\mathbb{F} = \{\overbrace{\{1\}}^{\mathcal{F}_1}, \overbrace{\{2\}}^{\mathcal{F}_2}\}$

Which $\omega$-word is accepted by which automata?

| $\omega$-word | $\mathcal{B}^{normal}$ | $\mathcal{B}^{general}$ |
|---|---|---|
| $(ab)^\omega$ | | |

# Example – Generalised Büchi Automaton

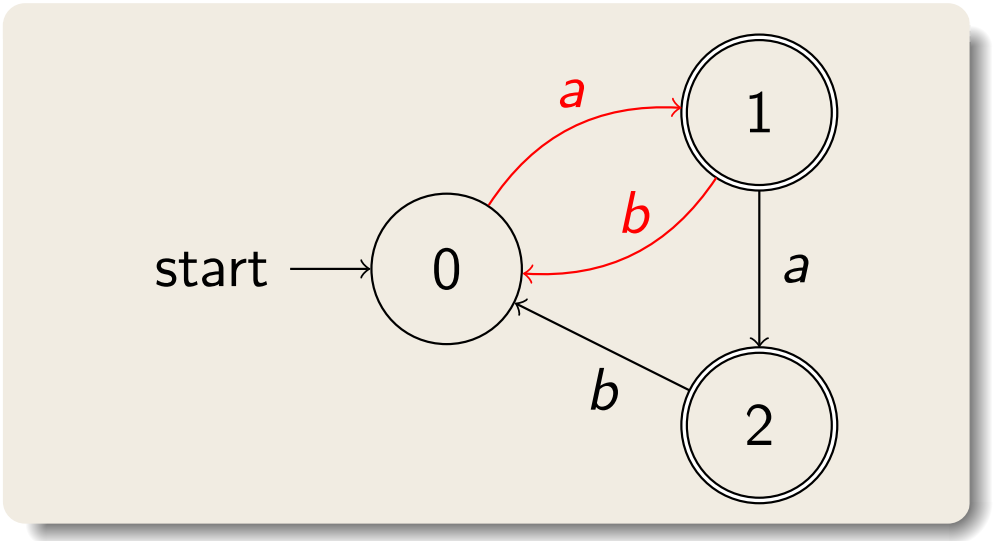



$\mathcal{B}^{normal}$ with $\mathcal{F} = \{1, 2\}$, $\mathcal{B}^{general}$ with $\mathbb{F} = \{\overbrace{\{1\}}^{\mathcal{F}_1}, \overbrace{\{2\}}^{\mathcal{F}_2}\}$

Which $\omega$-word is accepted by which automata?

| $\omega$-word | $\mathcal{B}^{normal}$ | $\mathcal{B}^{general}$ |
|---|---|---|
| $(ab)^\omega$ | ✔ | |

# Example – Generalised Büchi Automaton

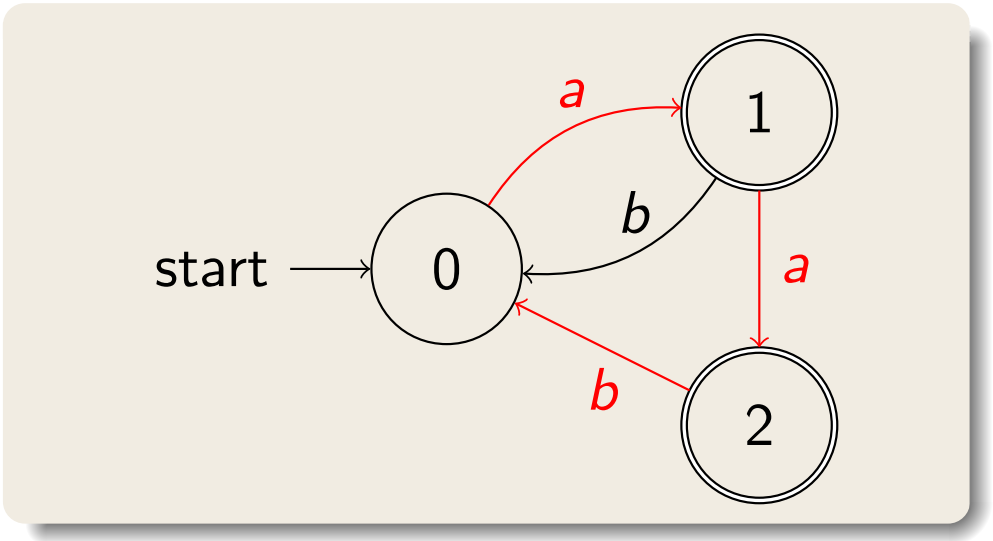



$\mathcal{B}^{normal}$ with $\mathcal{F} = \{1, 2\}$, $\mathcal{B}^{general}$ with $\mathbb{F} = \{\overbrace{\{1\}}^{\mathcal{F}_1}, \overbrace{\{2\}}^{\mathcal{F}_2}\}$

Which $\omega$-word is accepted by which automata?

| $\omega$-word | $\mathcal{B}^{normal}$ | $\mathcal{B}^{general}$ |
|---|---|---|
| $(ab)^\omega$ | ✔ | ✘ |

# Example – Generalised Büchi Automaton

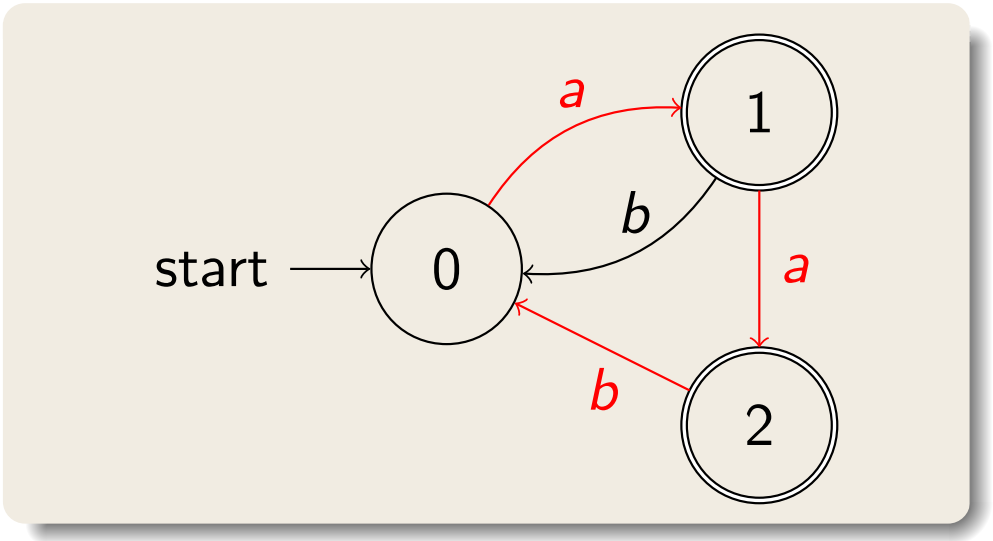



$\mathcal{B}^{normal}$ with $\mathcal{F} = \{1, 2\}$, $\mathcal{B}^{general}$ with $\mathbb{F} = \{\overbrace{\{1\}}^{\mathcal{F}_1}, \overbrace{\{2\}}^{\mathcal{F}_2}\}$

Which $\omega$-word is accepted by which automata?

| $\omega$-word | $\mathcal{B}^{normal}$ | $\mathcal{B}^{general}$ |
|---|---|---|
| $(ab)^\omega$ | ✔ | ✘ |
| $(aab)^\omega$ | | |

# Example – Generalised Büchi Automaton

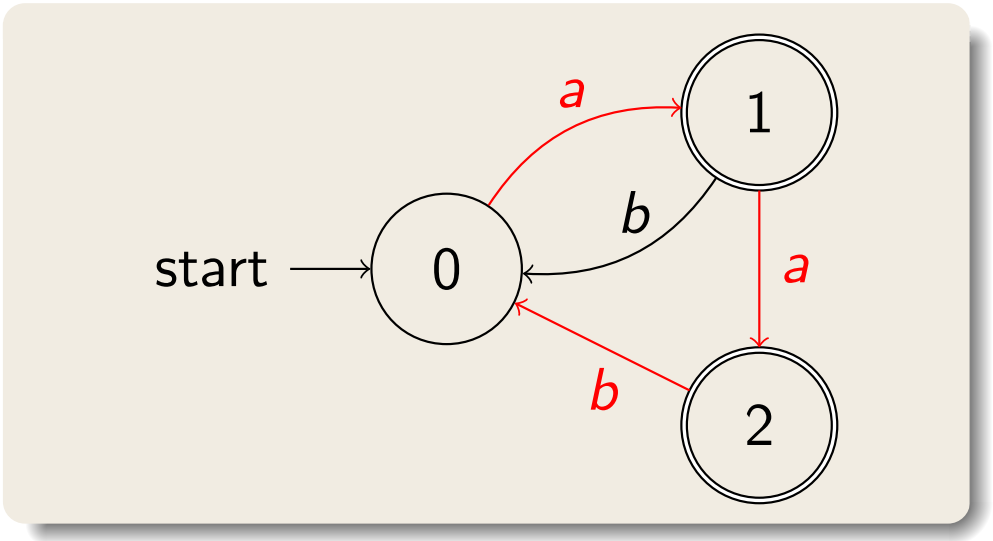



$\mathcal{B}^{normal}$ with $\mathcal{F} = \{1, 2\}$, $\mathcal{B}^{general}$ with $\mathbb{F} = \{\overbrace{\{1\}}^{\mathcal{F}_1}, \overbrace{\{2\}}^{\mathcal{F}_2}\}$

Which $\omega$-word is accepted by which automata?

| $\omega$-word | $\mathcal{B}^{normal}$ | $\mathcal{B}^{general}$ |
|---|---|---|
| $(ab)^\omega$ | ✔ | ✘ |
| $(aab)^\omega$ | ✔ | |

# Example – Generalised Büchi Automaton




$\mathcal{B}^{normal}$ with $\mathcal{F} = \{1, 2\}$, $\mathcal{B}^{general}$ with $\mathbb{F} = \{\overbrace{\{1\}}^{\mathcal{F}_1}, \overbrace{\{2\}}^{\mathcal{F}_2}\}$

Which $\omega$-word is accepted by which automata?

| $\omega$-word | $\mathcal{B}^{normal}$ | $\mathcal{B}^{general}$ |
|---|---|---|
| $(ab)^\omega$ | ✔ | ✘ |
| $(aab)^\omega$ | ✔ | ✔ |

# Fischer-Ladner Closure

Fischer-Ladner closure of an LTL-formula $\phi$

$$FL(\phi) = \{\varphi | \varphi \text{ is subformula or negated subformula of } \phi\}$$

Example

$$FL(r \mathcal{U} s) = \{r, \neg r, s, \neg s, r \mathcal{U} s, \neg(r \mathcal{U} s)\}$$

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

Locations $Q \subseteq 2^{FL(\phi)}$ where each $q \in Q$ satisfies:

1. $\psi \in FL(\phi)$: either $\psi \in q$ or $\neg\psi \in q$, but not both

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

Locations $Q \subseteq 2^{FL(\phi)}$ where each $q \in Q$ satisfies:

1. $\psi \in FL(\phi)$: either $\psi \in q$ or $\neg\psi \in q$, but not both
2. $\psi_1 \wedge \psi_2 \in q$: $\psi_1 \in q$ and $\psi_2 \in q$

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

Locations $Q \subseteq 2^{FL(\phi)}$ where each $q \in Q$ satisfies:

1. $\psi \in FL(\phi)$: either $\psi \in q$ or $\neg\psi \in q$, but not both
2. $\psi_1 \wedge \psi_2 \in q$: $\psi_1 \in q$ and $\psi_2 \in q$
3. other propositional connectives similar

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

Locations $Q \subseteq 2^{FL(\phi)}$ where each $q \in Q$ satisfies:

1. $\psi \in FL(\phi)$: either $\psi \in q$ or $\neg\psi \in q$, but not both
2. $\psi_1 \wedge \psi_2 \in q$: $\psi_1 \in q$ and $\psi_2 \in q$
3. other propositional connectives similar
4. $\psi_1 \mathcal{U} \psi_2 \in q$ then $\psi_1 \in q$ or $\psi_2 \in q$

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

Locations $Q \subseteq 2^{FL(\phi)}$ where each $q \in Q$ satisfies:

1. $\psi \in FL(\phi)$: either $\psi \in q$ or $\neg\psi \in q$, but not both
2. $\psi_1 \wedge \psi_2 \in q$: $\psi_1 \in q$ and $\psi_2 \in q$
3. other propositional connectives similar
4. $\psi_1 \mathcal{U} \psi_2 \in q$ then $\psi_1 \in q$ or $\psi_2 \in q$
5. $\psi_1 \mathcal{U} \psi_2 \in (FL(\phi) \backslash q)$ then $\psi_2 \notin q$

$$FL(r\,\mathcal{U}\,s) = \{r, \neg r, s, \neg s, r\,\mathcal{U}\,s, \neg(r\,\mathcal{U}\,s)\}$$
$$\in Q$$

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

Locations $Q \subseteq 2^{FL(\phi)}$ where each $q \in Q$ satisfies:

1. $\psi \in FL(\phi)$: either $\psi \in q$ or $\neg\psi \in q$, but not both
2. $\psi_1 \wedge \psi_2 \in q$: $\psi_1 \in q$ and $\psi_2 \in q$
3. other propositional connectives similar
4. $\psi_1 \mathcal{U} \psi_2 \in q$ then $\psi_1 \in q$ or $\psi_2 \in q$
5. $\psi_1 \mathcal{U} \psi_2 \in (FL(\phi) \backslash q)$ then $\psi_2 \notin q$

$$FL(r\,\mathcal{U}\,s) = \{r, \neg r, s, \neg s, r\,\mathcal{U}\,s, \neg(r\,\mathcal{U}\,s)\}$$

| | $\in Q$ |
|---|---|
| $\{r\,\mathcal{U}\,s, \neg r, s\}$ | ✔ |

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

Locations $Q \subseteq 2^{FL(\phi)}$ where each $q \in Q$ satisfies:

1. $\psi \in FL(\phi)$: either $\psi \in q$ or $\neg\psi \in q$, but not both
2. $\psi_1 \wedge \psi_2 \in q$: $\psi_1 \in q$ and $\psi_2 \in q$
3. other propositional connectives similar
4. $\psi_1 \mathcal{U} \psi_2 \in q$ then $\psi_1 \in q$ or $\psi_2 \in q$
5. $\psi_1 \mathcal{U} \psi_2 \in (FL(\phi) \backslash q)$ then $\psi_2 \notin q$

$$FL(r\,\mathcal{U}\,s) = \{r, \neg r, s, \neg s, r\,\mathcal{U}\,s, \neg(r\,\mathcal{U}\,s)\}$$

| | $\in Q$ |
|---|---|
| $\{r\,\mathcal{U}\,s, \neg r, s\}$ | ✔ |
| $\{r\,\mathcal{U}\,s, \neg r, \neg s\}$ | ✘ |

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

Locations $Q \subseteq 2^{FL(\phi)}$ where each $q \in Q$ satisfies:

1. $\psi \in FL(\phi)$: either $\psi \in q$ or $\neg\psi \in q$, but not both
2. $\psi_1 \wedge \psi_2 \in q$: $\psi_1 \in q$ and $\psi_2 \in q$
3. other propositional connectives similar
4. $\psi_1 \mathcal{U} \psi_2 \in q$ then $\psi_1 \in q$ or $\psi_2 \in q$
5. $\psi_1 \mathcal{U} \psi_2 \in (FL(\phi) \backslash q)$ then $\psi_2 \notin q$

$$FL(r\,\mathcal{U}\,s) = \{r, \neg r, s, \neg s, r\,\mathcal{U}\,s, \neg(r\,\mathcal{U}\,s)\}$$

| | $\in Q$ |
|---|---|
| $\{r\,\mathcal{U}\,s, \neg r, s\}$ | ✔ |
| $\{r\,\mathcal{U}\,s, \neg r, \neg s\}$ | ✘ |
| $\{\neg(r\,\mathcal{U}\,s), r, s\}$ | ✘ |

# $\mathcal{B}_\phi$-Construction: Locations

Assumption: $\mathcal{U}$ the only temporal logic operator in LTL-formula $\phi$ ($\Box, \Diamond$ expressable with $\mathcal{U}$)

Locations $Q \subseteq 2^{FL(\phi)}$ where each $q \in Q$ satisfies:

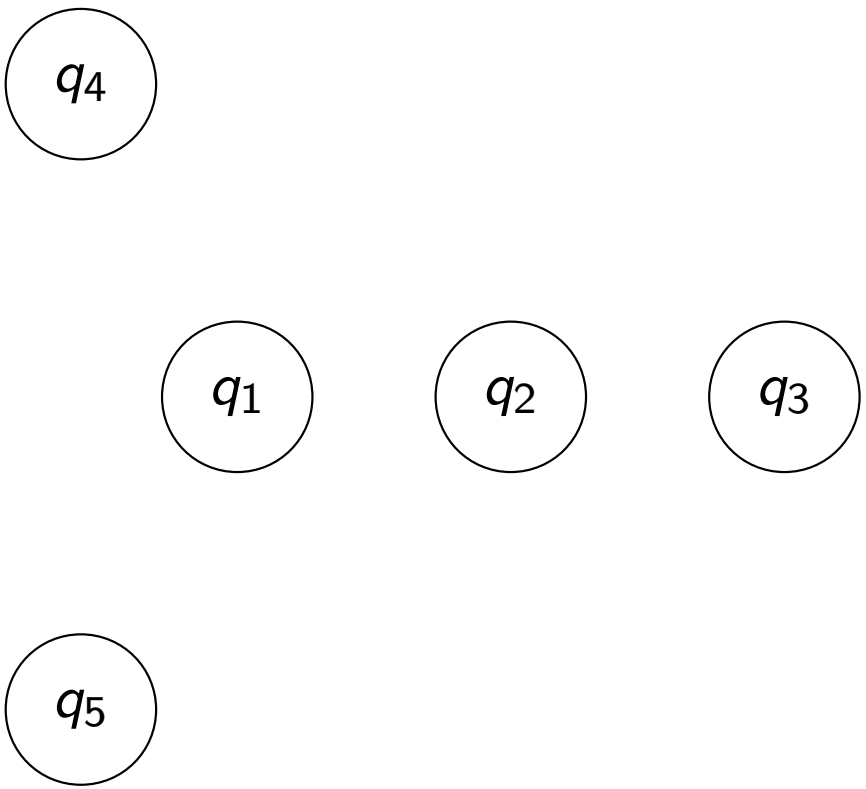
1. $\psi \in FL(\phi)$: either $\psi \in q$ or $\neg\psi \in q$, but not both
2. $\psi_1 \wedge \psi_2 \in q$: $\psi_1 \in q$ and $\psi_2 \in q$
3. other propositional connectives similar
4. $\psi_1 \mathcal{U} \psi_2 \in q$ then $\psi_1 \in q$ or $\psi_2 \in q$
5. $\psi_1 \mathcal{U} \psi_2 \in (FL(\phi) \backslash q)$ then $\psi_2 \notin q$

$$FL(r\,\mathcal{U}\,s) = \{r, \neg r, s, \neg s, r\,\mathcal{U}\,s, \neg(r\,\mathcal{U}\,s)\}$$

| | $\in Q$ |
|---|---|
| $\{r\,\mathcal{U}\,s, \neg r, s\}$ | ✔ |
| $\{r\,\mathcal{U}\,s, \neg r, \neg s\}$ | ✘ |
| $\{\neg(r\,\mathcal{U}\,s), r, s\}$ | ✘ |
| $\{\neg(r\,\mathcal{U}\,s), r, \neg s\}$ | ✔ |

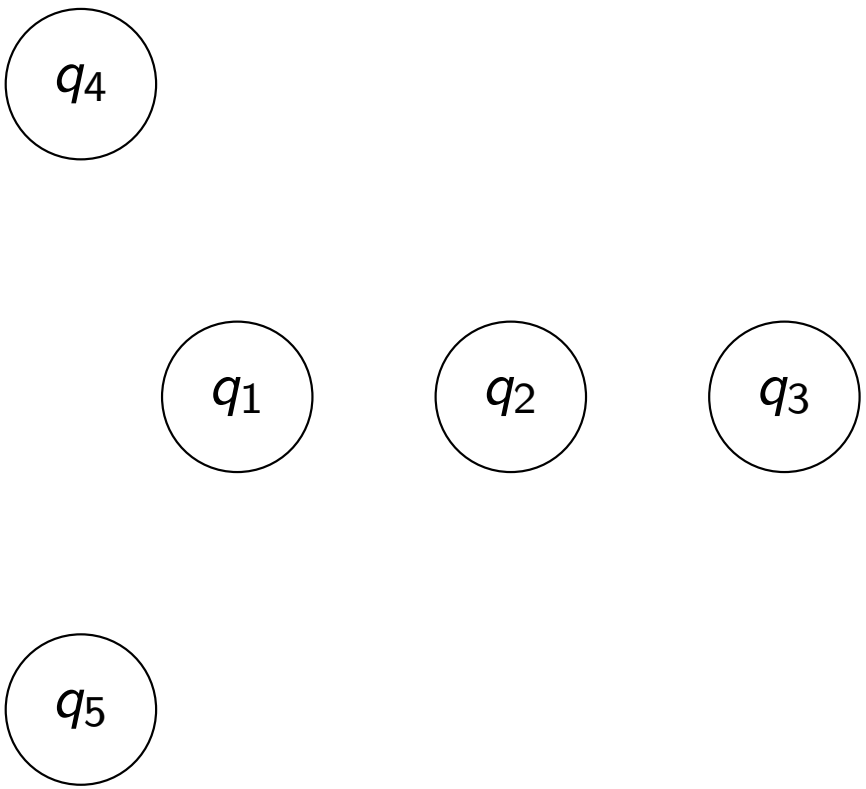
# $\mathcal{B}_\phi$-Construction: Transitions

$$\underbrace{\{r\,\mathcal{U}\,s, \neg r, s\}}_{q_1}, \underbrace{\{r\,\mathcal{U}\,s, r, \neg s\}}_{q_2}, \underbrace{\{r\,\mathcal{U}\,s, r, s\}}_{q_3}, \underbrace{\{\neg(r\,\mathcal{U}\,s), r, \neg s\}}_{q_4}, \underbrace{\{\neg(r\,\mathcal{U}\,s), \neg r, \neg s\}}_{q_5}$$



# $\mathcal{B}_\phi$-Construction: Transitions

$$\underbrace{\{r\,\mathcal{U}\,s, \neg r, s\}}_{q_1}, \underbrace{\{r\,\mathcal{U}\,s, r, \neg s\}}_{q_2}, \underbrace{\{r\,\mathcal{U}\,s, r, s\}}_{q_3}, \underbrace{\{\neg(r\,\mathcal{U}\,s), r, \neg s\}}_{q_4}, \underbrace{\{\neg(r\,\mathcal{U}\,s), \neg r, \neg s\}}_{q_5}$$

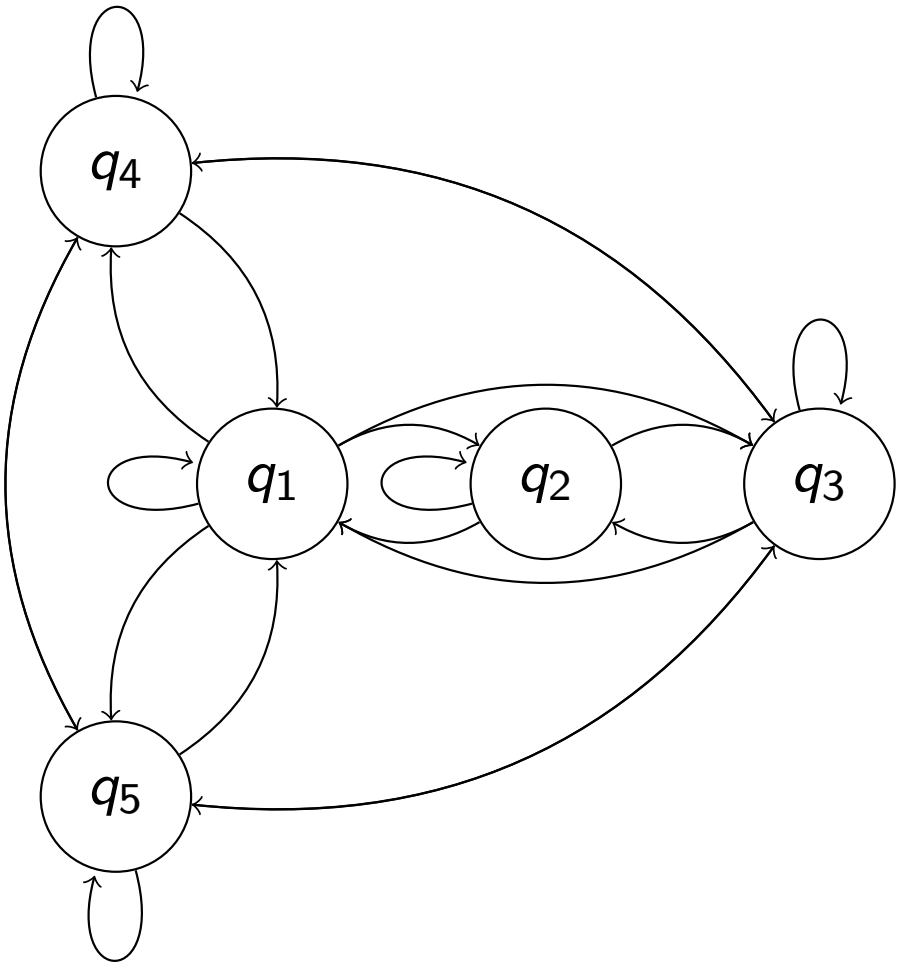



Transitions $(q, \alpha, q') \in \delta_\phi$:

- $\alpha = q \cap \mathcal{P}$
  ($\mathcal{P}$ set of propositional variables); e.g. outgoing edges of $q_1$ labeled $\{s\}$; of $q_2$ labeled $\{r\}$ etc.
- If $\psi_1\,\mathcal{U}\psi_2 \in q$ and $\psi_2 \notin q$ then $\psi_1\,\mathcal{U}\psi_2 \in q'$
- If $\psi_1\,\mathcal{U}\psi_2 \in (FL(\phi)\backslash q)$ and $\psi_1 \in q$ the $\psi_1\,\mathcal{U}\psi_2 \notin q'$

# $\mathcal{B}_\phi$-Construction: Transitions

$$\underbrace{\{r\,\mathcal{U}\,s, \neg r, s\}}_{q_1}, \underbrace{\{r\,\mathcal{U}\,s, r, \neg s\}}_{q_2}, \underbrace{\{r\,\mathcal{U}\,s, r, s\}}_{q_3}, \underbrace{\{\neg(r\,\mathcal{U}\,s), r, \neg s\}}_{q_4}, \underbrace{\{\neg(r\,\mathcal{U}\,s), \neg r, \neg s\}}_{q_5}$$

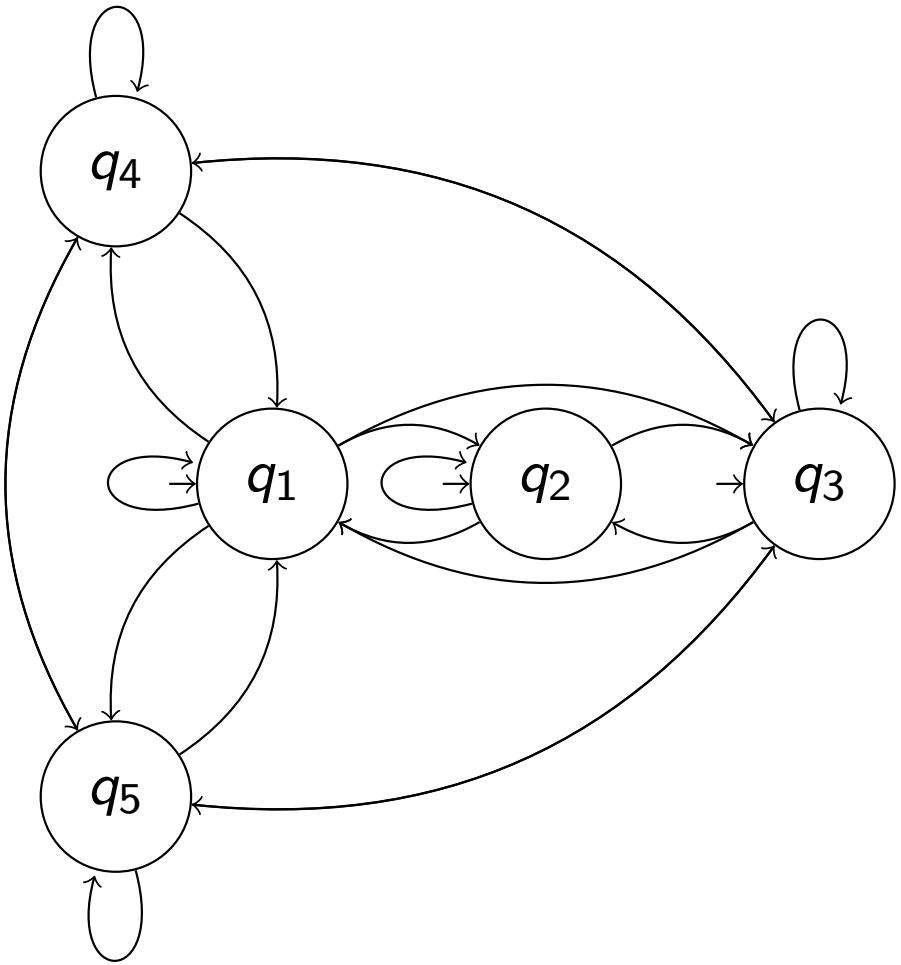



Transitions $(q, \alpha, q') \in \delta_\phi$:

- $\alpha = q \cap \mathcal{P}$
  ($\mathcal{P}$ set of propositional variables); e.g. outgoing edges of $q_1$ labeled $\{s\}$; of $q_2$ labeled $\{r\}$ etc.
- If $\psi_1\,\mathcal{U}\psi_2 \in q$ and $\psi_2 \notin q$ then $\psi_1\,\mathcal{U}\psi_2 \in q'$
- If $\psi_1\,\mathcal{U}\psi_2 \in (FL(\phi)\backslash q)$ and $\psi_1 \in q$ the $\psi_1\,\mathcal{U}\psi_2 \notin q'$

# $\mathcal{B}_\phi$-Construction: Transitions

$$\underbrace{\{r\,\mathcal{U}\,s, \neg r, s\}}_{q_1}, \underbrace{\{r\,\mathcal{U}\,s, r, \neg s\}}_{q_2}, \underbrace{\{r\,\mathcal{U}\,s, r, s\}}_{q_3}, \underbrace{\{\neg(r\,\mathcal{U}\,s), r, \neg s\}}_{q_4}, \underbrace{\{\neg(r\,\mathcal{U}\,s), \neg r, \neg s\}}_{q_5}$$

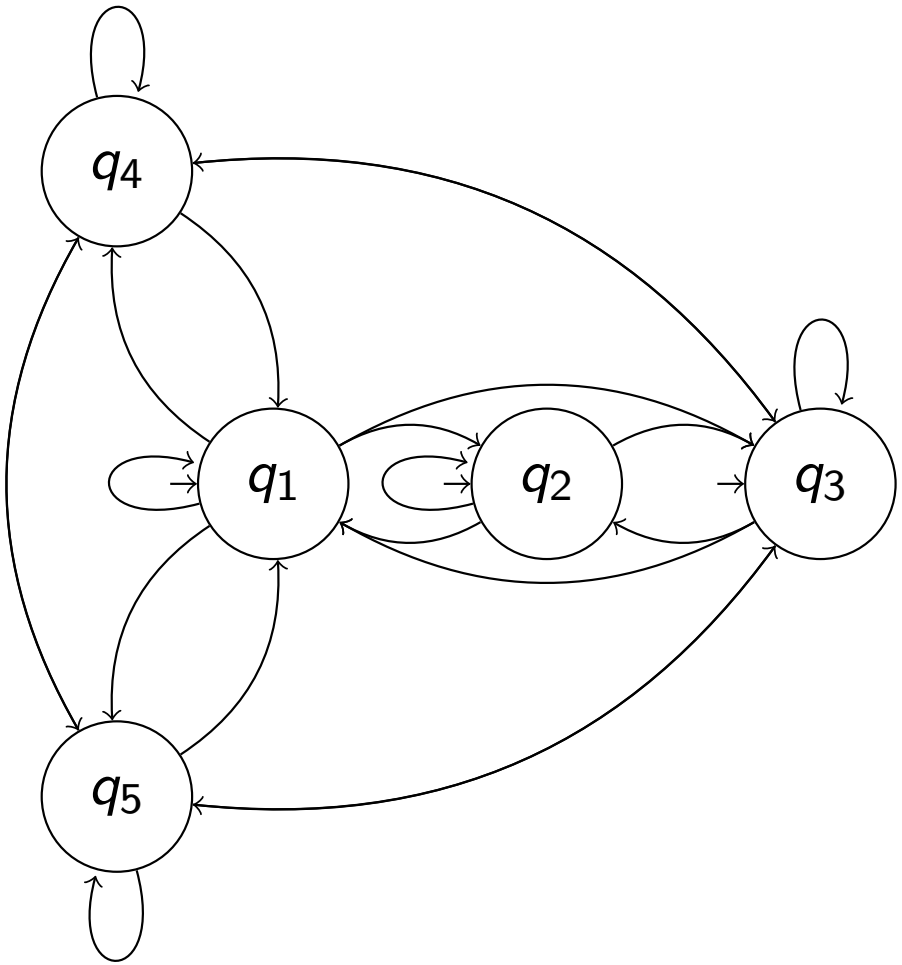



Initial locations

$q \in I_\phi$ iff. $\phi \in q$

# $\mathcal{B}_\phi$-Construction: Transitions

$$\underbrace{\{r\,\mathcal{U}\,s, \neg r, s\}}_{q_1}, \underbrace{\{r\,\mathcal{U}\,s, r, \neg s\}}_{q_2}, \underbrace{\{r\,\mathcal{U}\,s, r, s\}}_{q_3}, \underbrace{\{\neg(r\,\mathcal{U}\,s), r, \neg s\}}_{q_4}, \underbrace{\{\neg(r\,\mathcal{U}\,s), \neg r, \neg s\}}_{q_5}$$




## Initial locations

$$q \in I_\phi \text{ iff. } \phi \in q$$

## Accepting locations

$$\mathbb{F} = \{\mathcal{F}_1, \ldots, \mathcal{F}_n\}$$

- for each subformula $\psi_{i1}\,\mathcal{U}\psi_{i2} \in FL(\phi)$ exists an $\mathcal{F}_i$; in example: $\mathbb{F} = \{\mathcal{F}_1\}$
- $\mathcal{F}_i$ set of locations *not* containing $\psi_{i1}\,\mathcal{U}\psi_{i2}$ or that contain $\psi_{i2}$
  in ex. $\mathcal{F}_1 = \{q_1, q_3, q_4, q_5\}$