From Regular Expressions to Star Fragments

Todd Schmid

St. Mary's College of California (Bucknell University starting in July)

Based on *Coalgebraic Completeness Theorems for Effectful Process Calculi*, UCL, 2023 and joint work with

Wojciech Rozowski (UCL) Tobias Kappé (Open Universiteit)

Dexter Kozen (Cornell University) Jurriaan Rot (Radboud University) Alexandra Silva (Cornell University)

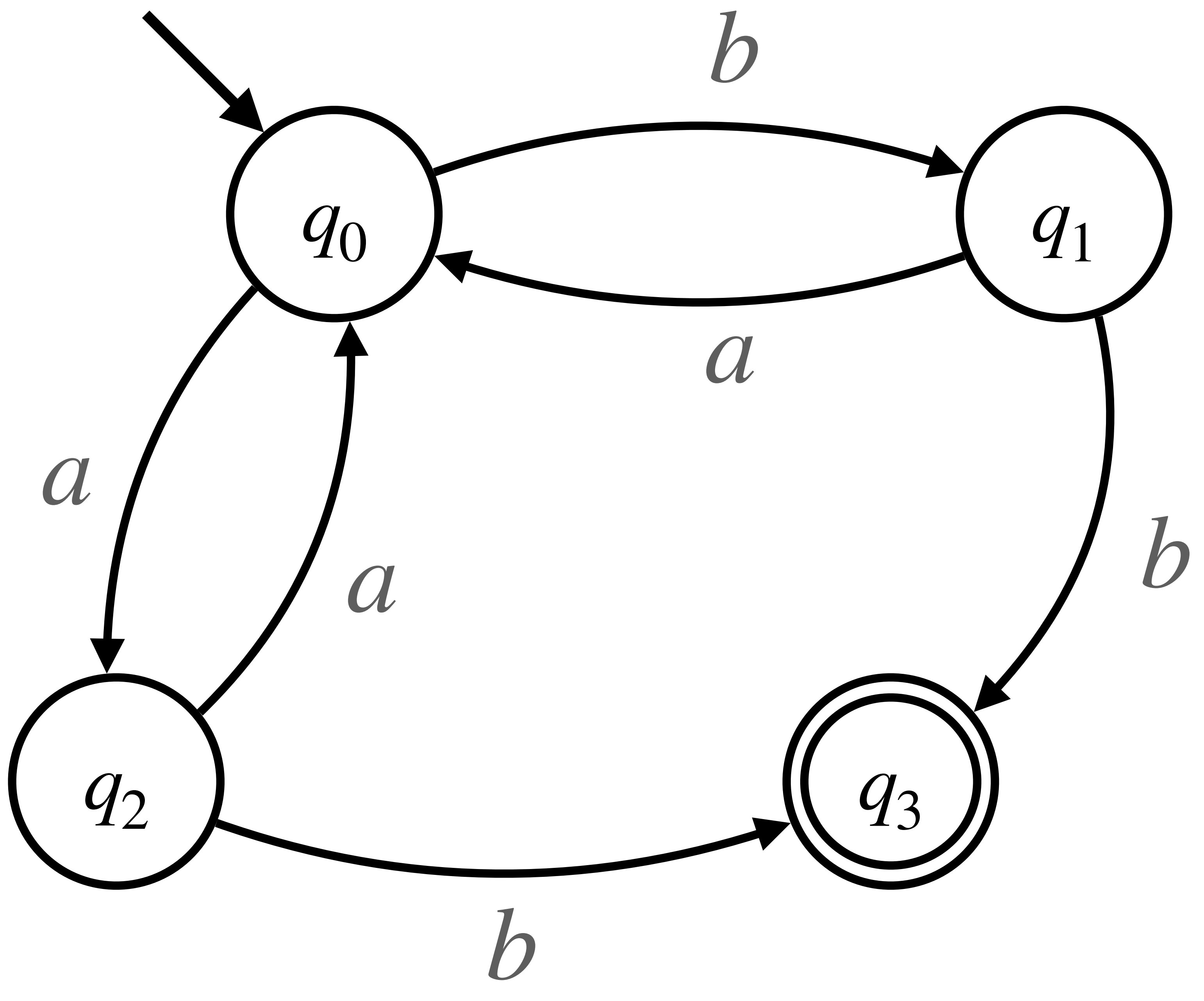
LLAMA Seminar

This Talk

1. Regular expressions and regular languages
2. Axioms for language equivalence *á la* Salomaa
3. Process (bisimilarity) semantics of regular expressions
4. Guarded Kleene Algebra with Tests mod bisimilarity
5. What these process algebras have in common
6. Star Fragments
7. Open Problems

Regular Languages

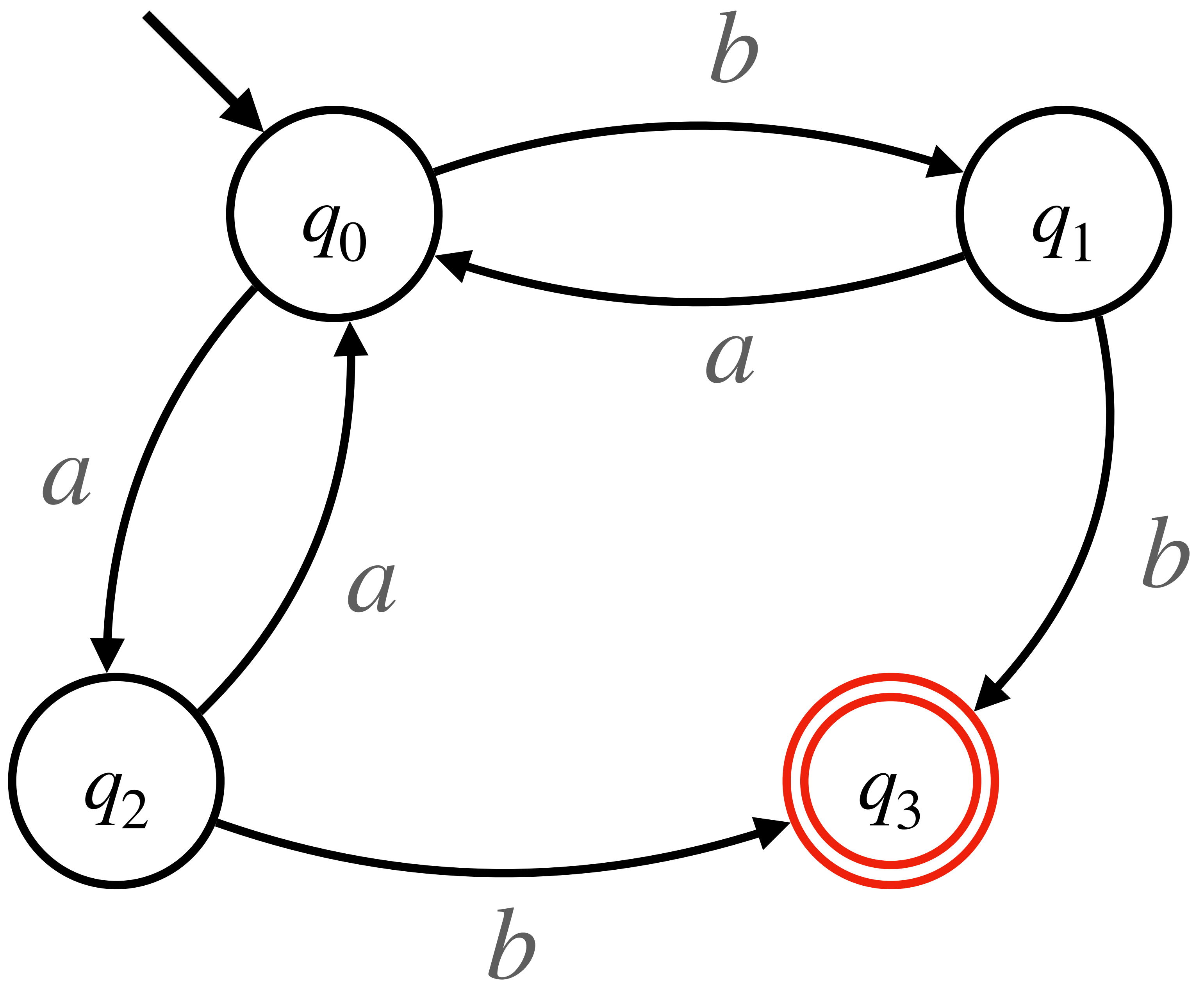
$$X \to \{\bot, \top\} \times X^A$$



(Kleene, 1956)

Regular Languages

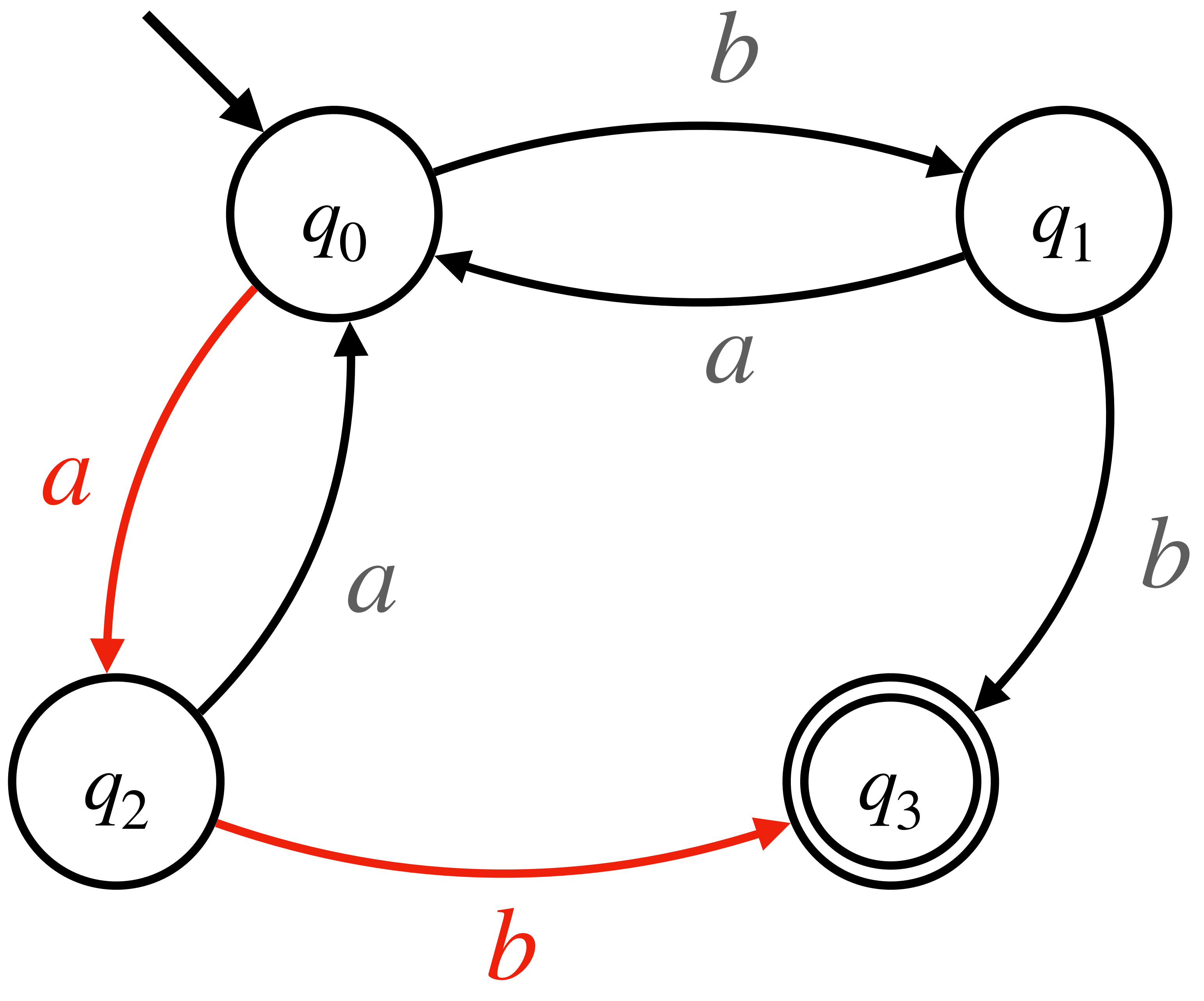
$$X \to \{\bot, \top\} \times X^A$$



(Kleene, 1956)

Regular Languages

$X \to \{\bot, \top\} \times X^A$

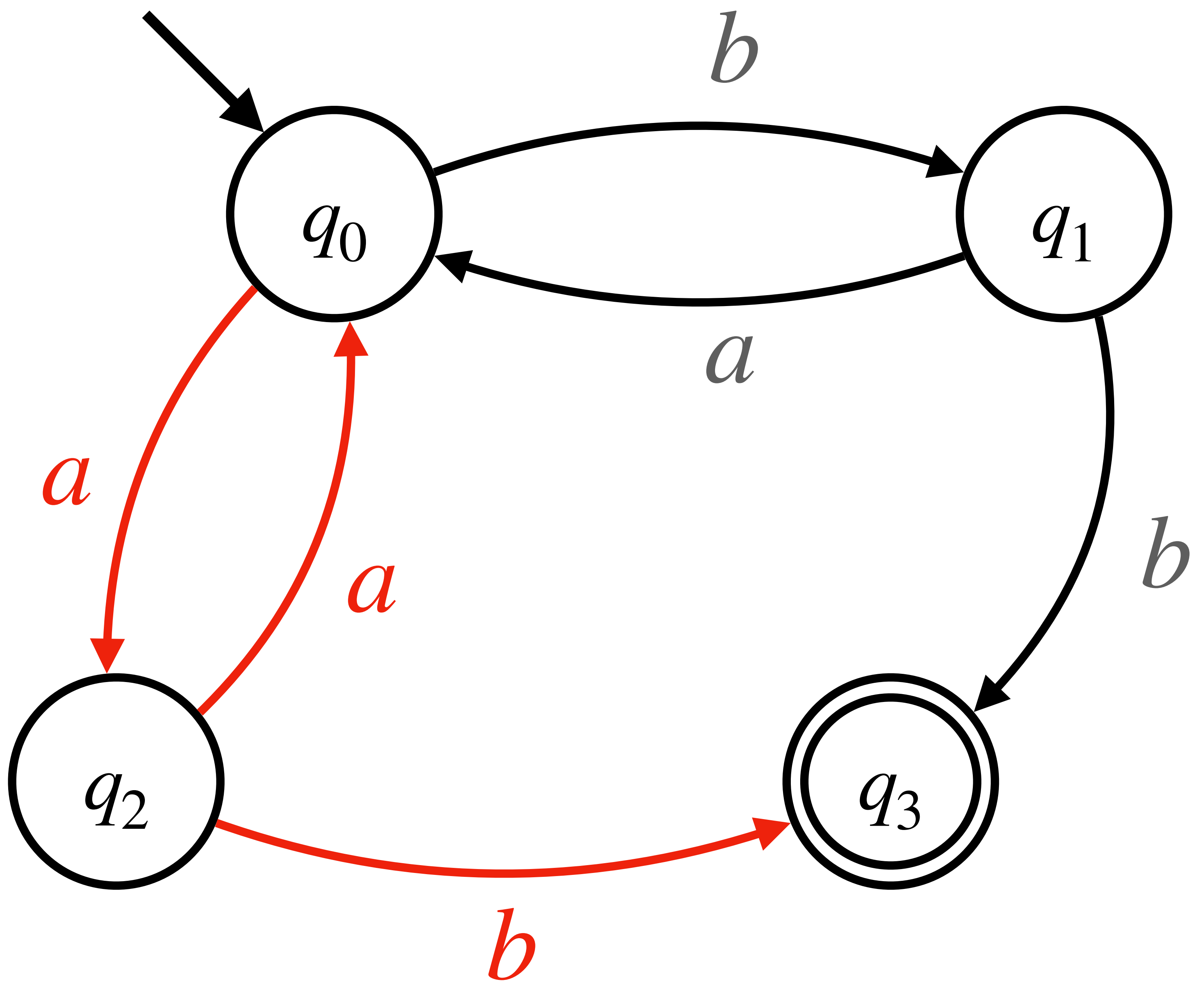


ab

(Kleene, 1956)

Regular Languages

$$X \to \{\bot, \top\} \times X^A$$

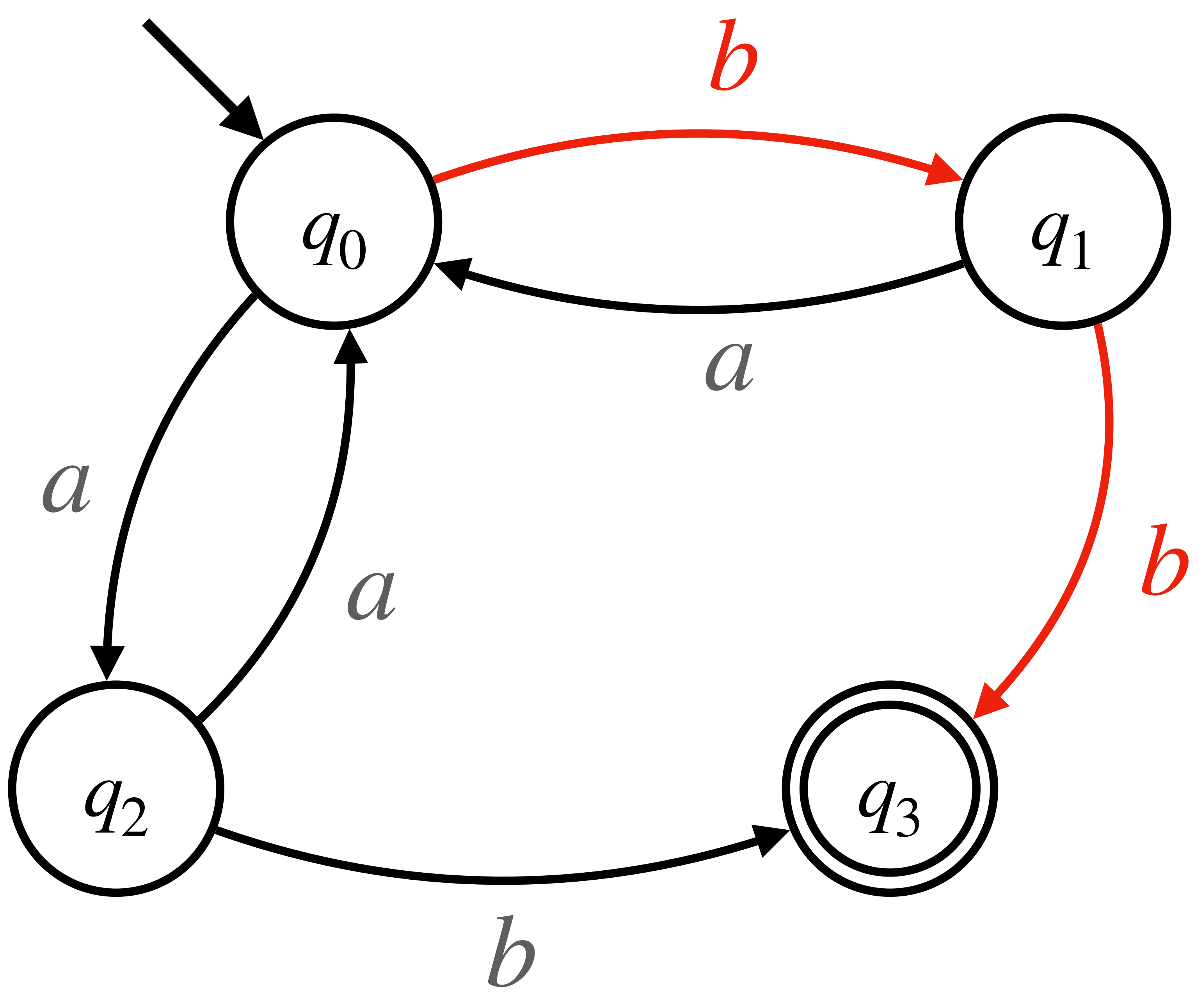


$$ab, aaab$$

(Kleene, 1956)

Regular Languages

$X \to \{\bot, \top\} \times X^A$

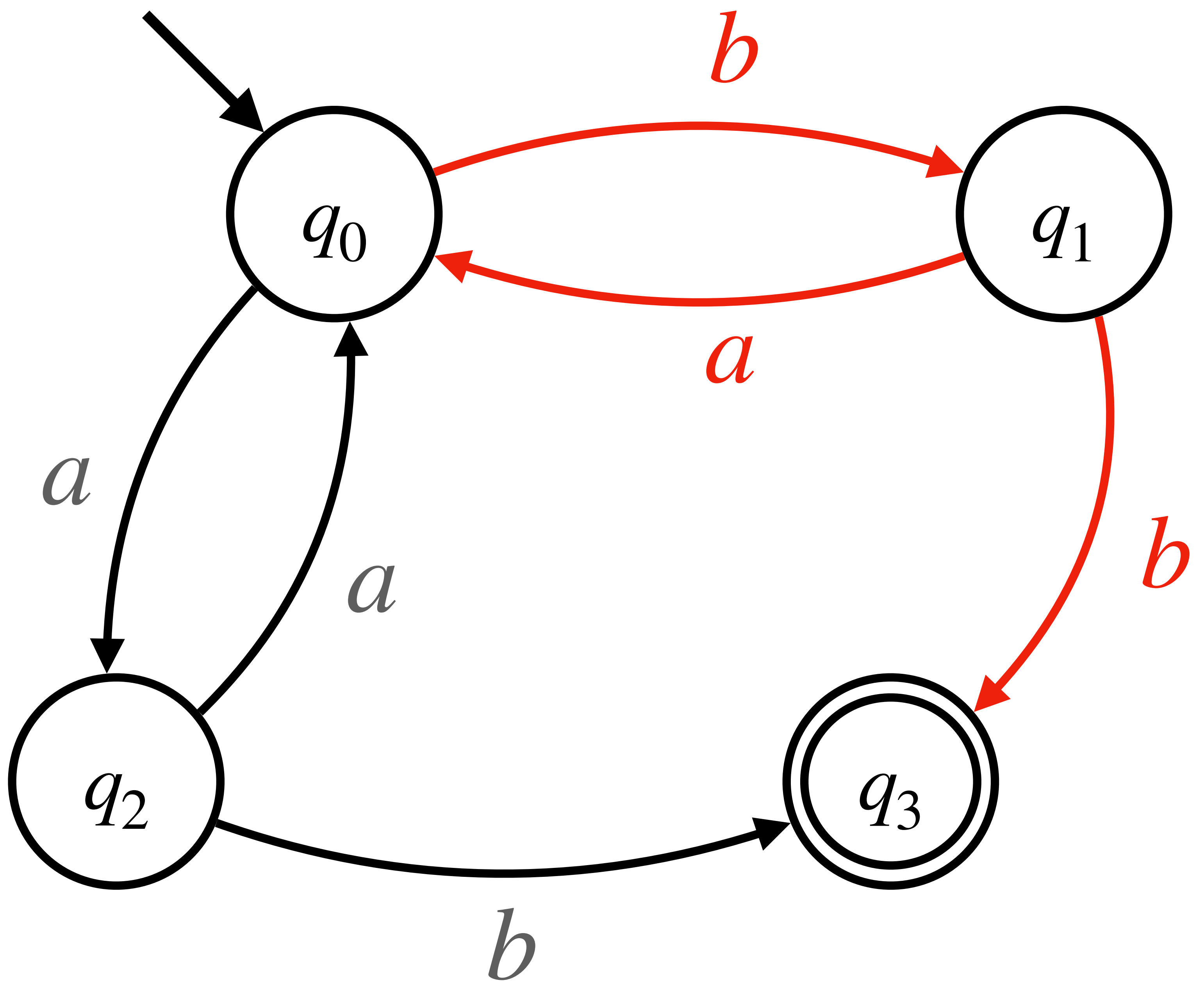


$ab, aaab, bb$

(Kleene, 1956)

Regular Languages

$$X \to \{\bot, \top\} \times X^A$$

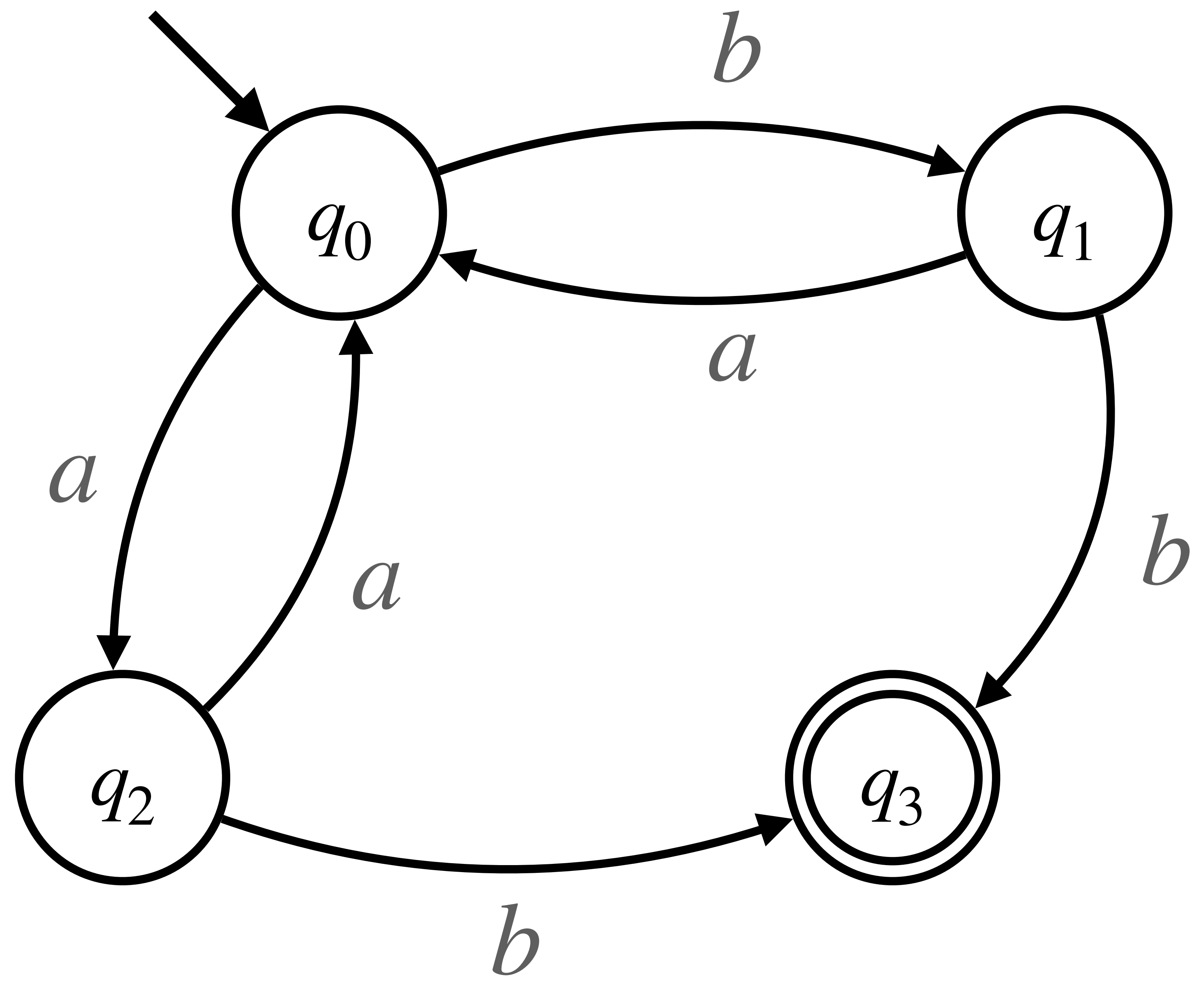


$$ab, aaab, bb, babb$$

(Kleene, 1956)

Regular Languages

$X \to \{\bot, \top\} \times X^A$

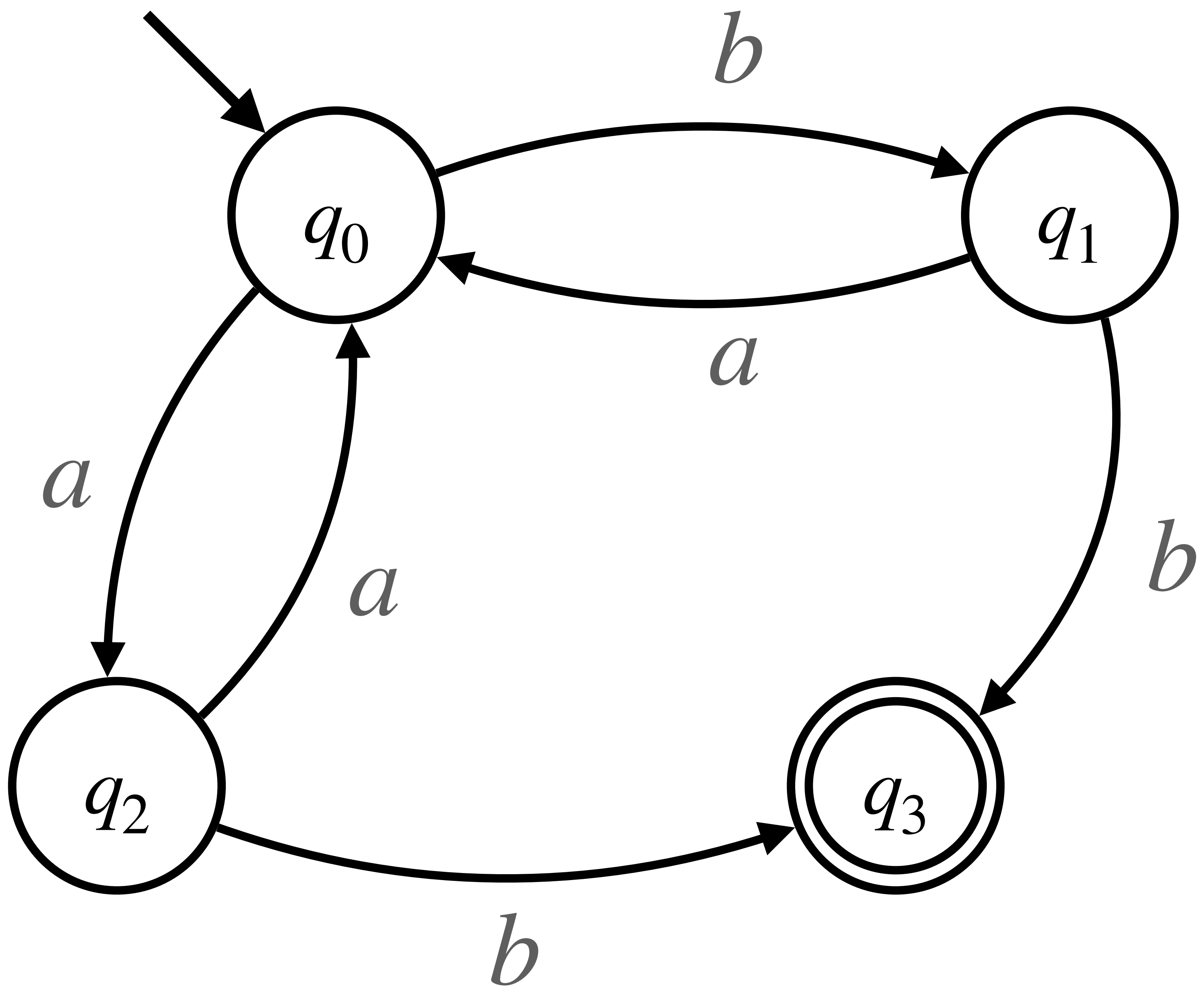


$$L = \{ab, aaab, bb, babb, \ldots\}$$

(Kleene, 1956)

Regular Languages

$X \to \{\bot, \top\} \times X^A$



Regular expressions: syntax for regular languages

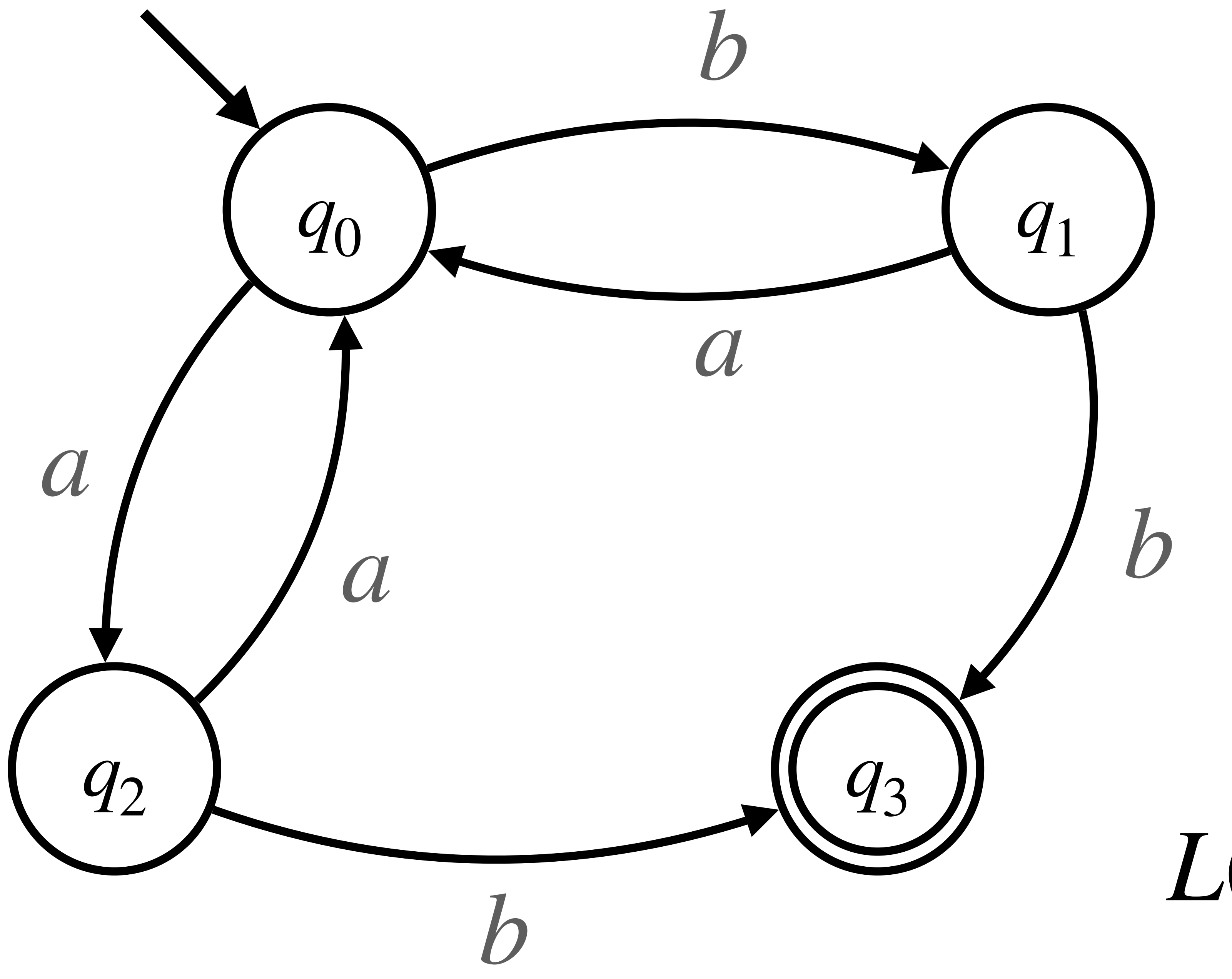
$$L = \{ab, aaab, bb, babb, \ldots\}$$

$$= (aa + ba)^*(ab + bb)$$

(Kleene, 1956)

Regular Expressions and Regular Languages

$(aa + ba)^*(ab + bb)$



$$\mathsf{RegEx} \ni e, f ::= 0 \mid 1 \mid p \in \Sigma \mid e + f \mid ef \mid e^*$$

$$L \colon \mathsf{RegEx} \longrightarrow \mathscr{P}(\Sigma^*)$$

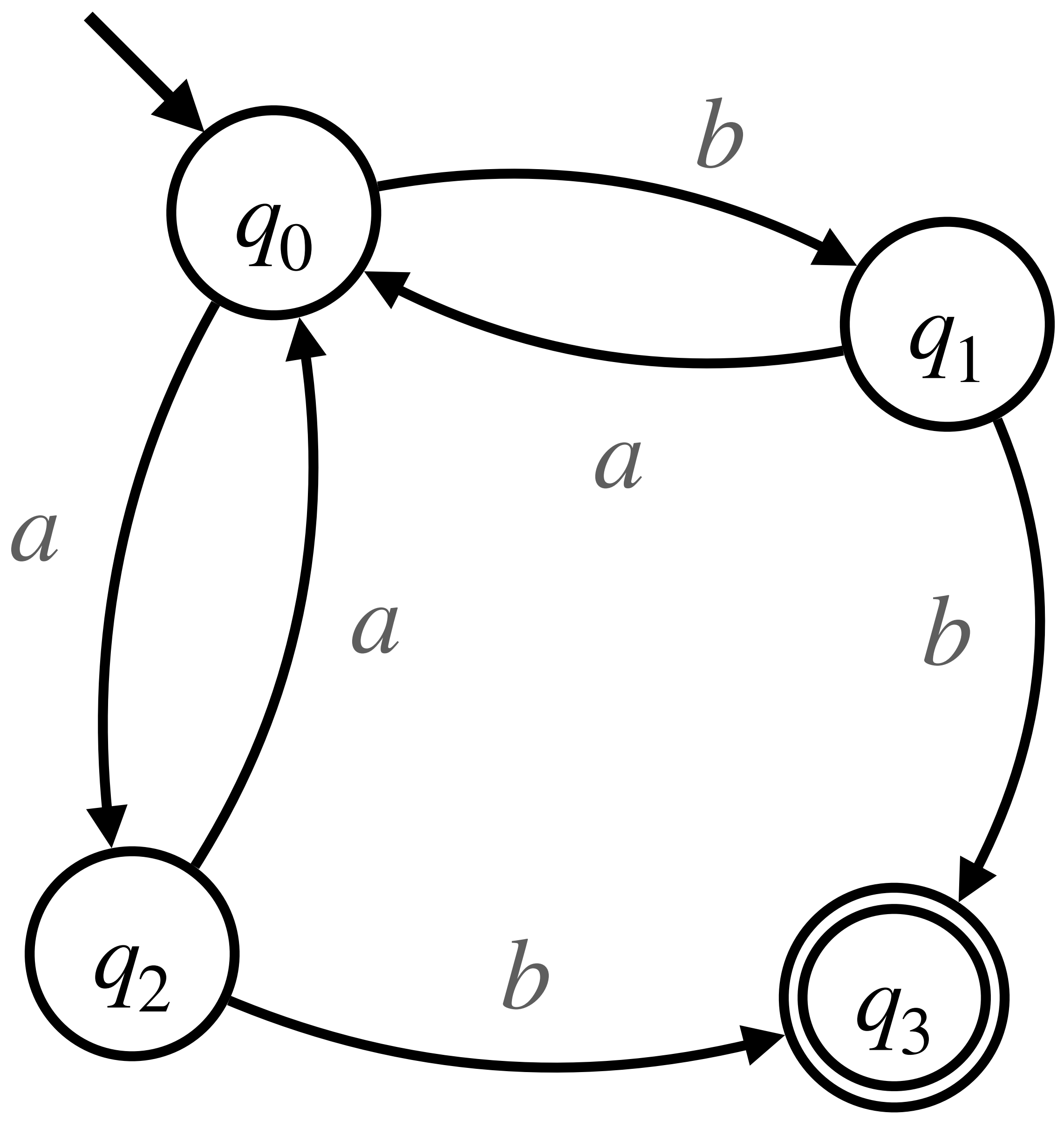
$$L(0) = \varnothing \quad L(1) = \{\varepsilon\} \quad L(p) = \{p\}$$

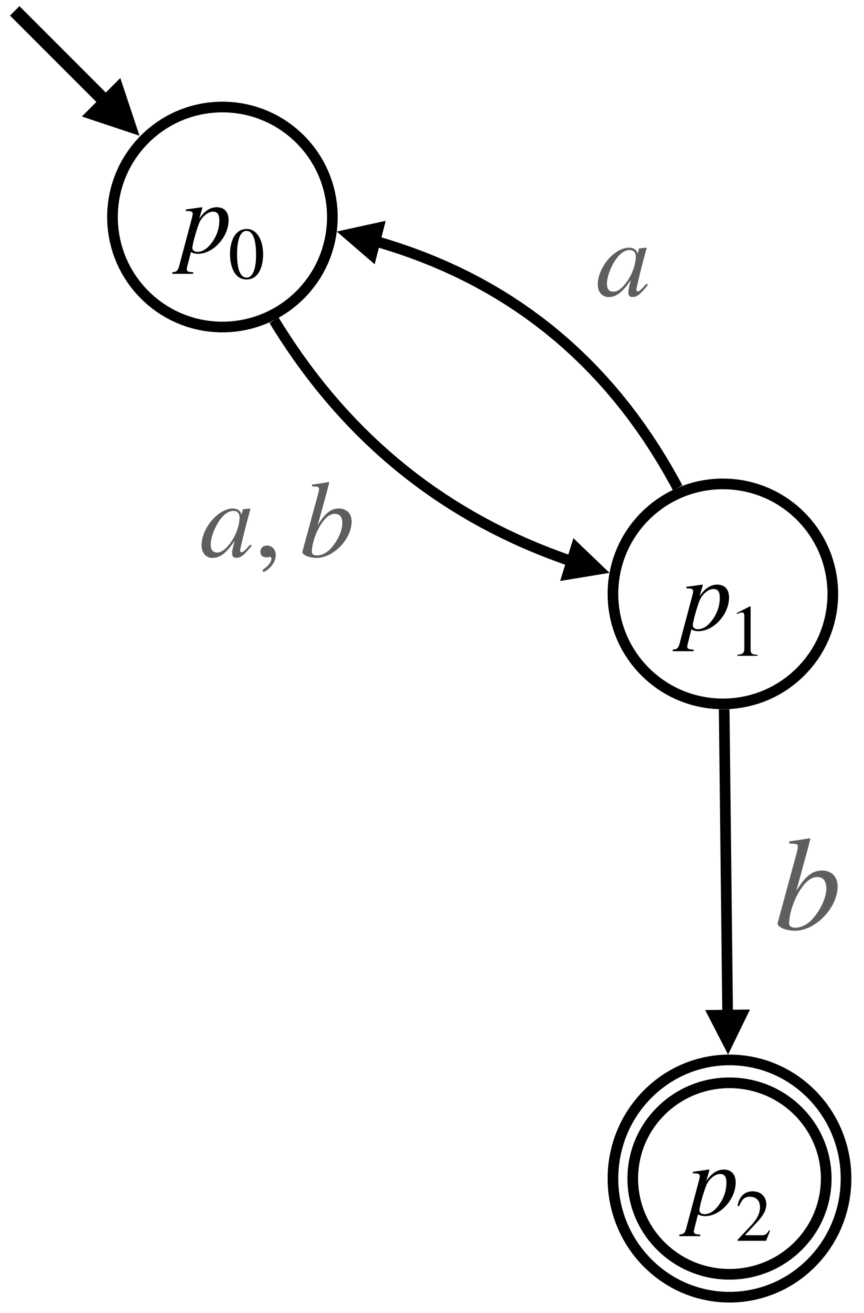
$$L(e+f) = L(e) \cup L(f) \quad L(ef) = L(e)L(f) \quad L(e^*) = \bigcup_{n \in \omega} L(e)^n$$

(Kleene, 1956)

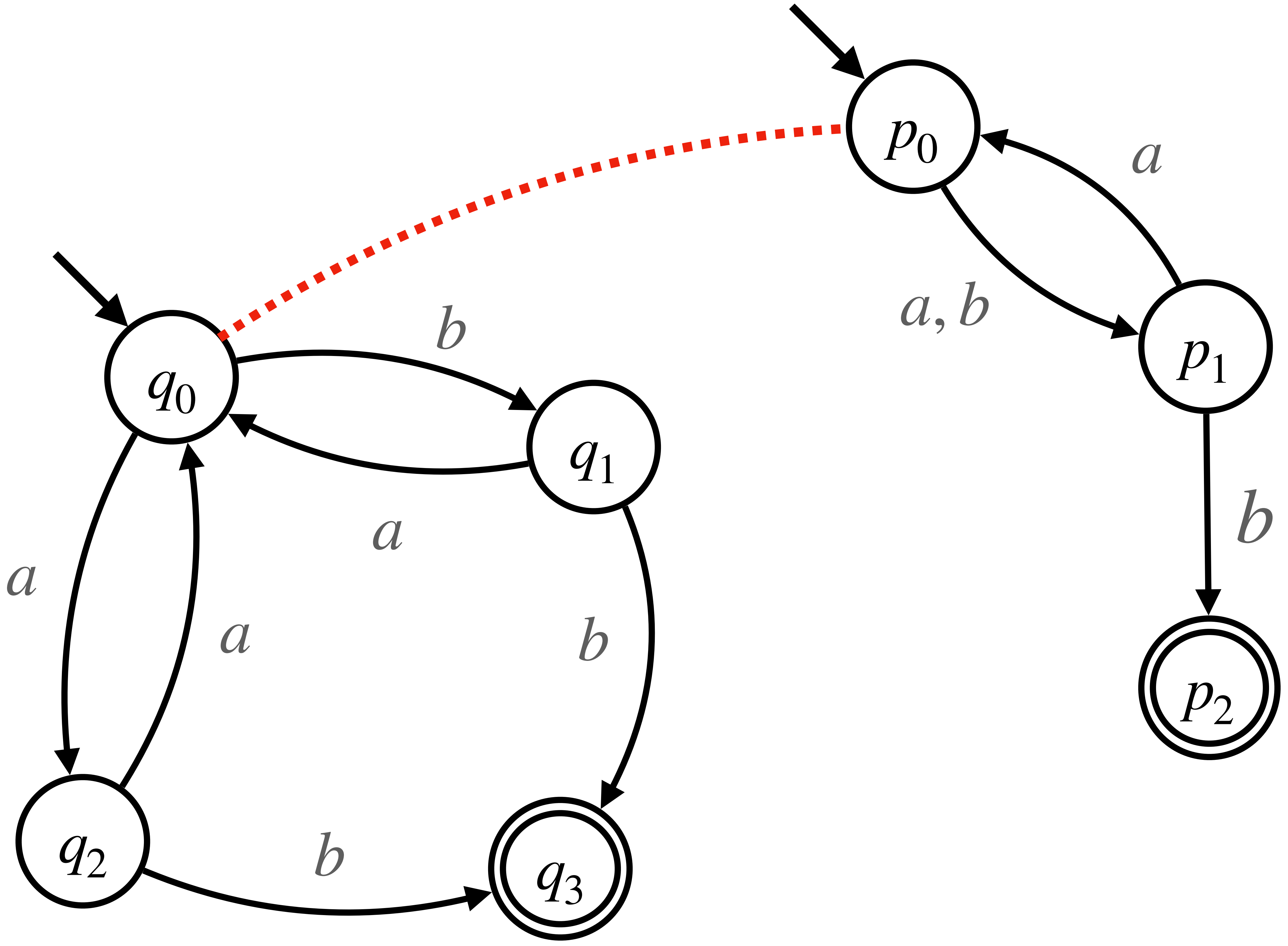
$L = L(r)$ iff L is recognized by a deterministic finite automaton.

Regular Expressions and Regular Languages

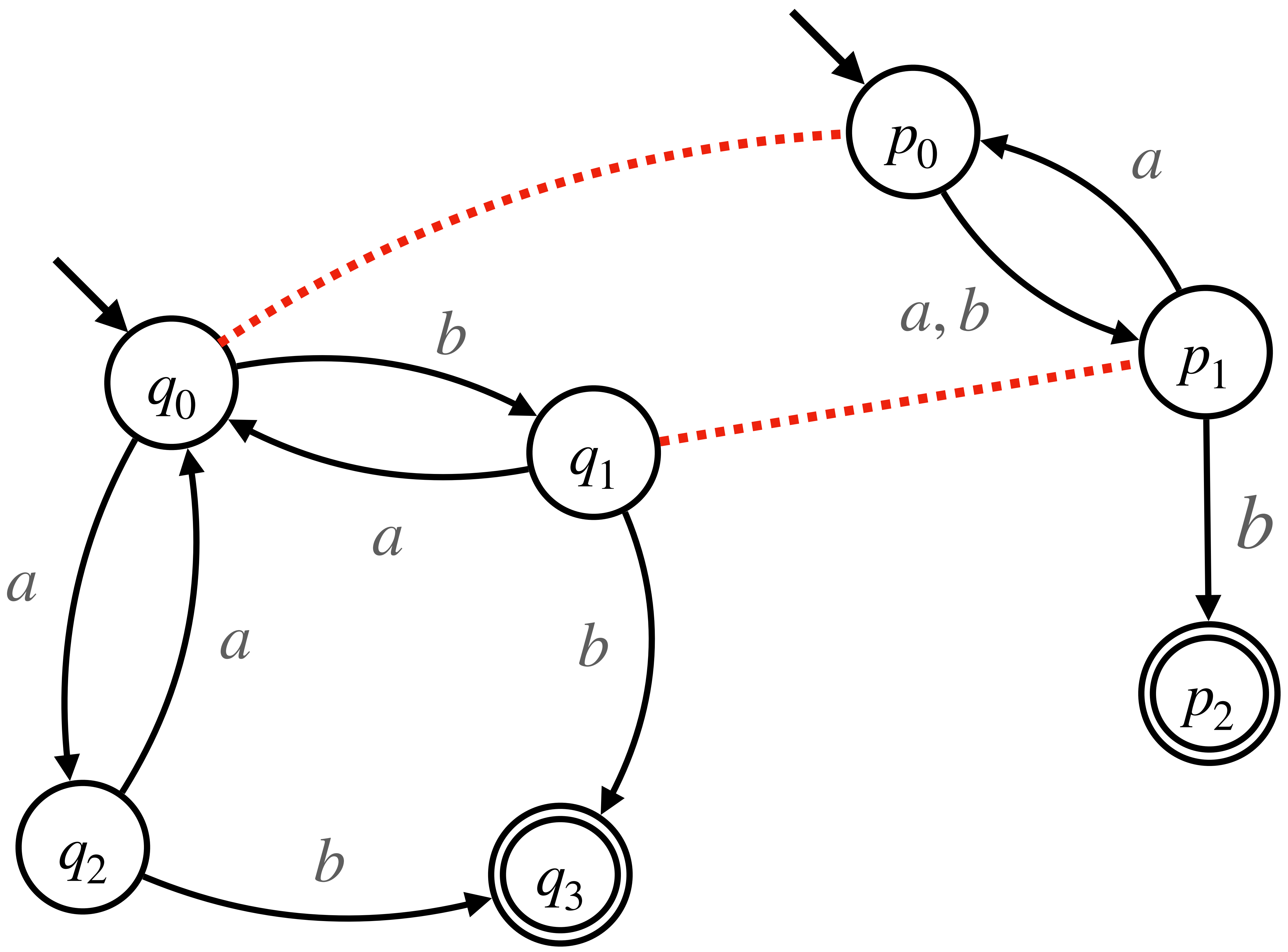




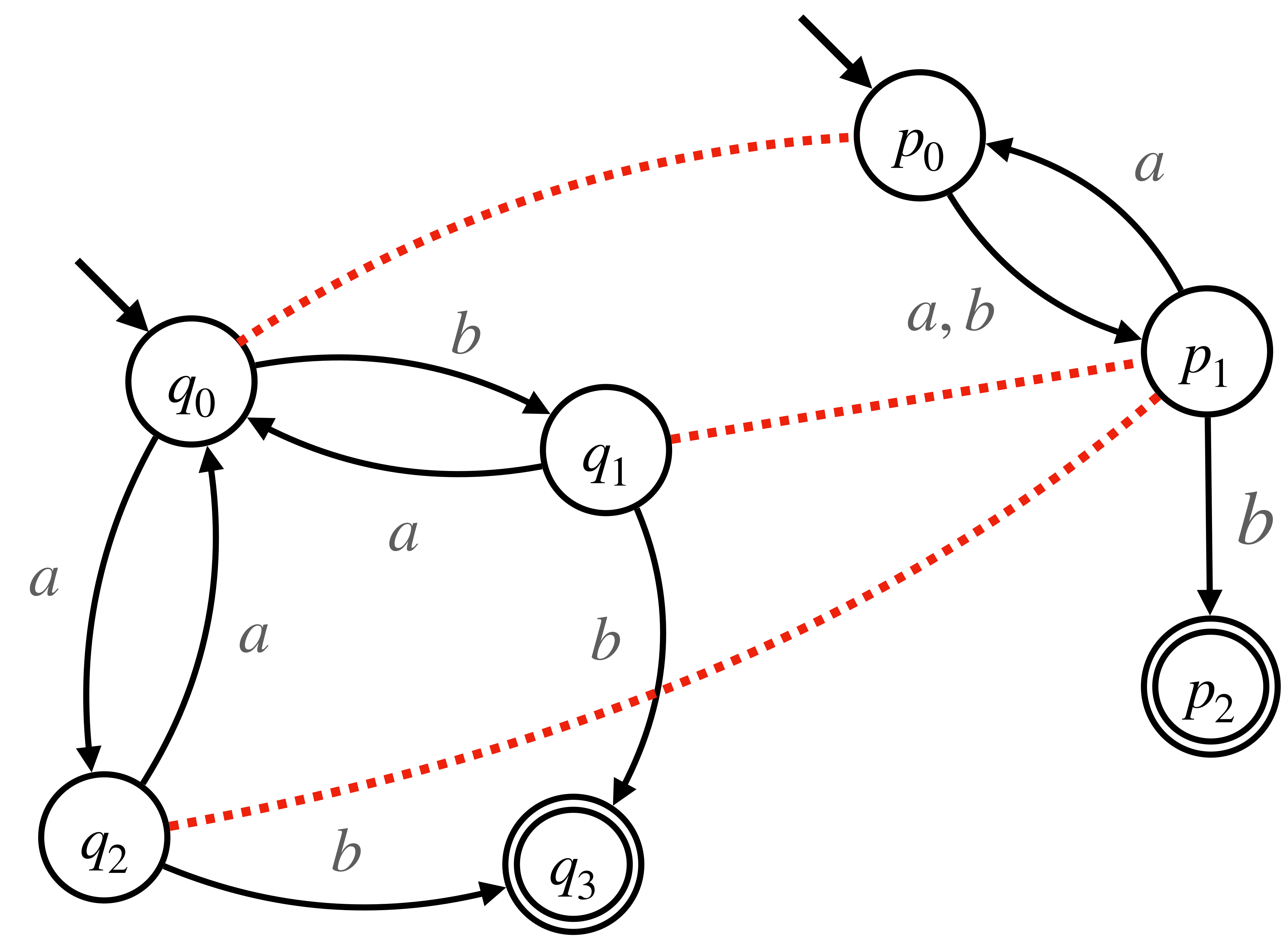
Regular Expressions and Regular Languages



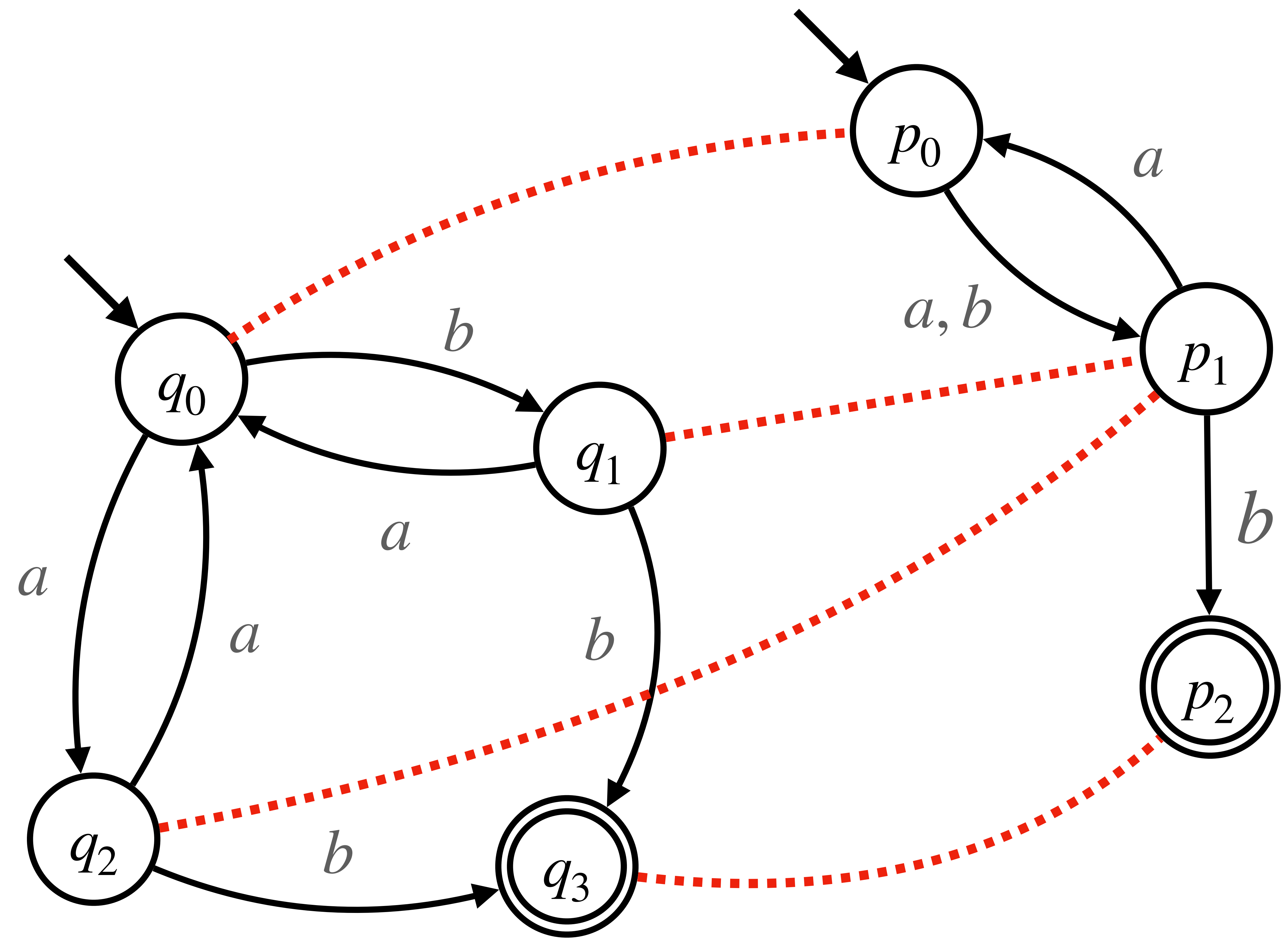
Regular Expressions and Regular Languages



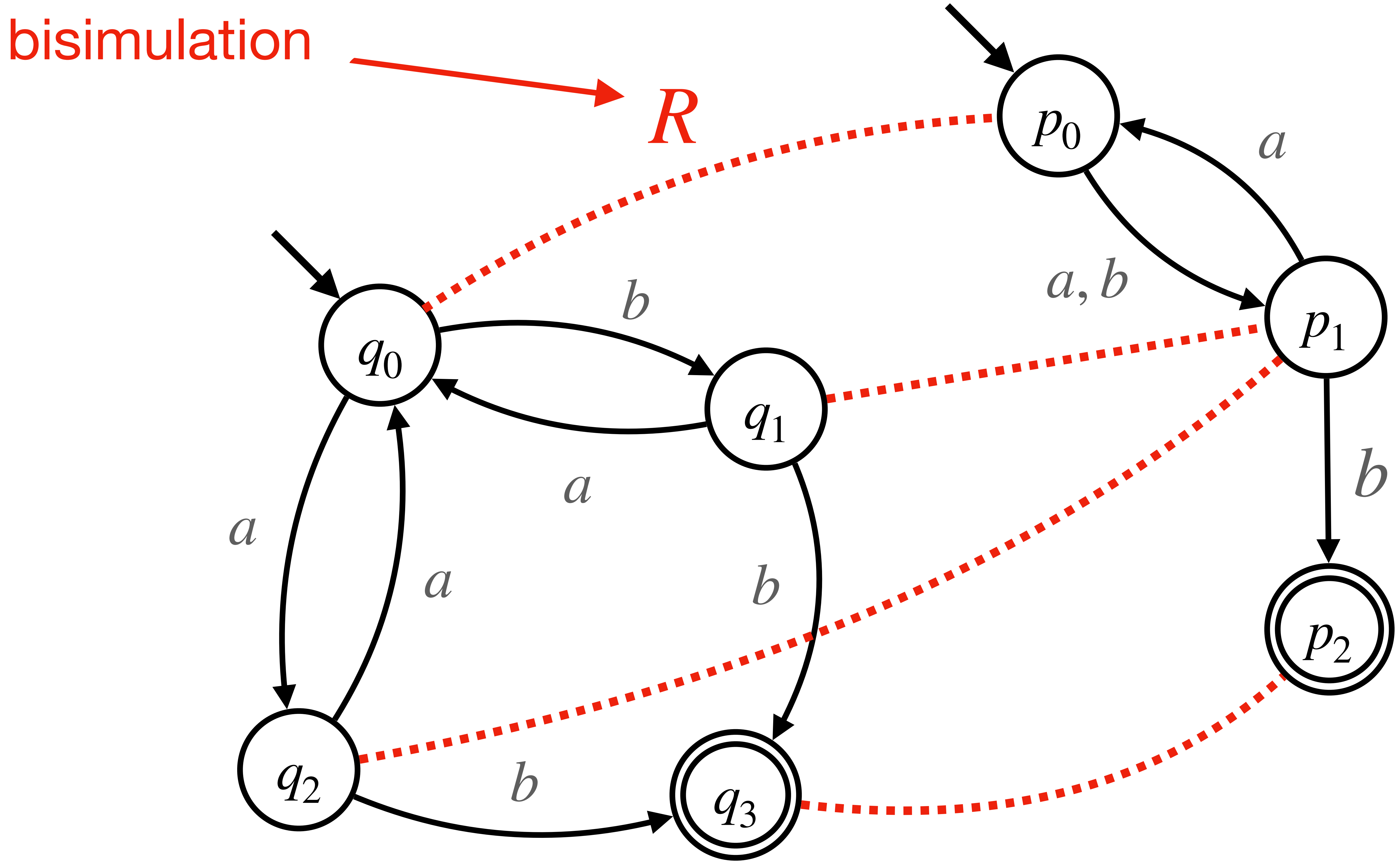
Regular Expressions and Regular Languages



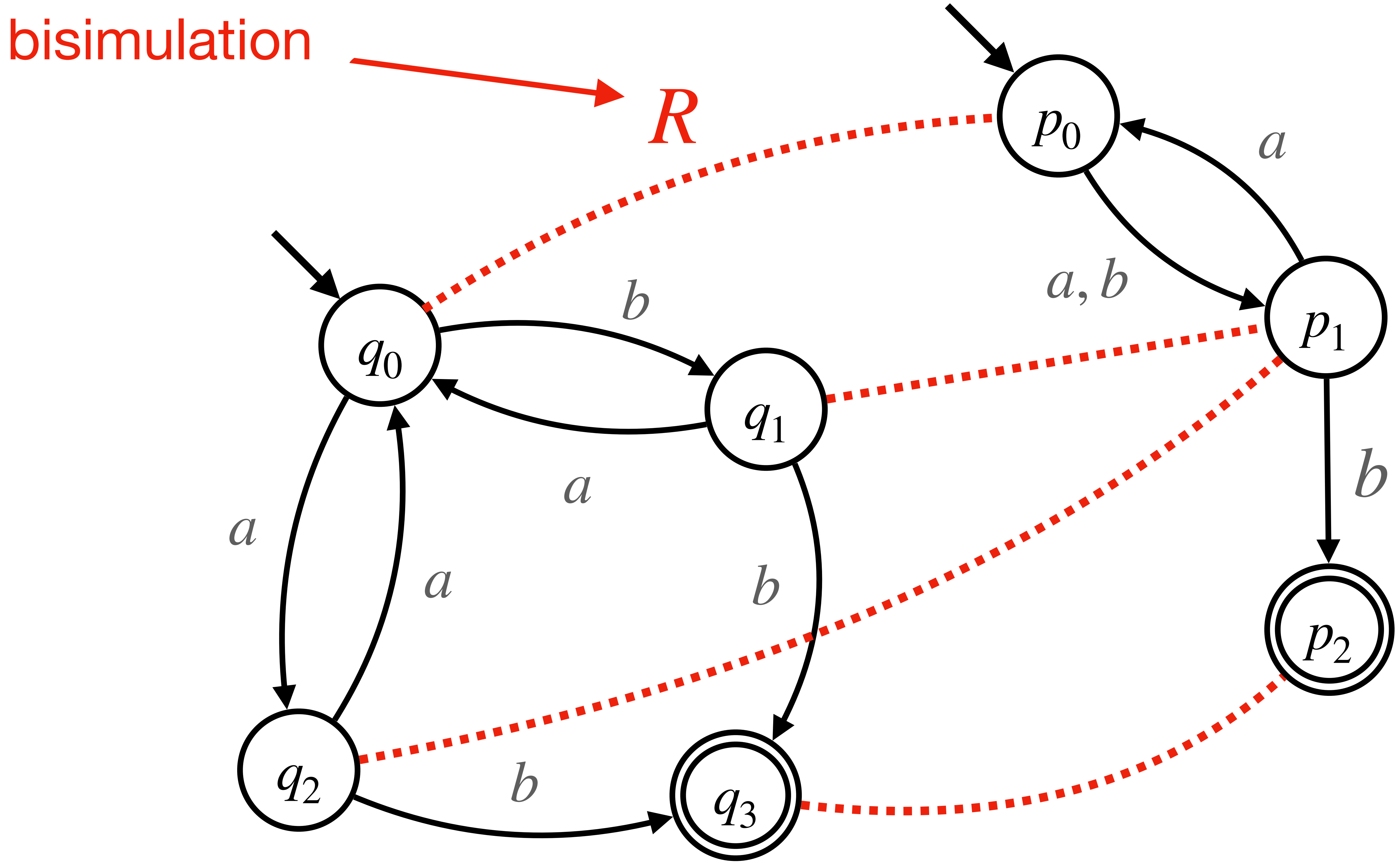
Regular Expressions and Regular Languages



Regular Expressions and Regular Languages

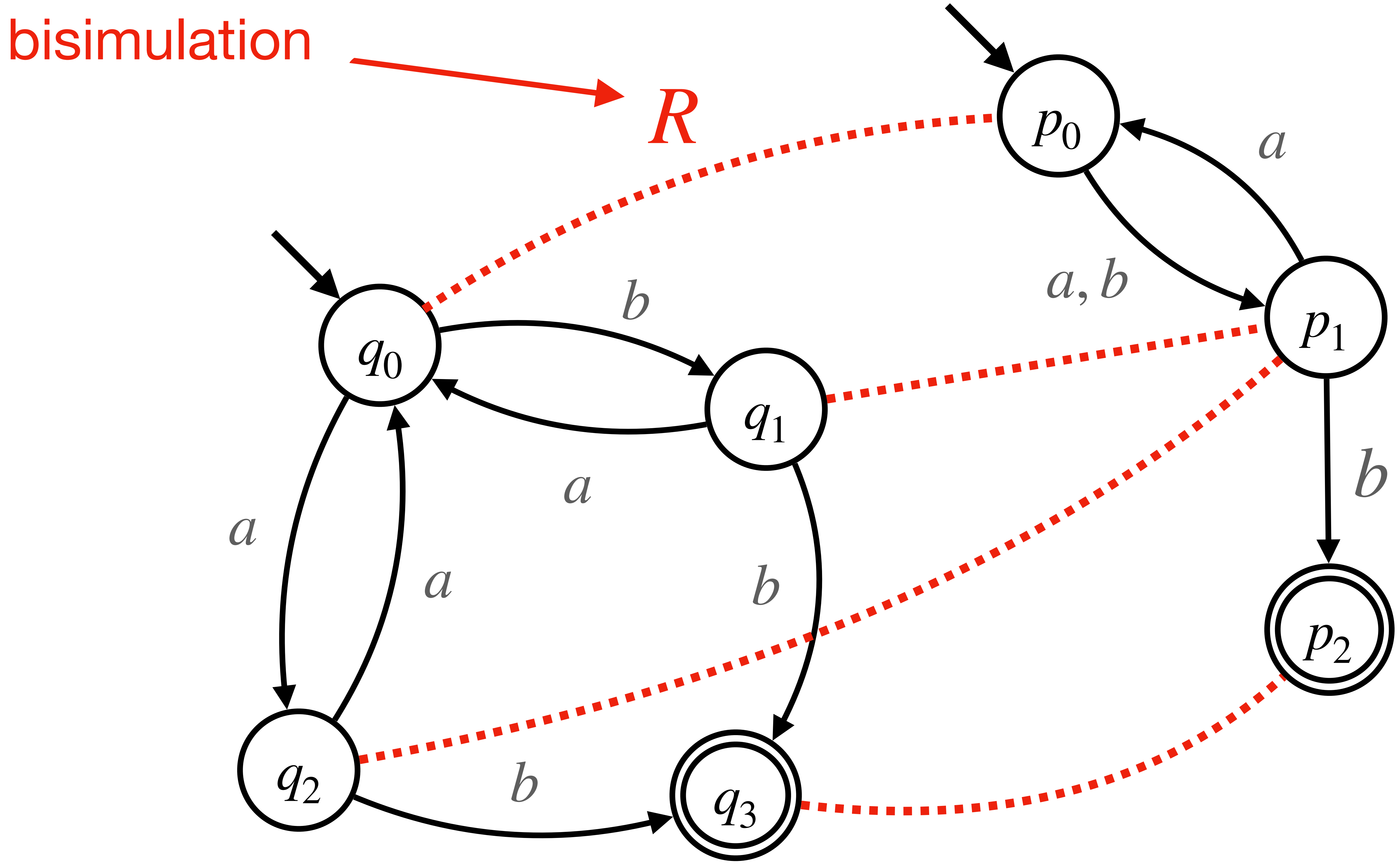


Regular Expressions and Regular Languages



For DFAs,

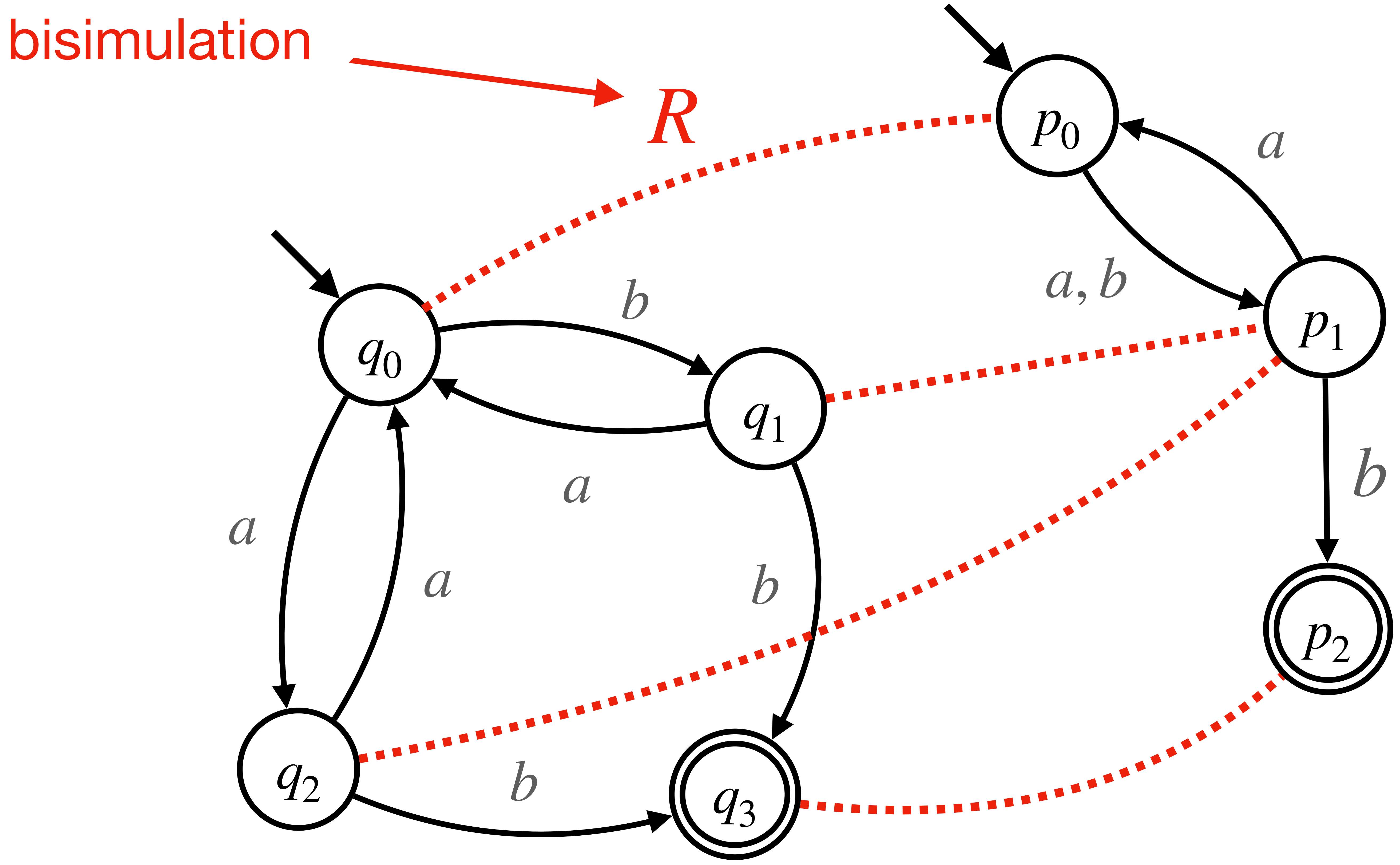
Regular Expressions and Regular Languages



For DFAs,

- **bisimilarity = language equivalence**

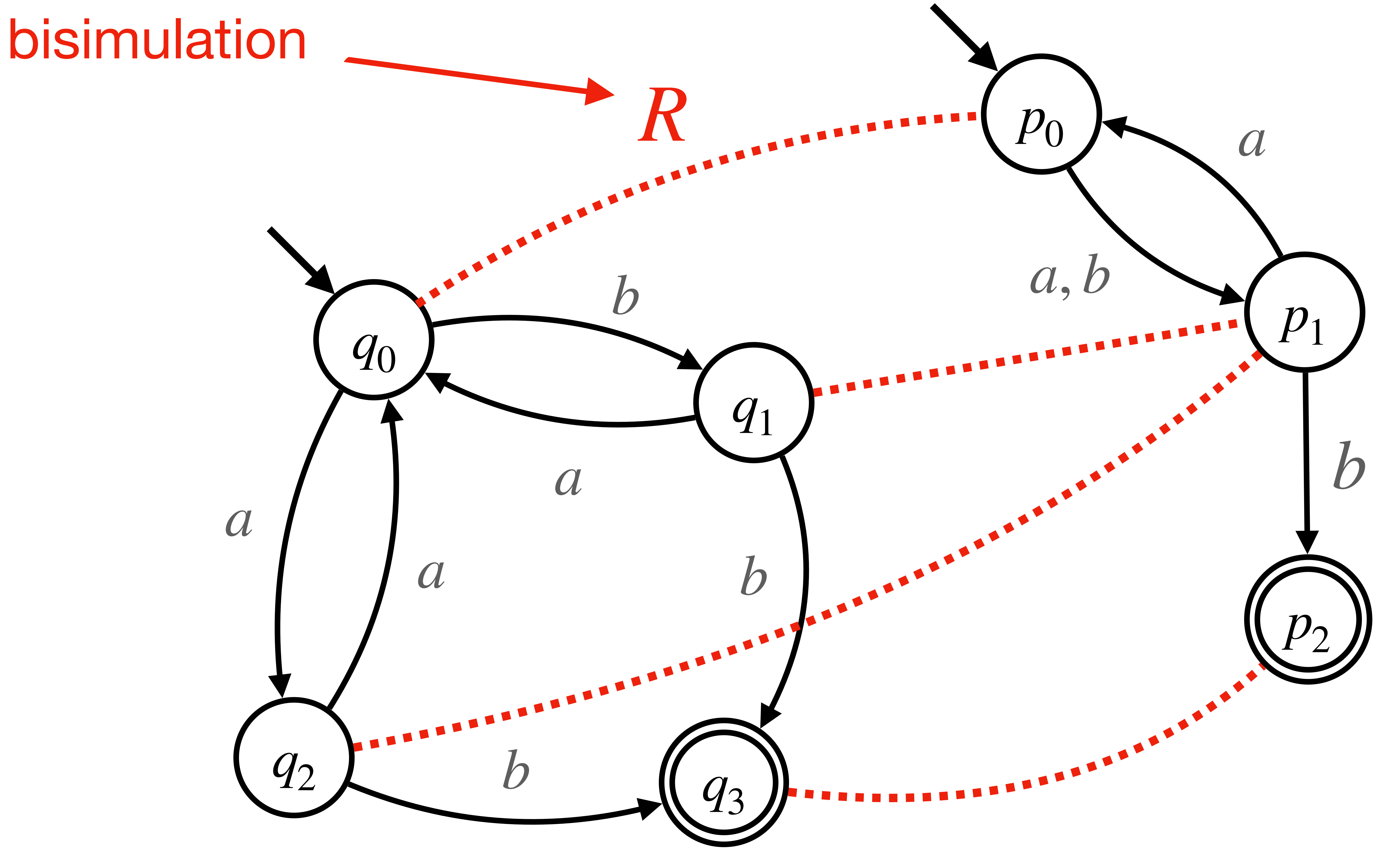
Regular Expressions and Regular Languages



For DFAs,

- bisimilarity = language equivalence
- Using (Hopcroft, Karp, 1971), bisimilarity is checked in almost linear time

Regular Expressions and Regular Languages

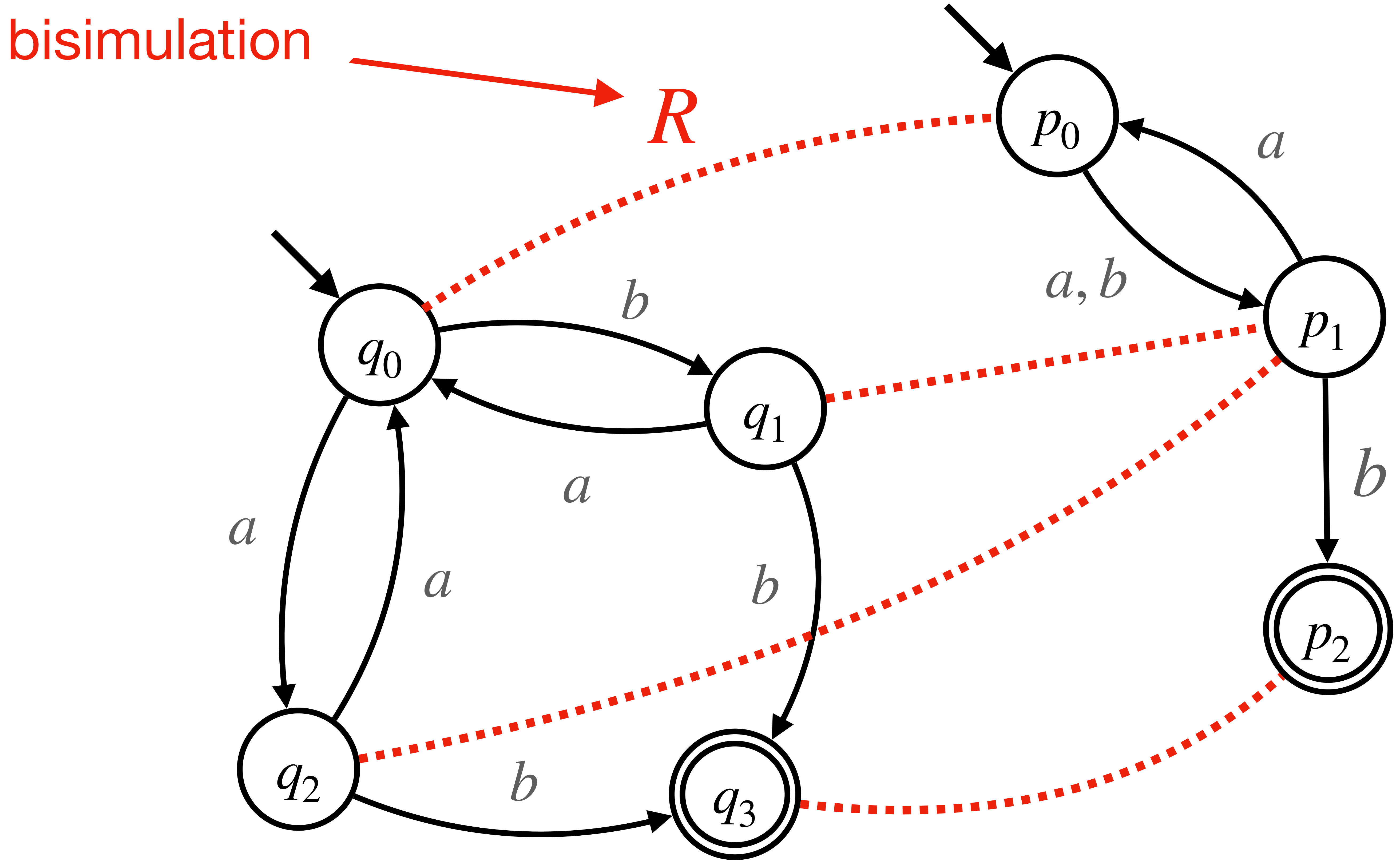


For DFAs,

- bisimilarity = language equivalence
- Using (Hopcroft, Karp, 1971), bisimilarity is checked in almost linear time

$$L((aa + ba)^*(ab + bb)) = L(((a + b)a)^*b)$$

Regular Expressions and Regular Languages



For DFAs,

- bisimilarity = language equivalence
- (Hopcroft, Karp, 1971) Bisimilarity is checked in nearly linear time

(Kleene, 1956)
Give a complete axiomatization of language equivalence of regular expressions

$$L((aa + ba)^*(ab + bb)) = L(((a + b)a)^*b)$$

$$\vdash (aa + ba)^*(ab + bb) = ((a + b)a)^*b \quad ?$$

Axiomatizing Language Equivalence

(Salomaa, 1964) A complete axiomatization of language equivalence of regular expressions:

A_1	$\alpha + (\beta + \gamma) = (\alpha + \beta) + \gamma,$	A_7	$\phi^* \alpha = \alpha,$
A_2	$\alpha(\beta\gamma) = (\alpha\beta)\gamma,$	A_8	$\phi\alpha = \phi,$
A_3	$\alpha + \beta = \beta + \alpha,$	A_9	$\alpha + \phi = \alpha,$
A_4	$\alpha(\beta + \gamma) = \alpha\beta + \alpha\gamma,$	A_{10}	$\alpha^* = \phi^* + \alpha^*\alpha,$
A_5	$(\alpha + \beta)\gamma = \alpha\gamma + \beta\gamma,$	A_{11}	$\alpha^* = (\phi^* + \alpha)^*.$
A_6	$\alpha + \alpha = \alpha,$		

R1 (Substitution). Assume that γ' is the result of replacing an occurrence of α by β in γ. Then from the equations $\alpha = \beta$ and $\gamma = \delta$ one may infer the equation $\gamma' = \delta$ and the equation $\gamma' = \gamma$.

R2 (Solution of equations). Assume that β does not possess e.w.p. Then from the equation $\alpha = \alpha\beta + \gamma$ one may infer the equation $\alpha = \gamma\beta^*$.

Axiomatizing Language Equivalence

(Milner, 1984) Rephrased Salomaa's rules as follows:

Salomaa [9] provides a complete inference system for star expressions under standard interpretation. When we dualise it, by writing $f \circ e$ for $e \circ f$ everywhere in Salomaa's rules (which gives an equipotent system), it has the following rules:

A_1	$e + (f + g) = (e + f) + g$	A_7	$e \circ \phi^* = e$
A_2	$(e \circ f) \circ g = e \circ (f \circ g)$	A_8	$e \circ \phi = \phi$
A_3	$e + f = f + e$	A_9	$e + \phi = e$
A_4	$(e + f) \circ g = e \circ g + f \circ g$	A_{10}	$e^* = \phi^* + e \circ e^*$
A_5	$e \circ (f + g) = e \circ f + e \circ g$	A_{11}	$e^* = (\phi^* + e)^*$
A_6	$e + e = e$		

R_2 *If* f does not possess e.w.p. then

$$\text{from } e = f \circ e + h \text{ infer } e = f^* \circ h.$$

(We have omitted R_1, the substitution rule.)

Axiomatizing Language Equivalence

(Milner, 1984) Rephrased Salomaa's rules as follows:

Salomaa [9] provides a complete inference system for star expressions under standard interpretation. When we dualise it, by writing $f \circ e$ for $e \circ f$ everywhere in Salomaa's rules (which gives an equipotent system), it has the following rules:

A_1 $e + (f + g) = (e + f) + g$

A_2 $(e \circ f) \circ g = e \circ (f \circ g)$

A_3 $e + f = f + e$

A_4 $(e + f) \circ g = e \circ g + f \circ g$

A_5 $e \circ (f + g) = e \circ f + e \circ g$

A_6 $e + e = e$

A_7 $e \circ \phi^* = e$

A_8 $e \circ \phi = \phi$

A_9 $e + \phi = e$

A_{10} $e^* = \phi^* + e \circ e^*$

A_{11} $e^* = (\phi^* + e)^*$

R_2 *If f* does not possess e.w.p. then

$$\text{from } e = f \circ e + h \text{ infer } e = f^* \circ h.$$

(We have omitted R_1, the substitution rule.)

Milner rephrased Salomaa's axioms to make them easier to adapt to a different (process) semantics.

Deciding language equivalence

$(aa + ba)^*(ab + bb)$

Regular Expressions

Deciding language equivalence

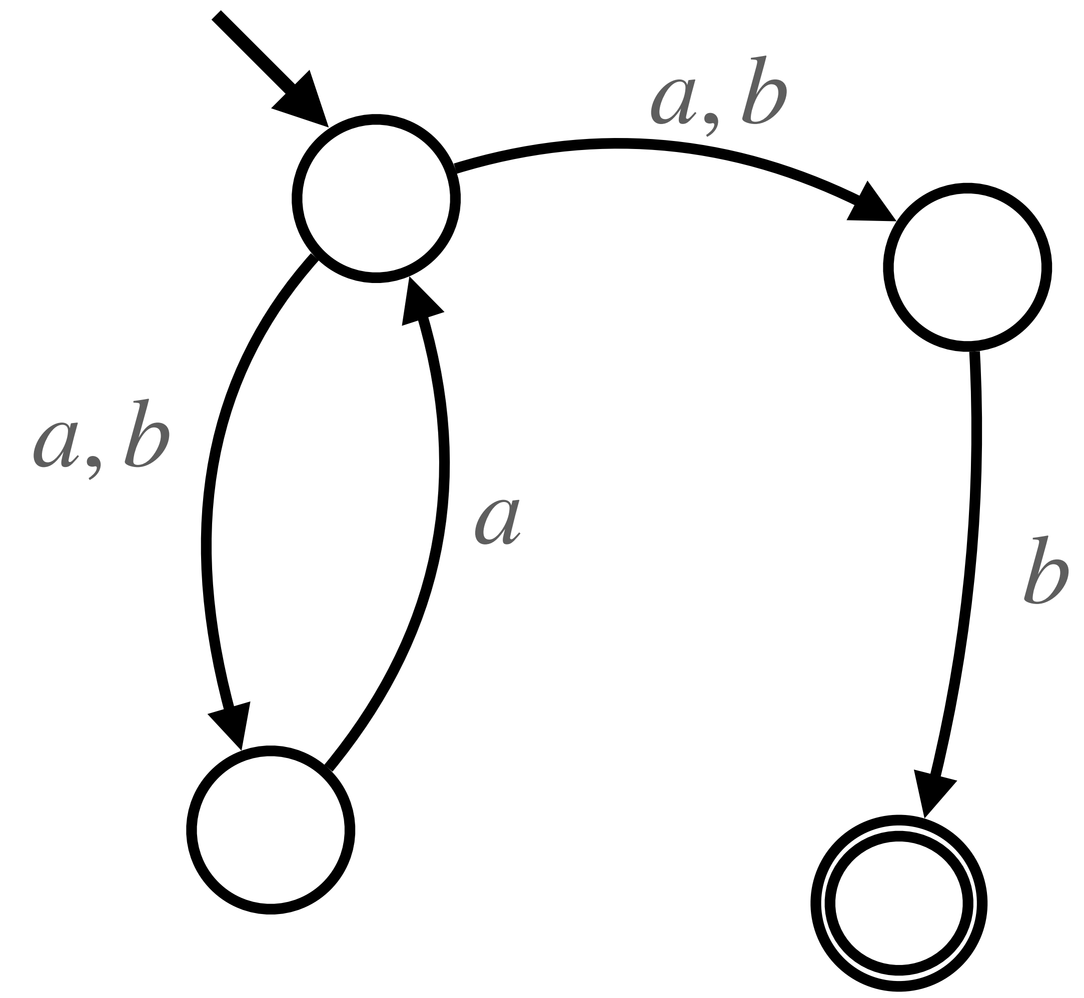
$(aa + ba)^*(ab + bb)$

Regular Expressions →

Thompson Construction,
SOS,
Antimirov Derivatives

Deciding language equivalence

$(aa + ba)^*(ab + bb)$



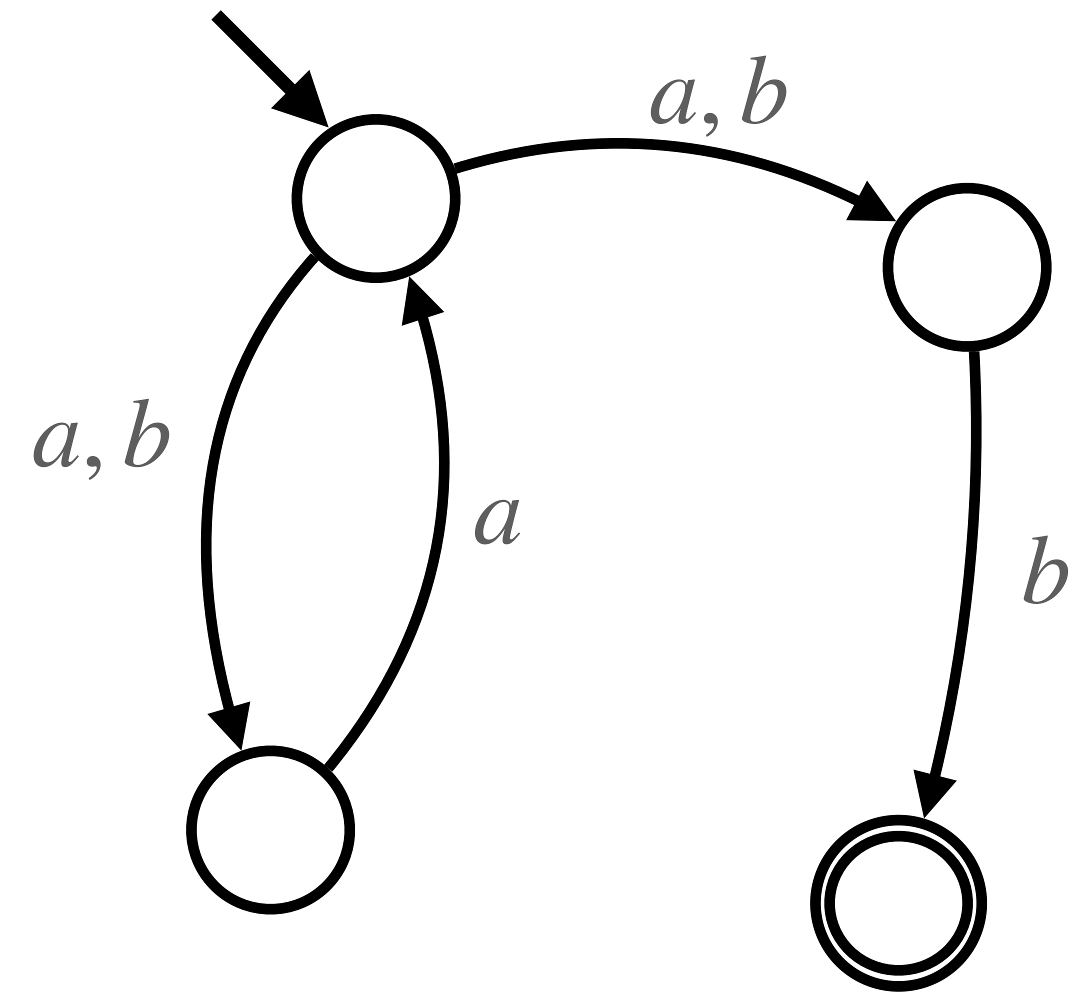
$$X \to \{\bot, \top\} \times \mathscr{P}(X)^A$$

Regular Expressions ➡ Nondeterministic FAs

Thompson Construction,
SOS,
Antimirov Derivatives

Deciding language equivalence

$(aa + ba)^*(ab + bb)$



$X \to \{\bot, \top\} \times \mathscr{P}(X)^A$

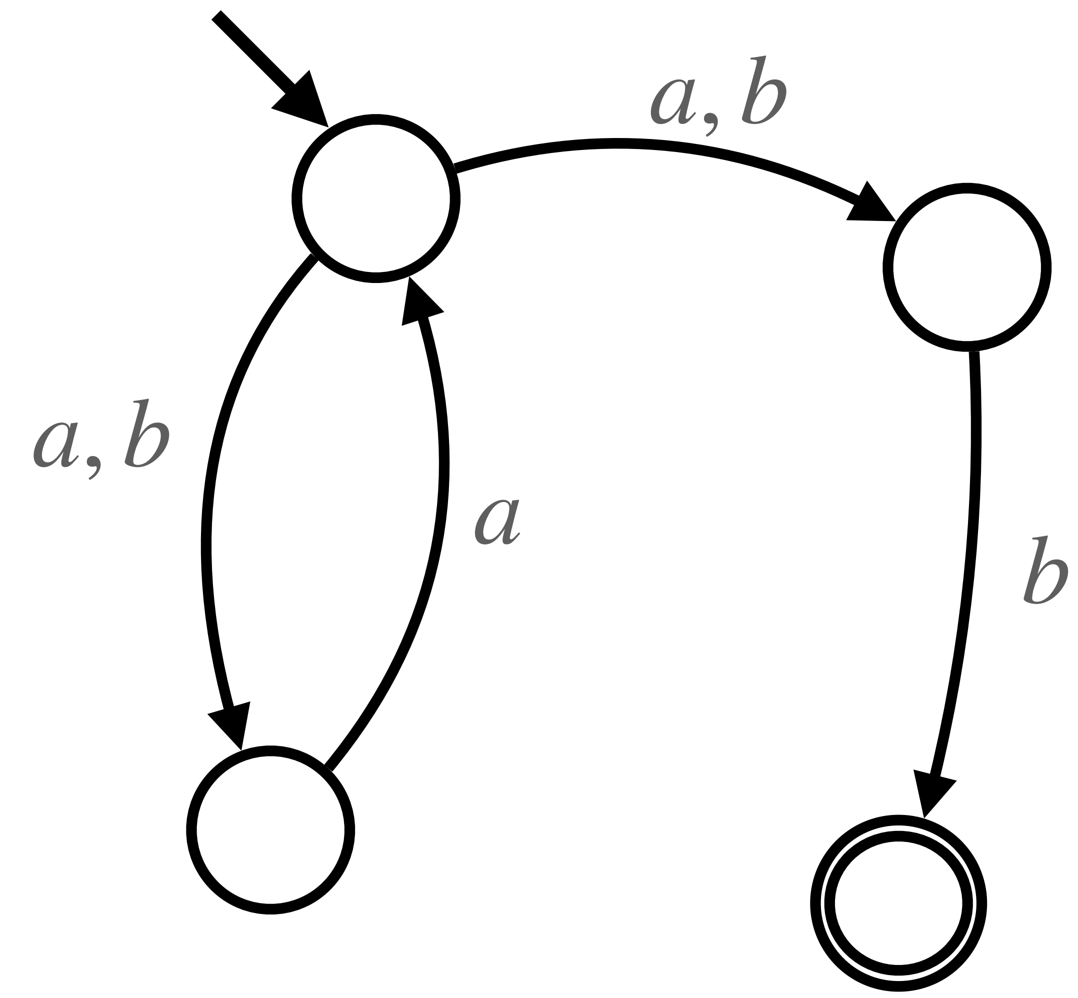
(Determinize)

Regular Expressions → Nondeterministic FAs →

Thompson Construction,
SOS,
Antimirov Derivatives

Deciding language equivalence

$(aa + ba)^*(ab + bb)$



$X \to \{\bot, \top\} \times \mathscr{P}(X)^A$

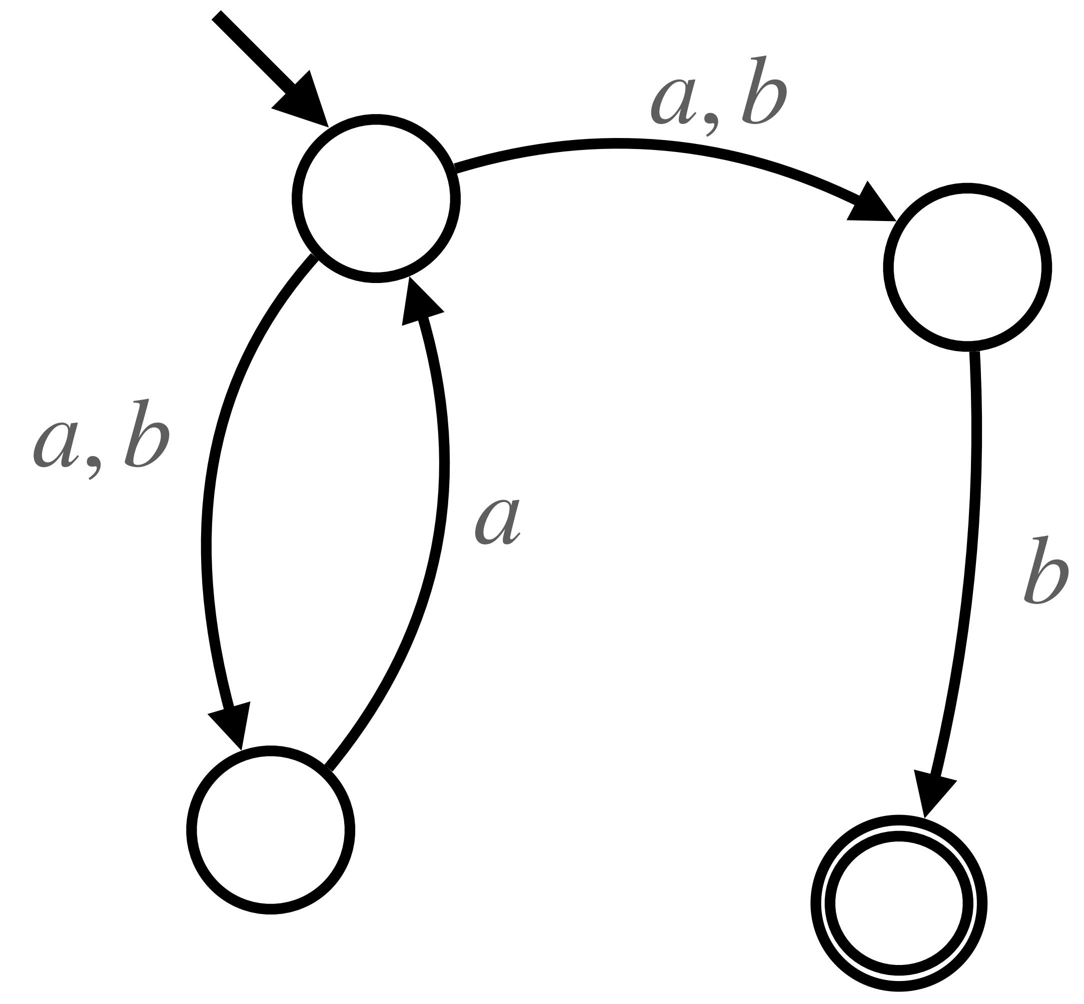
(Determinize)

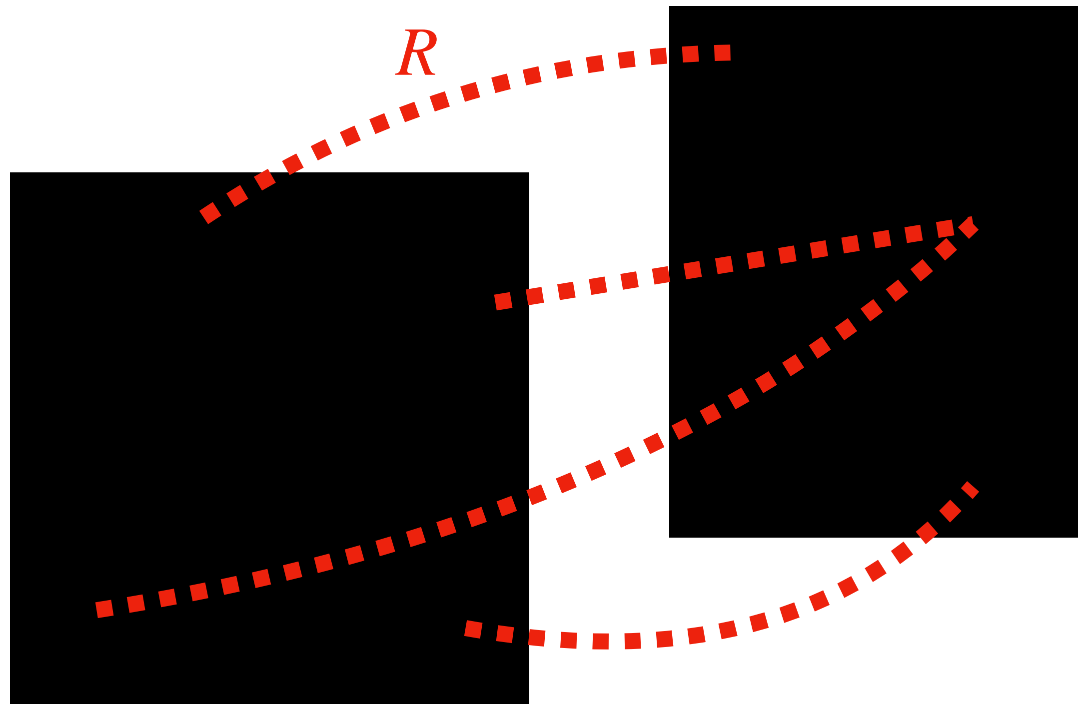
Regular Expressions → Nondeterministic FAs → DFAs

Thompson Construction,
SOS,
Antimirov Derivatives

Deciding language equivalence

$(aa + ba)^*(ab + bb)$





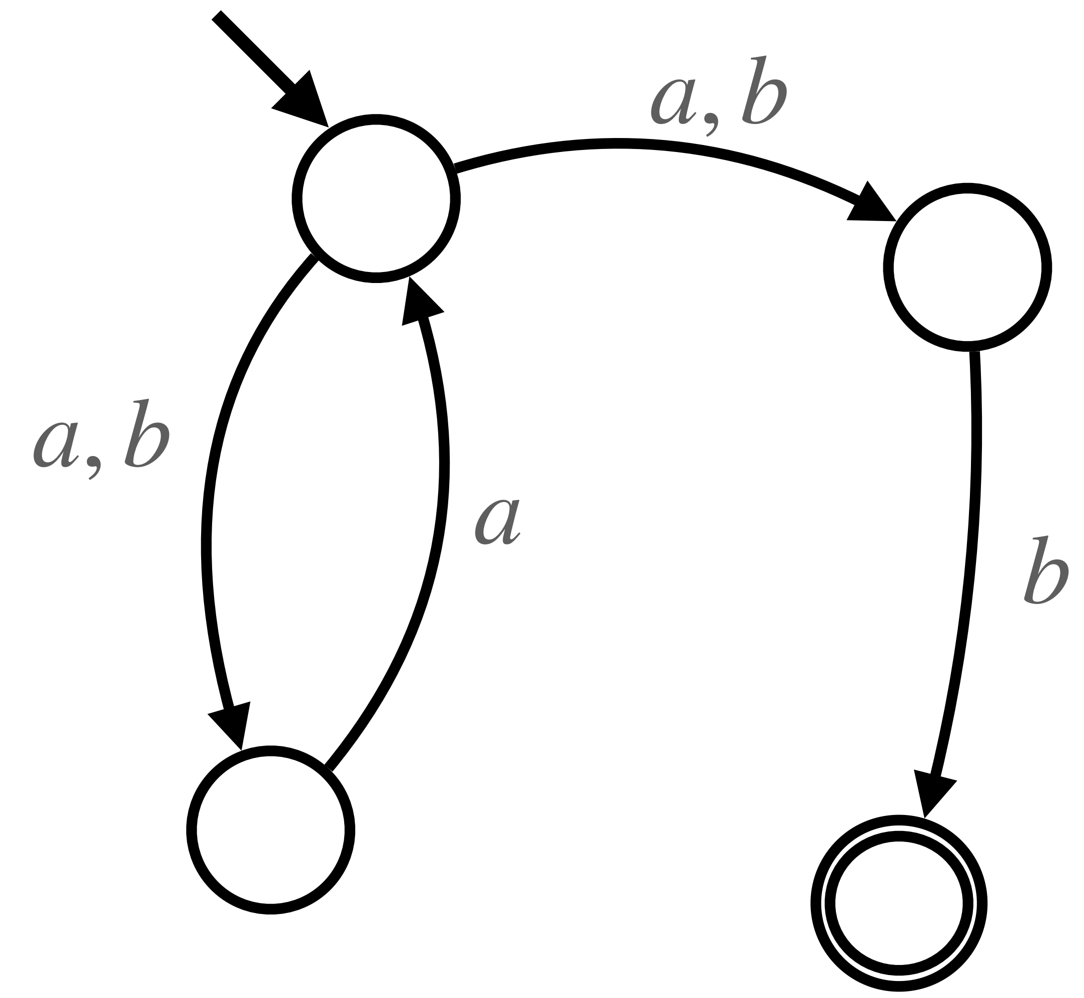
$X \to \{\bot, \top\} \times \mathscr{P}(X)^A$

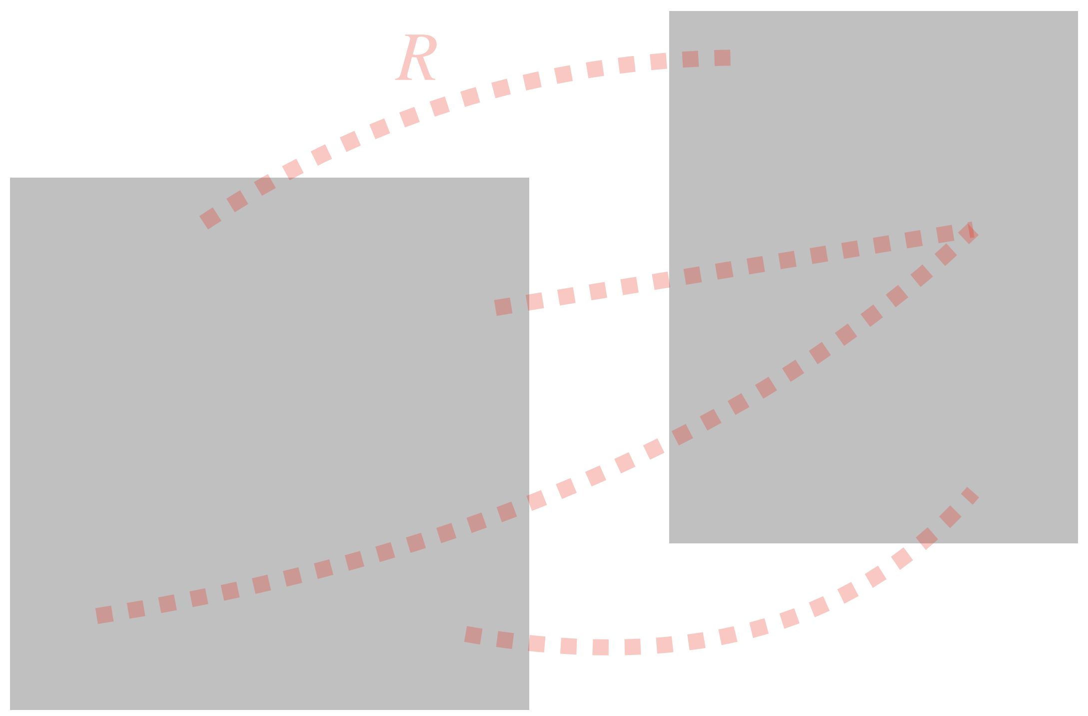
Regular Expressions → Nondeterministic FAs → (Determinize) DFAs → Check for Bisimilarity

Thompson Construction,
SOS,
Antimirov Derivatives

Deciding language equivalence

$(aa + ba)^*(ab + bb)$





$X \to \{\bot, \top\} \times \mathscr{P}(X)^A$

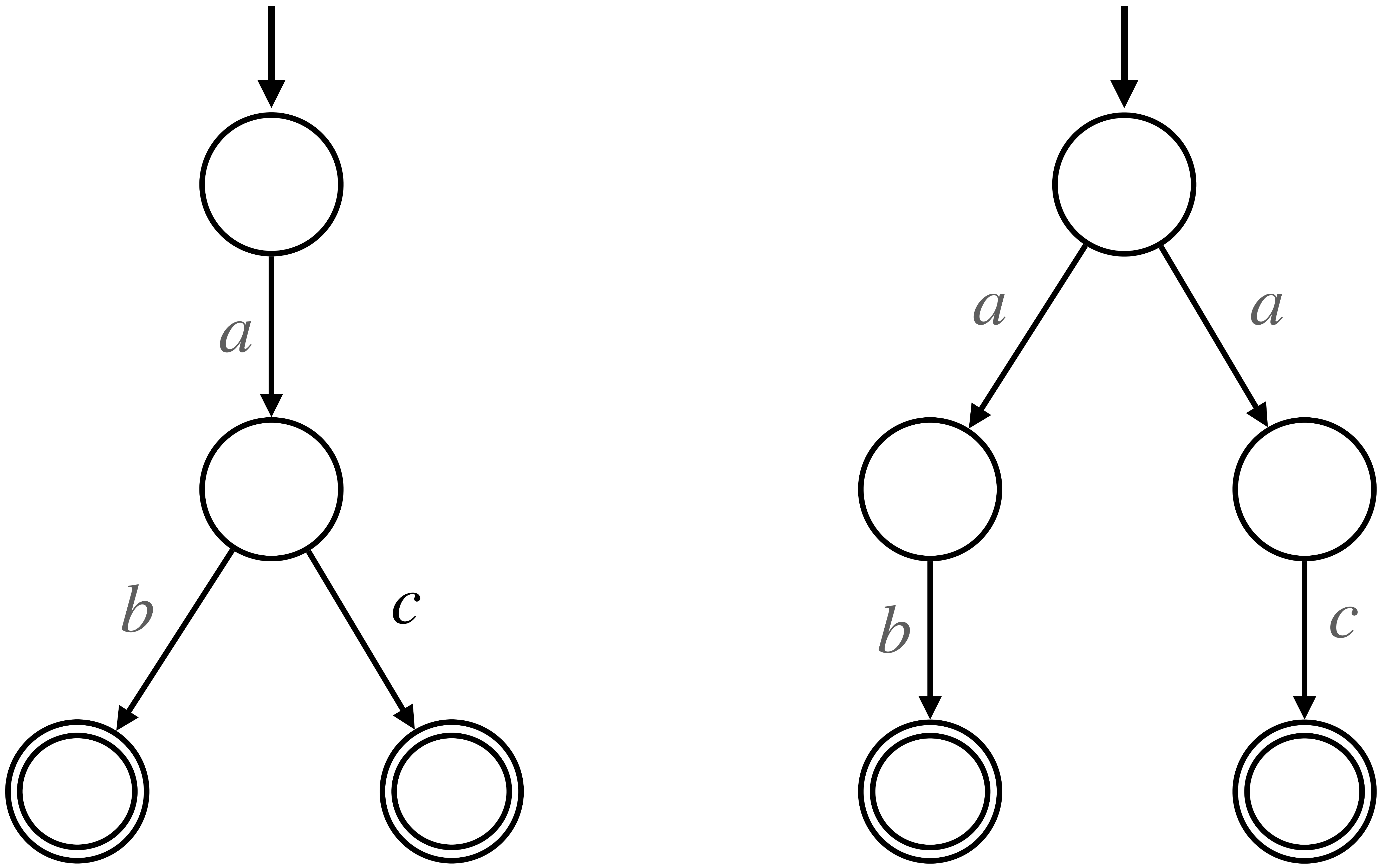
Regular Expression → Nondeterministic FAs → (Determinize) DFAs → Check for Bisimilarity

Thompson Construction, Operational Semantics, Antimirov Derivatives

Bisimilarity here?

(Milner, 1984)

Bisimilarity for NFAs is Finer than Language Equivalence



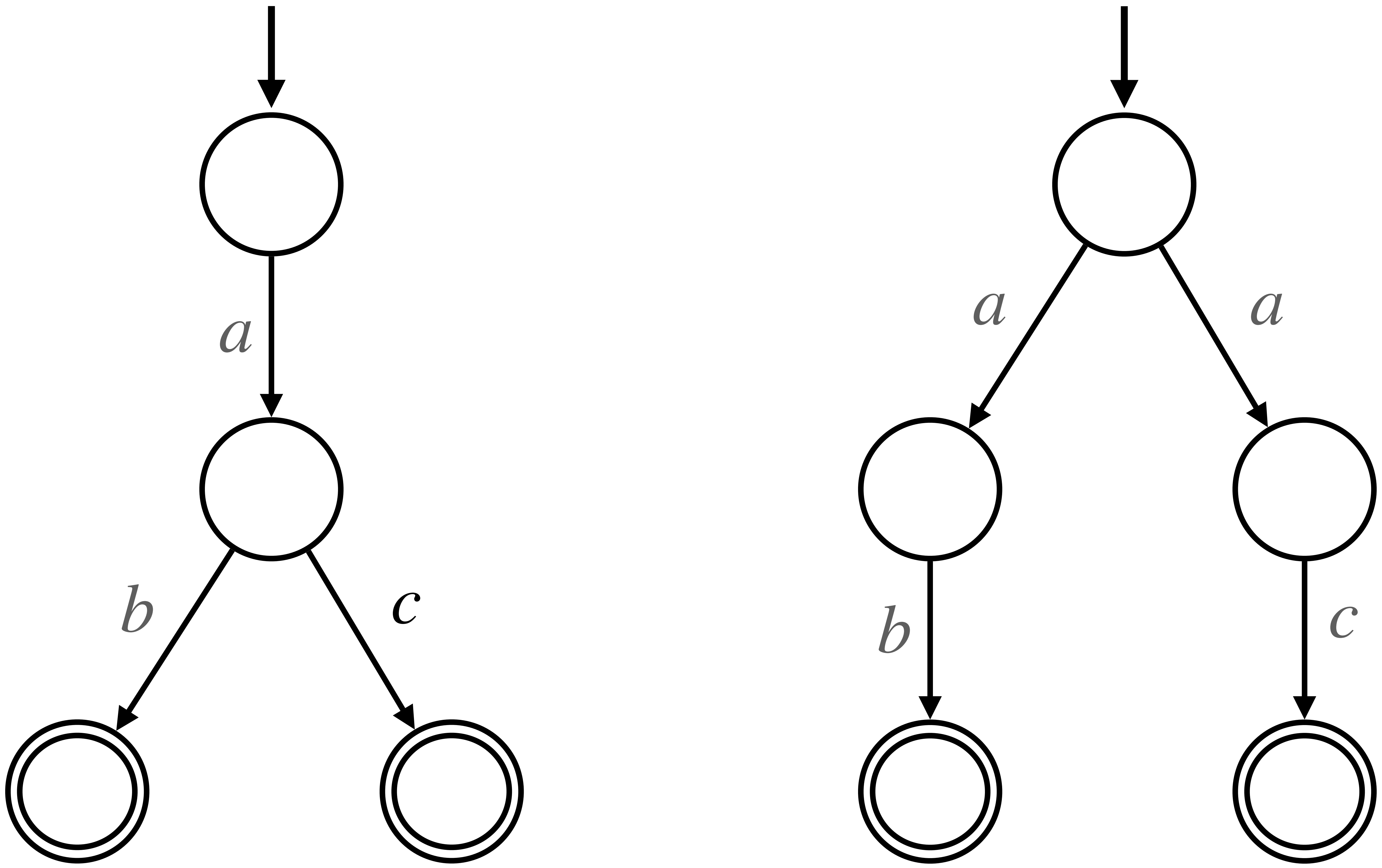
(Milner, 1984)

Bisimilarity for NFAs is Finer than Language Equivalence

Bisimilarity

$\Downarrow$

Language
Equivalence



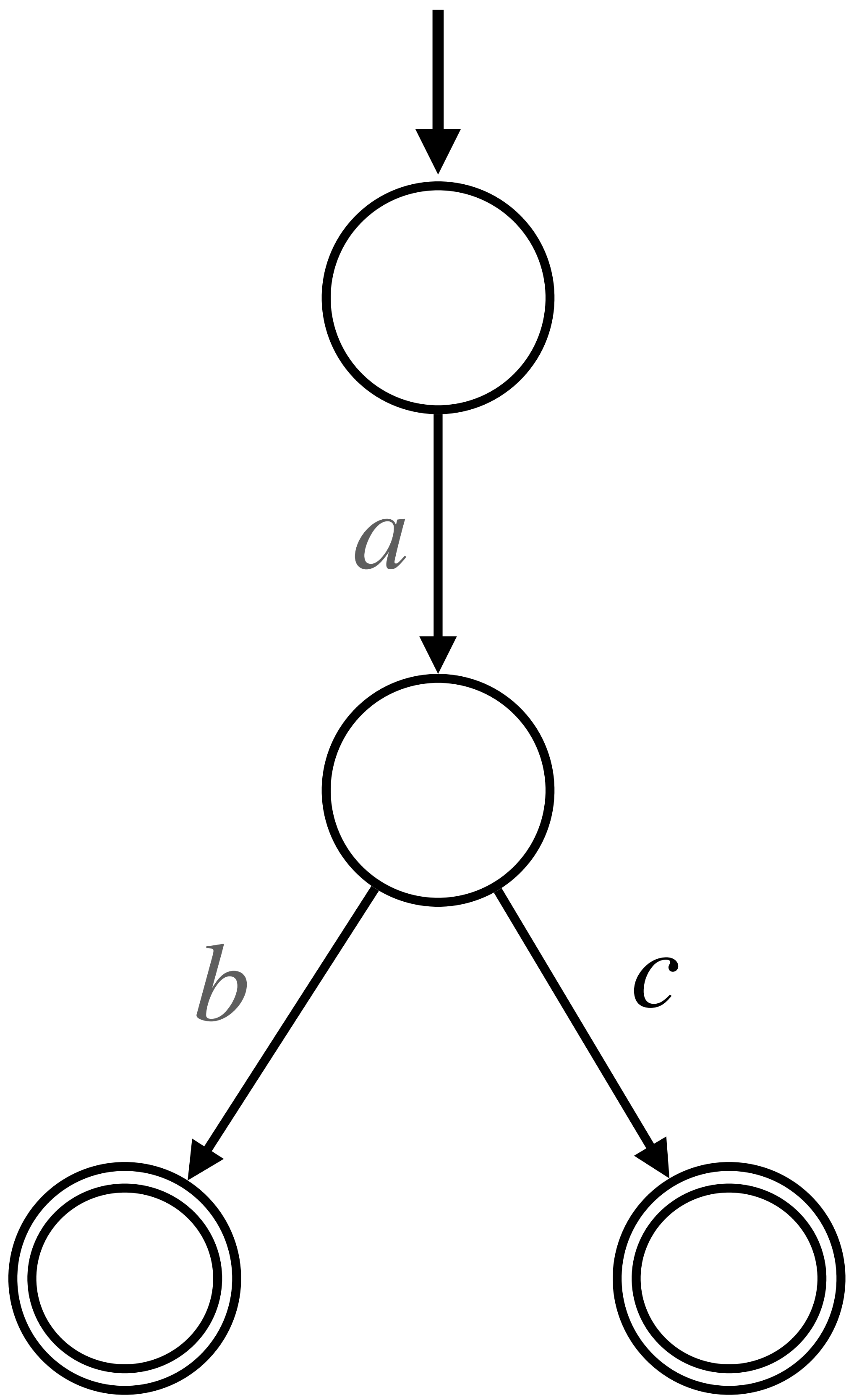
(Milner, 1984)

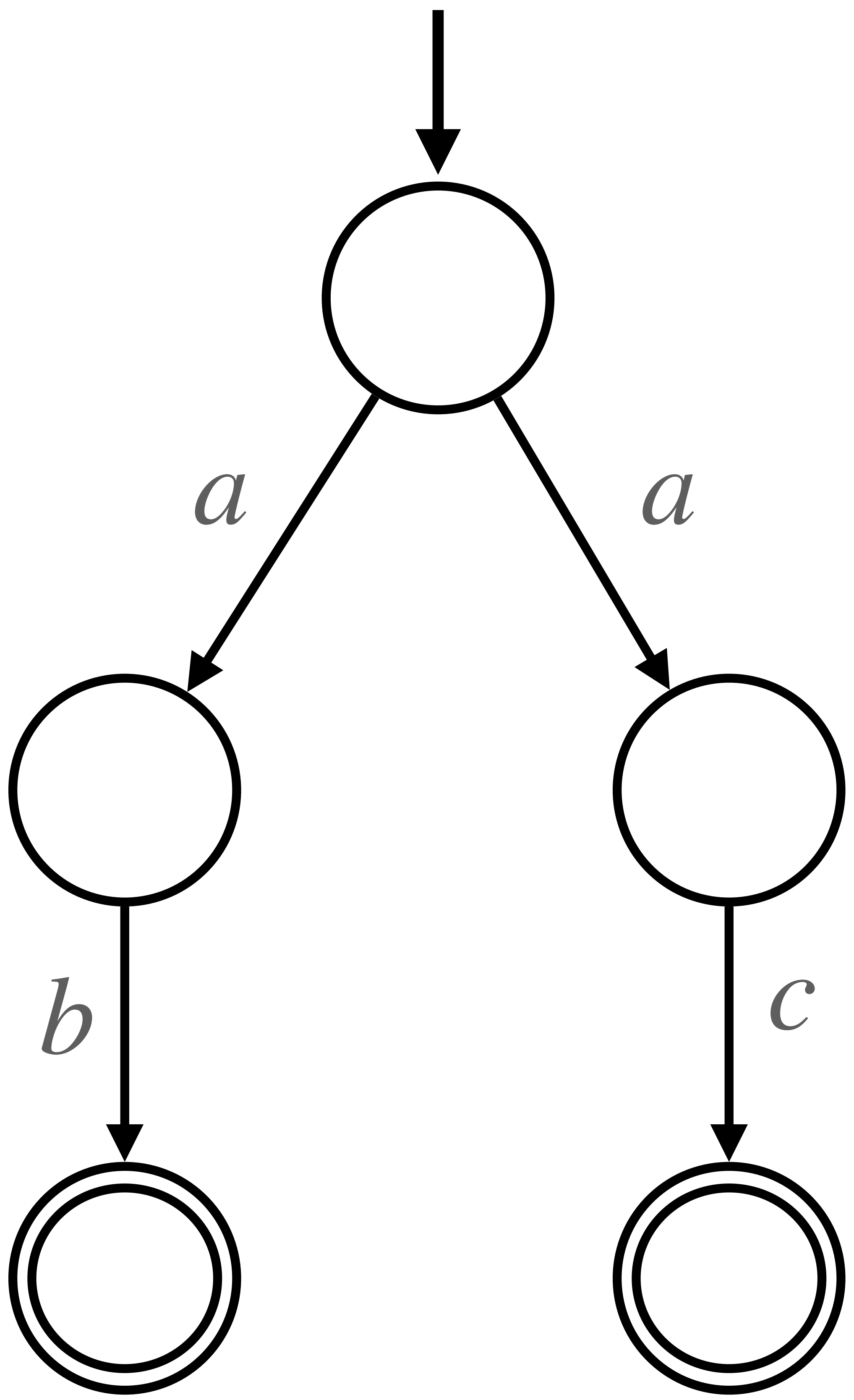
Bisimilarity for NFAs is Finer than Language Equivalence

Bisimilarity

$\Downarrow$

Language Equivalence





Bisimilarity

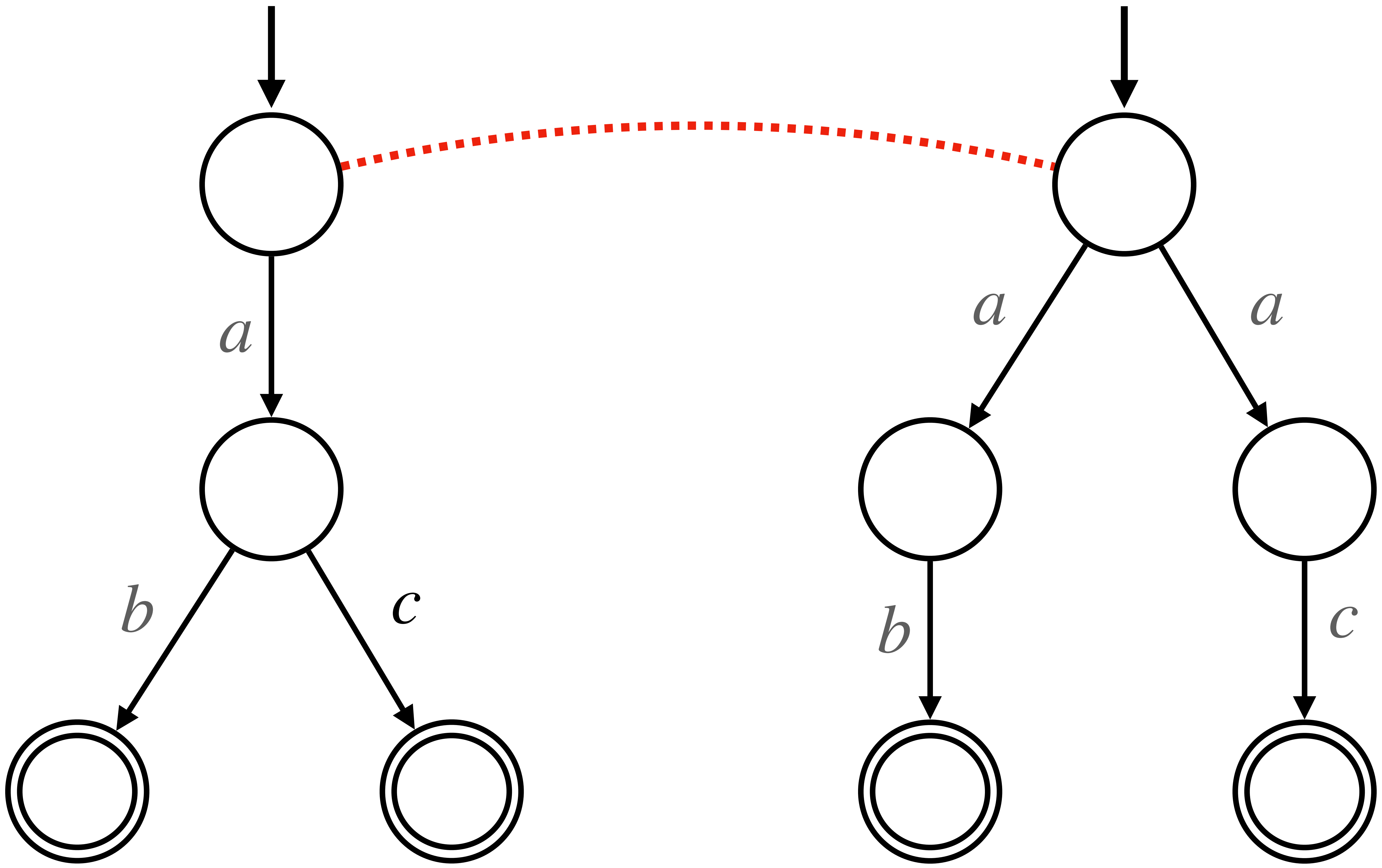
~~$\Uparrow$~~

Language Equivalence

(Milner, 1984)

Bisimilarity for NFAs is Finer than Language Equivalence

Bisimilarity
$\Downarrow$
Language Equivalence

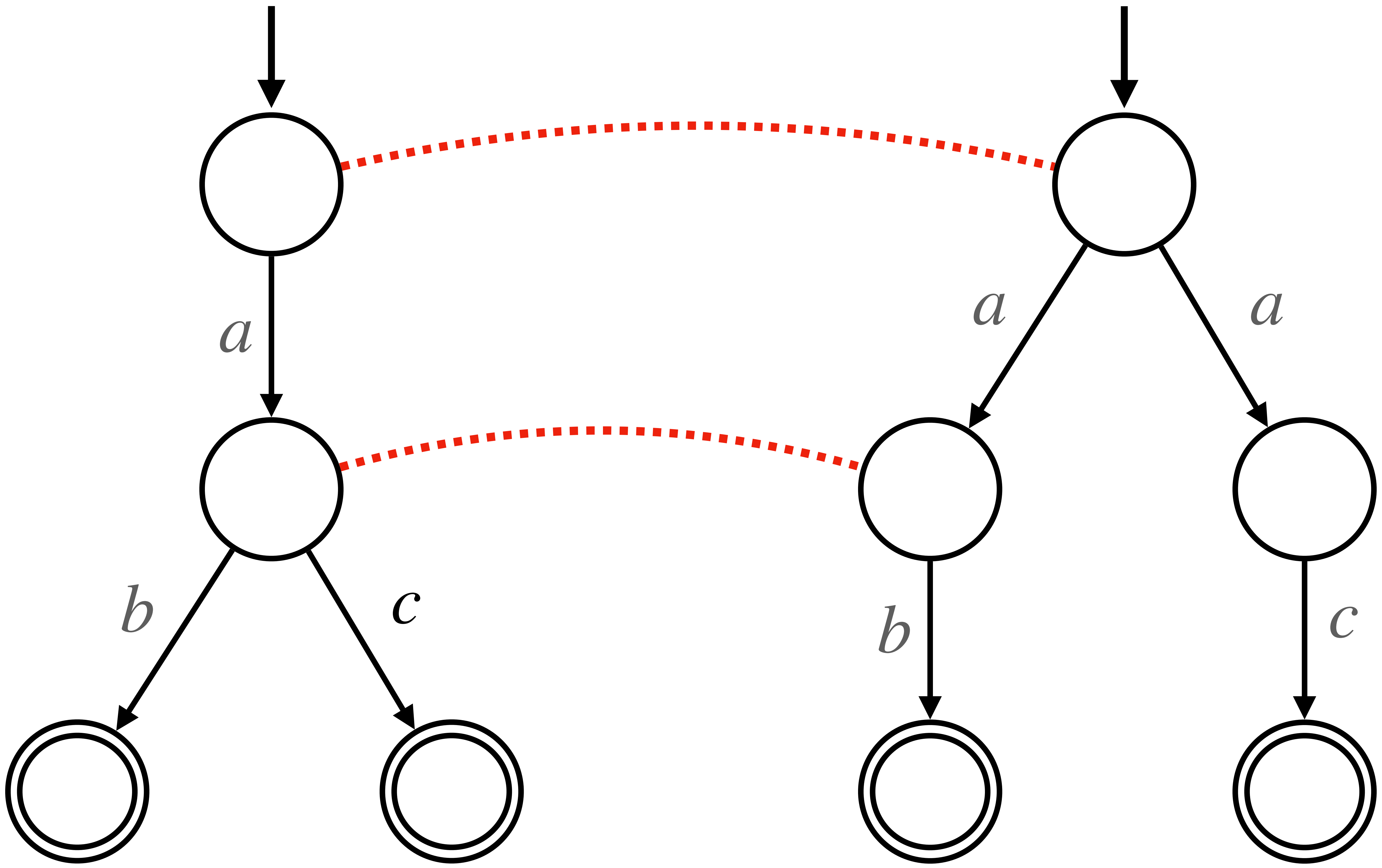


Bisimilarity
$\not\Uparrow$
Language Equivalence

(Milner, 1984)

Bisimilarity for NFAs is Finer than Language Equivalence

Bisimilarity
$\Downarrow$
Language Equivalence

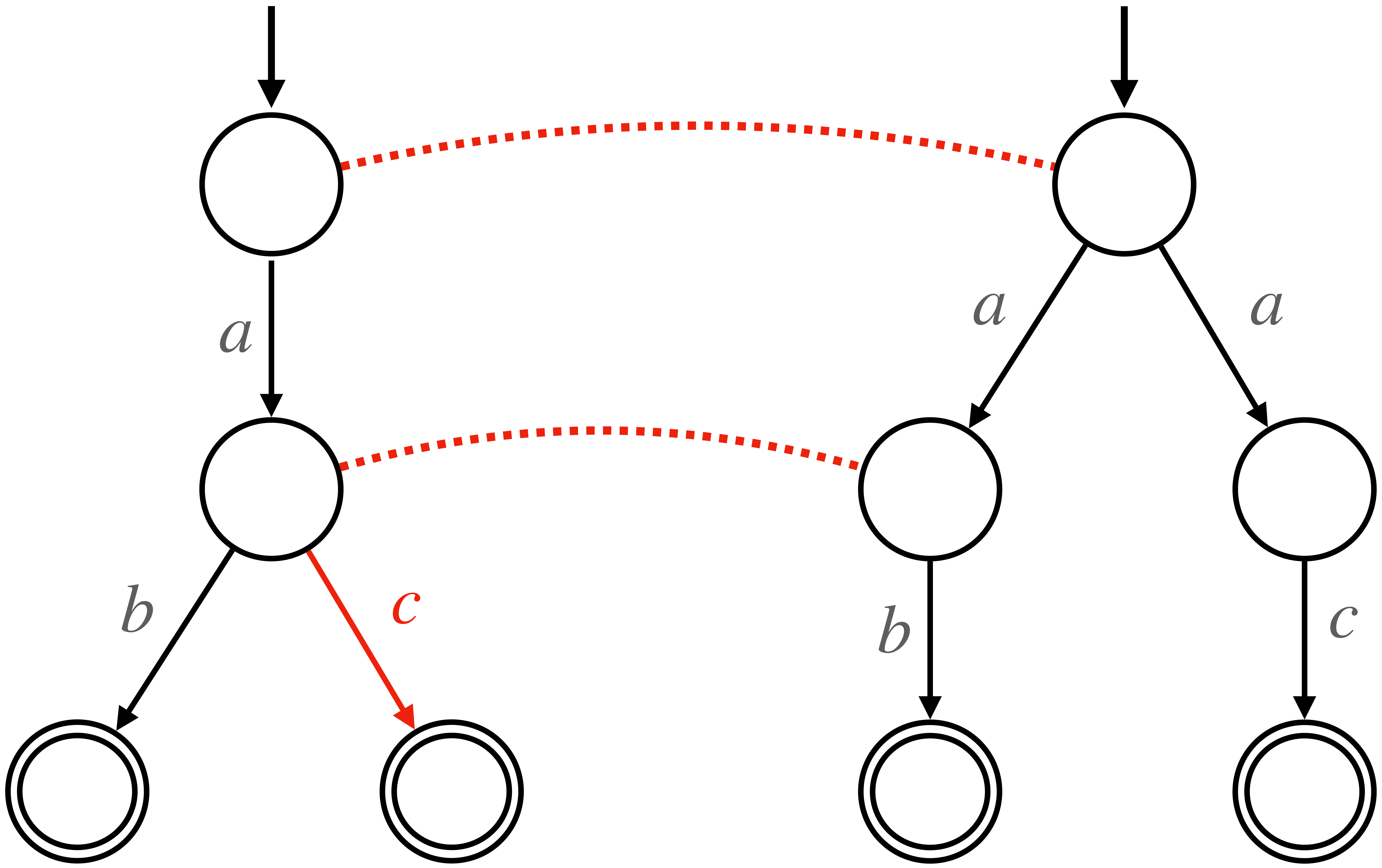


Bisimilarity
$\not\Uparrow$
Language Equivalence

(Milner, 1984)

Bisimilarity for NFAs is Finer than Language Equivalence

Bisimilarity
$\Downarrow$
Language Equivalence

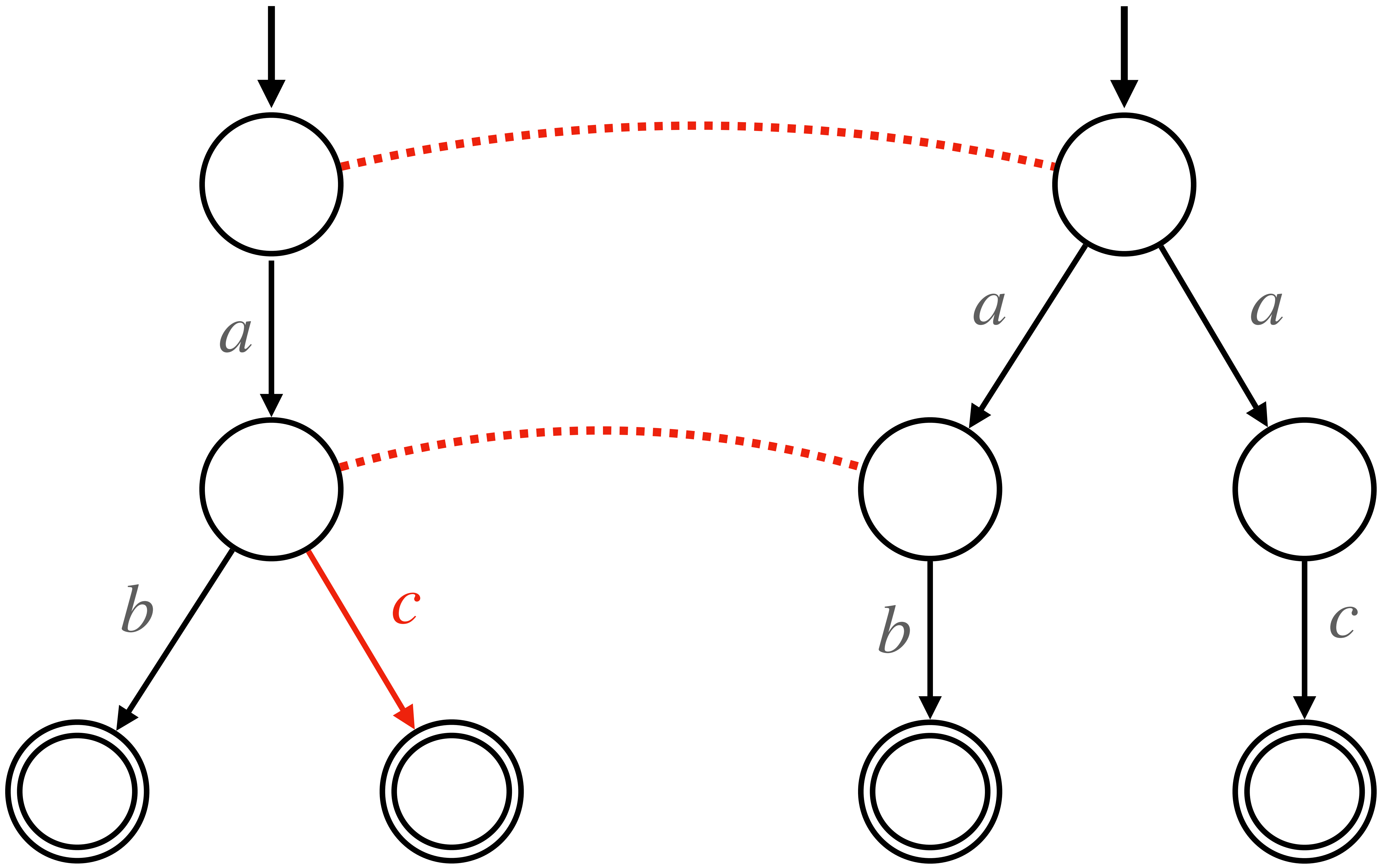


Bisimilarity
$\not\Uparrow$
Language Equivalence

(Milner, 1984)

Bisimilarity for NFAs is Finer than Language Equivalence

Bisimilarity
$\Downarrow$
Language Equivalence



Bisimilarity
$\not\Uparrow$
Language Equivalence

Not all axioms are sound!

(Milner, 1984)

Axiomatizing Bisimilarity of Regular Expressions

Salomaa [9] provides a complete inference system for star expressions under standard interpretation. When we dualise it, by writing $f \circ e$ for $e \circ f$ everywhere in Salomaa's rules (which gives an equipotent system), it has the following rules:

A_1	$e + (f + g) = (e + f) + g$	A_7	$e \circ \phi^* = e$
A_2	$(e \circ f) \circ g = e \circ (f \circ g)$	A_8	$e \circ \phi = \phi$
A_3	$e + f = f + e$	A_9	$e + \phi = e$
A_4	$(e + f) \circ g = e \circ g + f \circ g$	A_{10}	$e^* = \phi^* + e \circ e^*$
A_5	$e \circ (f + g) = e \circ f + e \circ g$	A_{11}	$e^* = (\phi^* + e)^*$
A_6	$e + e = e$		

R_2 *If* f does not possess e.w.p. then

$$\text{from } e = f \circ e + h \text{ infer } e = f^* \circ h.$$

(We have omitted R_1, the substitution rule.)

(Milner, 1984)

Axiomatizing Bisimilarity of Regular Expressions

Salomaa [9] provides a complete inference system for star expressions under standard interpretation. When we dualise it, by writing $f \circ e$ for $e \circ f$ everywhere in Salomaa's rules (which gives an equipotent system), it has the following rules:

A_1 $e + (f + g) = (e + f) + g$

A_2 $(e \circ f) \circ g = e \circ (f \circ g)$

A_3 $e + f = f + e$

A_4 $(e + f) \circ g = e \circ g + f \circ g$

~~A_5 $e \circ (f + g) = e \circ f + e \circ g$~~

A_6 $e + e = e$

A_7 $e \circ \phi^* = e$

~~A_8 $e \circ \phi = \phi$~~

A_9 $e + \phi = e$

A_{10} $e^* = \phi^* + e \circ e^*$

A_{11} $e^* = (\phi^* + e)^*$

R_2 *If* f does not possess e.w.p. then

$$\text{from } e = f \circ e + h \text{ infer } e = f^* \circ h.$$

(We have omitted R_1, the substitution rule.)

(Milner, 1984)

Axiomatizing Bisimilarity of Regular Expressions

Salomaa [9] provides a complete inference system for star e
standard interpretation. When we dualise it, by writing $f \circ e$ for e
Salomaa's rules (which gives an equipotent system), it has the follo

A_1	$e + (f + g) = (e + f) + g$	A_7	$e \circ \phi^* = e$
A_2	$(e \circ f) \circ g = e \circ (f \circ g)$	~~A_8~~	~~$e \circ \phi = \phi$~~
A_3	$e + f = f + e$	A_9	$e + \phi = e$
A_4	$(e + f) \circ g = e \circ g + f \circ g$	A_{10}	$e^* = \phi^* + e \circ e^*$
~~A_5~~	~~$e \circ (f + g) = e \circ f + e \circ g$~~	A_{11}	$e^* = (\phi^* + e)^*$
A_6	$e + e = e$		

R_2 *If* f does not possess e.w.p. then

$$\text{from } e = f \circ e + h \text{ infer } e = f^* \circ h.$$

(We have omitted R_1, the substitution rule.)

By deleting these axioms, Milner obtains a sound axiomatization of *bisimilarity.*

(Milner, 1984)

Axiomatizing Bisimilarity of Regular Expressions

Salomaa [9] provides a complete inference system for star e
standard interpretation. When we dualise it, by writing $f \circ e$ for e
Salomaa's rules (which gives an equipotent system), it has the follo

A_1	$e+(f+g)=(e+f)+g$	A_7	$e \circ \phi^* = e$
A_2	$(e \circ f) \circ g = e \circ (f \circ g)$	~~A_8~~	~~$e \circ \phi = \phi$~~
A_3	$e+f=f+e$	A_9	$e+\phi=e$
A_4	$(e+f) \circ g = e \circ g + f \circ g$	A_{10}	$e^* = \phi^* + e \circ e^*$
~~A_5~~	~~$e \circ (f+g) = e \circ f + e \circ g$~~	A_{11}	$e^* = (\phi^* + e)^*$
A_6	$e+e=e$	A_8'	$\phi \circ e = \phi$

R_2 *If* f does not possess e.w.p. then

$$\text{from } e = f \circ e + h \text{ infer } e = f^* \circ h.$$

(We have omitted R_1, the substitution rule.)

By deleting these axioms, Milner obtains a sound axiomatization of *bisimilarity.*

(Milner, 1984)

Axiomatizing Bisimilarity of Regular Expressions

Salomaa [9] provides a complete inference system for star e… standard interpretation. When we dualise it, by writing $f \circ e$ for e … Salomaa's rules (which gives an equipotent system), it has the follo…

A_1	$e + (f + g) = (e + f) + g$	A_7	$e \circ \phi^* = e$
A_2	$(e \circ f) \circ g = e \circ (f \circ g)$	~~A_8~~	~~$e \circ \phi = \phi$~~
A_3	$e + f = f + e$	A_9	$e + \phi = e$
A_4	$(e + f) \circ g = e \circ g + f \circ g$	A_{10}	$e^* = \phi^* + e \circ e^*$
~~A_5~~	~~$e \circ (f + g) = e \circ f + e \circ g$~~	A_{11}	$e^* = (\phi^* + e)^*$
A_6	$e + e = e$	A'_8	$\phi \circ e = \phi$

R_2 *If* f does not possess e.w.p. then

$$\text{from } e = f \circ e + h \text{ infer } e = f^* \circ h.$$

(We have omitted R_1, the substitution rule.)

By deleting these axioms, Milner obtains a sound axiomatization of *bisimilarity.*

(Milner, 1984)
Is this axiomatization complete?

(Milner, 1984)

Axiomatizing Bisimilarity of Regular Expressions

Salomaa [9] provides a complete inference system for star e
standard interpretation. When we dualise it, by writing $f \circ e$ for e
Salomaa's rules (which gives an equipotent system), it has the follo

A_1	$e + (f + g) = (e + f) + g$	A_7	$e \circ \phi^* = e$
A_2	$(e \circ f) \circ g = e \circ (f \circ g)$	~~A_8~~	~~$e \circ \phi = \phi$~~
A_3	$e + f = f + e$	A_9	$e + \phi = e$
A_4	$(e + f) \circ g = e \circ g + f \circ g$	A_{10}	$e^* = \phi^* + e \circ e^*$
~~A_5~~	~~$e \circ (f + g) = e \circ f + e \circ g$~~	A_{11}	$e^* = (\phi^* + e)^*$
A_6	$e + e = e$	A_8'	$\phi \circ e = \phi$

R_2 *If* f does not possess e.w.p. then

from $e = f \circ e + h$ infer $e = f^* \circ h$.

(We have omitted R_1, the substitution rule.)

(Milner, 1984)

By deleting these axioms, Milner obtains a sound axiomatization of *bisimilarity.*

(Milner, 1984)
Is this axiomatization complete?

(Grabmayer, 2022)
Yes!

Axiomatizing Bisimilarity of Regular Expressions

An equivalent rendering of Milner's axioms for regular expressions modulo bisimilarity:

$$e = e + 0$$
$$e = e + e$$
$$f + e = e + f$$
$$e + (f + g) = (e + f) + g$$

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e + f)g = eg + fg$$

$$e^* = (1 + e)^*$$

$$e^* = ee^* + 1$$

$$\frac{g = eg + f \quad e \text{ guarded}}{g = e^* f}$$

Axiomatizing Bisimilarity of Regular Expressions

An equivalent rendering of Milner's axioms for regular expressions modulo bisimilarity:

$$e = e + 0$$
$$e = e + e$$
$$f + e = e + f$$
$$e + (f + g) = (e + f) + g$$

Equational Branching Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e + f)g = eg + fg$$

$$e^* = (1 + e)^*$$

$$e^* = ee^* + 1$$

$$\frac{g = eg + f \quad e \text{ guarded}}{g = e^* f}$$

Axiomatizing Bisimilarity of Regular Expressions

An equivalent rendering of Milner's axioms for regular expressions modulo bisimilarity:

$$e = e + 0$$
$$e = e + e$$
$$f + e = e + f$$
$$e + (f + g) = (e + f) + g$$

Equational Branching Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e + f)g = eg + fg$$

Sequencing Axioms

$$e^* = (1 + e)^*$$

$$e^* = ee^* + 1$$

$$\frac{g = eg + f \quad e \text{ guarded}}{g = e^* f}$$

Axiomatizing Bisimilarity of Regular Expressions

An equivalent rendering of Milner's axioms for regular expressions modulo bisimilarity:

$$e = e + 0$$
$$e = e + e$$
$$f + e = e + f$$
$$e + (f + g) = (e + f) + g$$

Equational Branching Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e + f)g = eg + fg$$

Sequencing Axioms

$$e^* = (1 + e)^*$$

$$e^* = ee^* + 1$$
$$\frac{g = eg + f \quad e \text{ guarded}}{g = e^* f}$$

Unique Guarded Fixed-point Axioms

Axiomatizing Bisimilarity of Regular Expressions

An equivalent rendering of Milner's axioms for regular expressions modulo bisimilarity:

$$e = e + 0$$
$$e = e + e$$
$$f + e = e + f$$
$$e + (f + g) = (e + f) + g$$

Equational Branching Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e + f)g = eg + fg$$

Sequencing Axioms

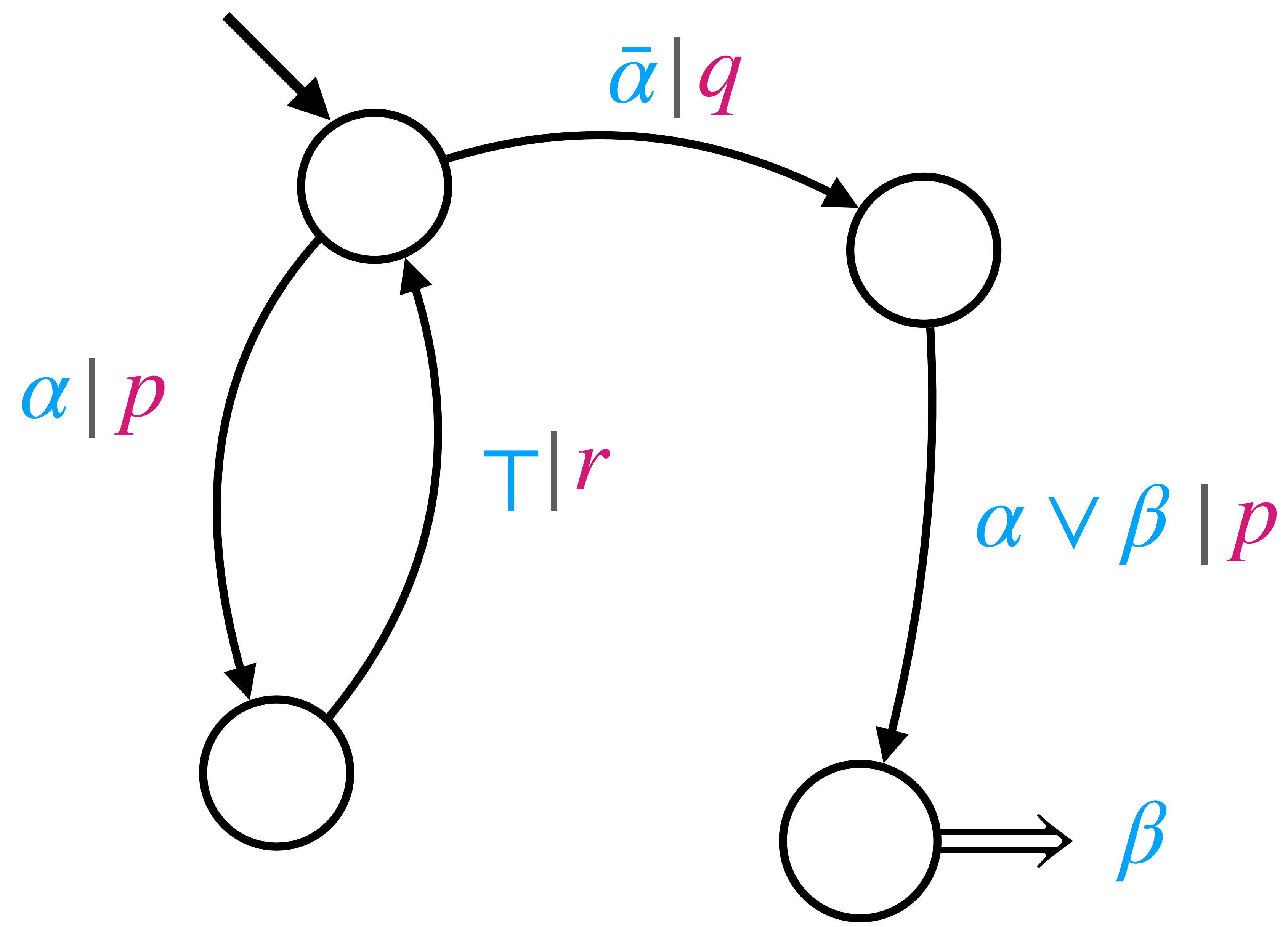
Unguarded Fixed-point Axiom

$$e^* = (1 + e)^*$$

$$e^* = ee^* + 1$$
$$\frac{g = eg + f \quad e \text{ guarded}}{g = e^* f}$$

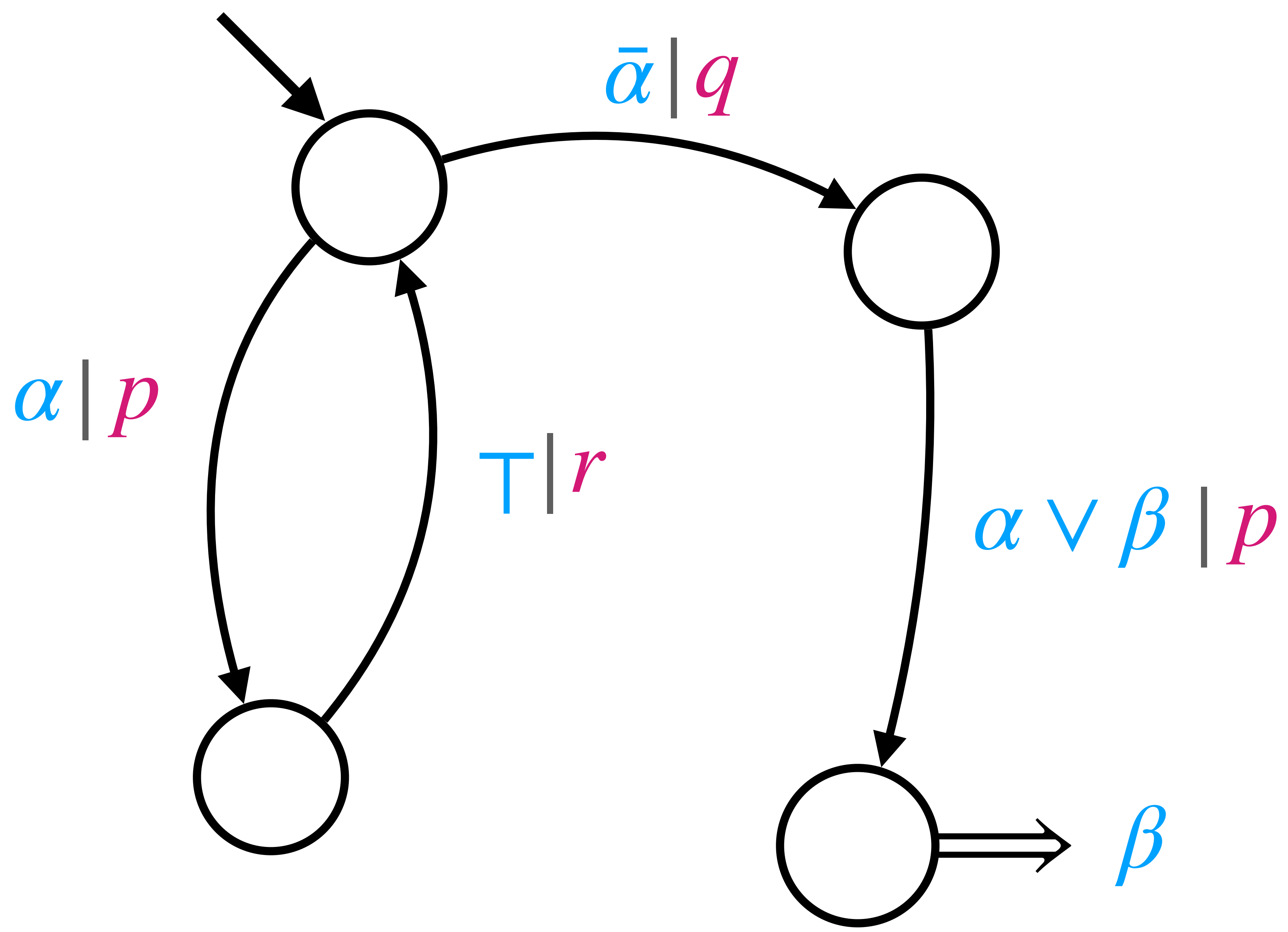
Unique Guarded Fixed-point Axioms

A Similar Situation: Guarded Kleene Algebra with Tests



A Similar Situation: Guarded Kleene Algebra with Tests

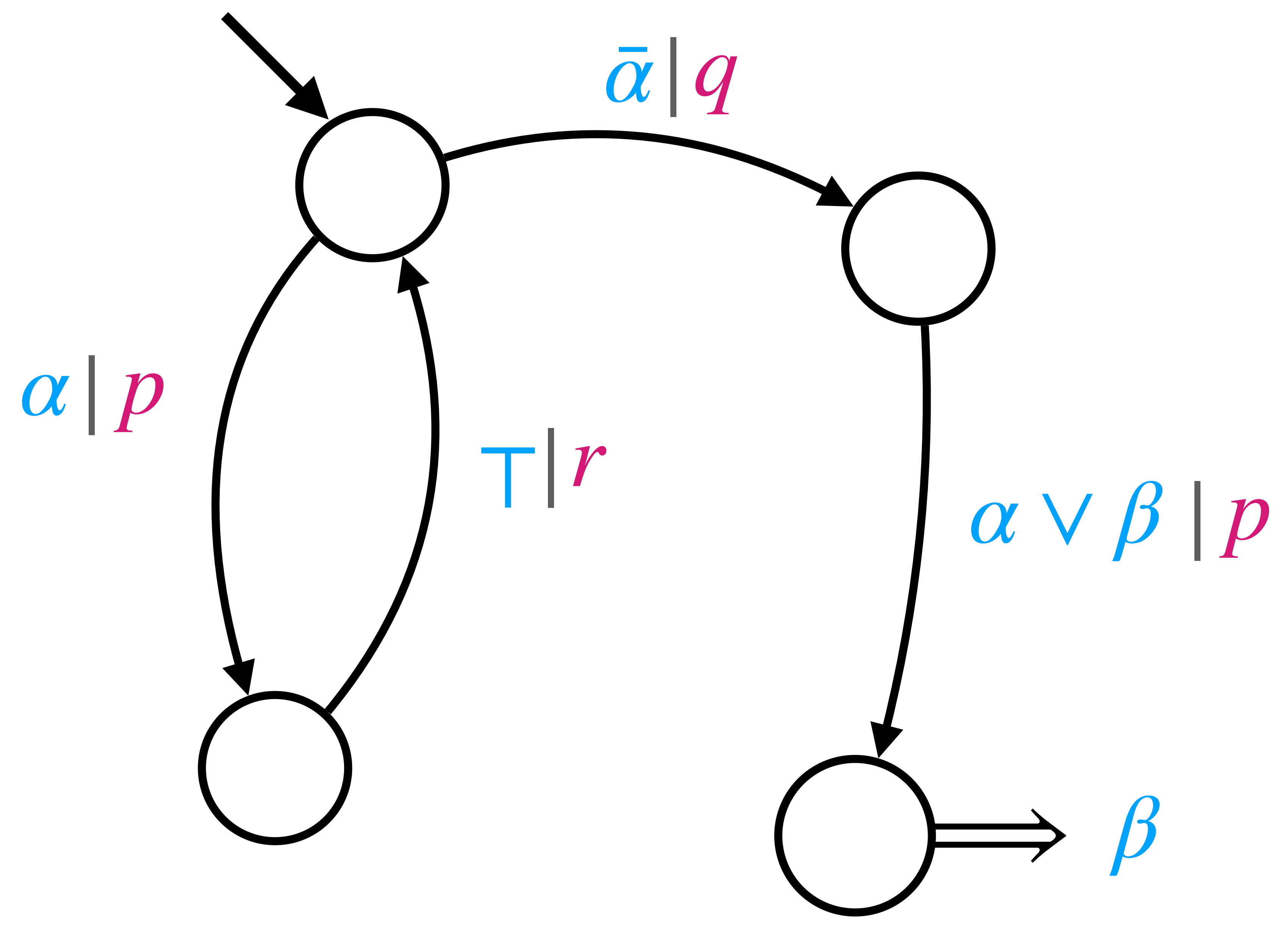
- An algebra of propositional WHILE programs



A Similar Situation: Guarded Kleene Algebra with Tests

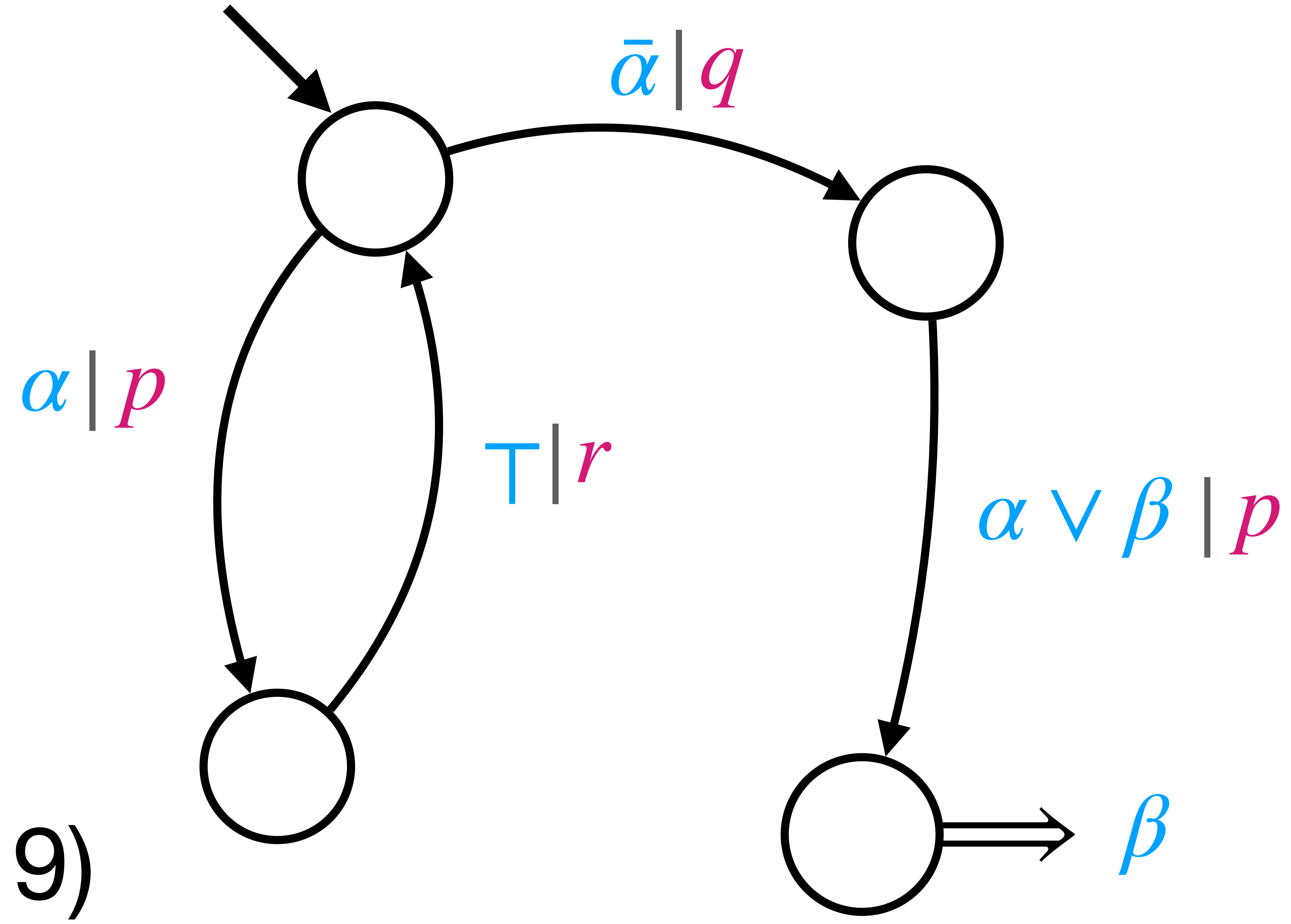
- An algebra of propositional WHILE programs

- (Kozen, Tseng, 2008) Syntax and language semantics from Kleene Algebra with Tests

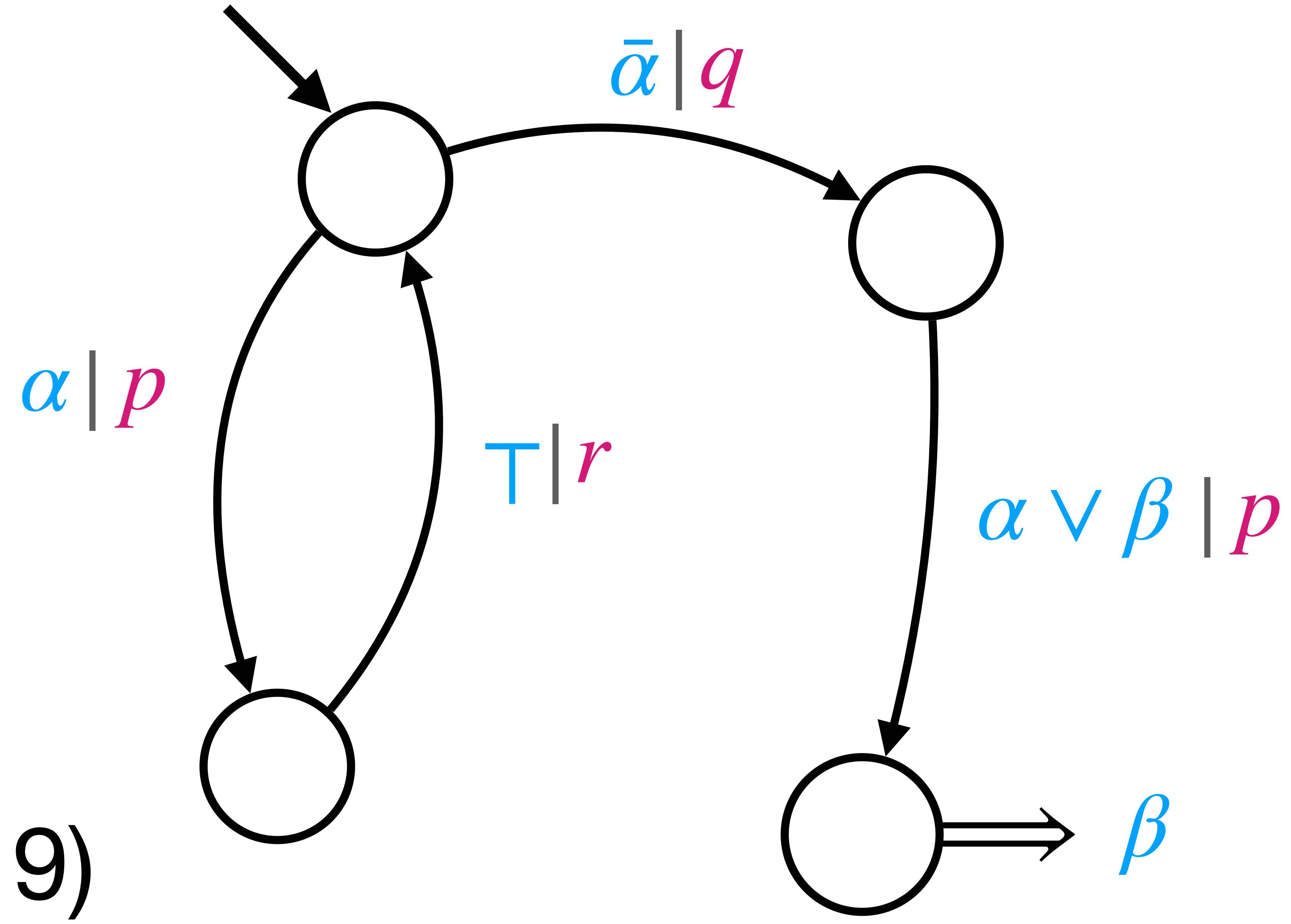


A Similar Situation: Guarded Kleene Algebra with Tests

- An algebra of propositional WHILE programs
- (Kozen, Tseng, 2008) Syntax and language semantics from Kleene Algebra with Tests
- (Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)
  - Operational semantics, almost linear decision procedure
  - Propose a Salomaa-like axiomatization of language equivalence



A Similar Situation: Guarded Kleene Algebra with Tests



- An algebra of propositional WHILE programs
- (Kozen, Tseng, 2008) Syntax and language semantics from Kleene Algebra with Tests
- (Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)
  - Operational semantics, almost linear decision procedure
  - Propose a Salomaa-like axiomatization of language equivalence
- (S., Kappé, Kozen, Silva, 2021)
  - Infinite tree semantics = bisimilarity
  - Propose a Salomaa-like axiomatization of bisimilarity

Guarded Kleene Algebra with Tests

$$\mathsf{BExp} \ni b, c ::= 0 \mid 1 \mid t \in T \mid b \vee c \mid b \wedge c \mid \bar{b}$$

Guarded Kleene Algebra with Tests

Generates an atomic Boolean algebra with atoms $At = 2^T$.
$\mathrm{BExp}/=_{\mathrm{BA}} \cong \mathscr{P}(2^T)$

$$\mathrm{BExp} \ni b, c ::= 0 \mid 1 \mid t \in T \mid b \vee c \mid b \wedge c \mid \bar{b}$$

Guarded Kleene Algebra with Tests

Generates an atomic Boolean algebra with atoms $At = 2^T$.
$\mathsf{BExp}/ =_{\mathsf{BA}} \cong \mathscr{P}(2^T)$

$$\mathsf{BExp} \ni b, c ::= 0 \mid 1 \mid t \in T \mid b \vee c \mid b \wedge c \mid \bar{b}$$

$$\mathsf{GExp} \ni e, f ::= b \in \mathsf{BExp} \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

(Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)

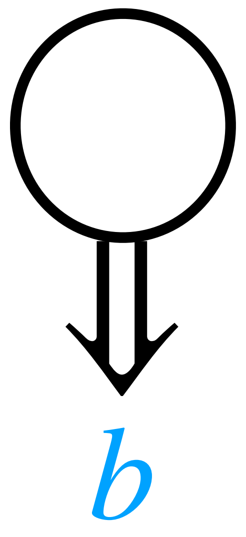
Guarded Kleene Algebra with Tests

Generates an atomic Boolean algebra with atoms $At = 2^T$.
$\mathsf{BExp}/=_{\mathsf{BA}} \cong \mathscr{P}(2^T)$

$$\mathsf{BExp} \ni b, c ::= 0 \mid 1 \mid t \in T \mid b \vee c \mid b \wedge c \mid \bar{b}$$

$$\mathsf{GExp} \ni e, f ::= b \in \mathsf{BExp} \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

assert b



(Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)

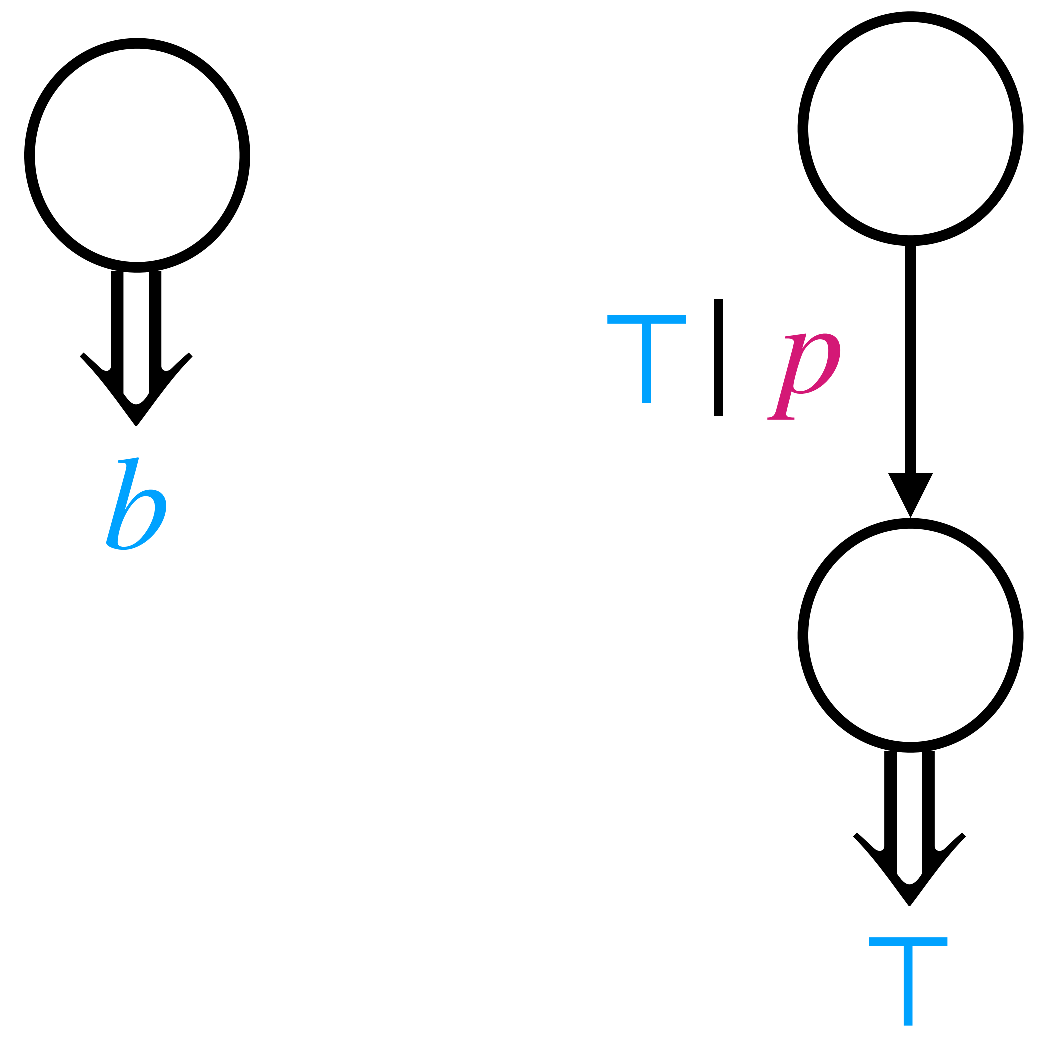
Guarded Kleene Algebra with Tests

Generates an atomic Boolean algebra with atoms $At = 2^T$. $\mathsf{BExp}/ =_{\mathsf{BA}} \cong \mathscr{P}(2^T)$

$$\mathsf{BExp} \ni b, c ::= 0 \mid 1 \mid t \in T \mid b \vee c \mid b \wedge c \mid \bar{b}$$

$$\mathsf{GExp} \ni e, f ::= b \in \mathsf{BExp} \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

assert b do p



(Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)

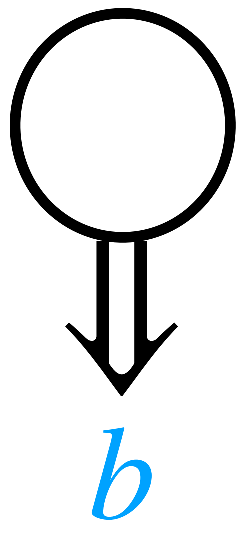
Guarded Kleene Algebra with Tests

Generates an atomic Boolean algebra with atoms $At = 2^T$.
$\mathrm{BExp}/=_{\mathrm{BA}} \cong \mathscr{P}(2^T)$

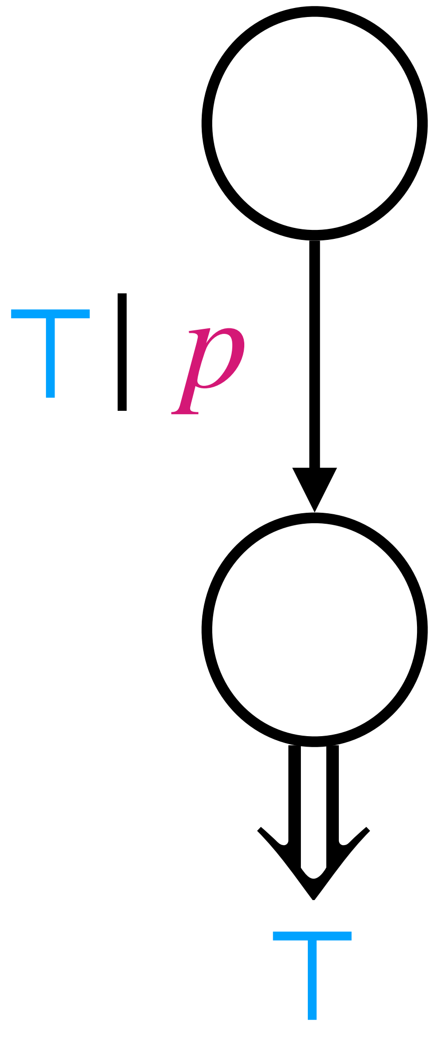
$$\mathrm{BExp} \ni b, c ::= 0 \mid 1 \mid t \in T \mid b \vee c \mid b \wedge c \mid \bar{b}$$

$$\mathrm{GExp} \ni e, f ::= b \in \mathrm{BExp} \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

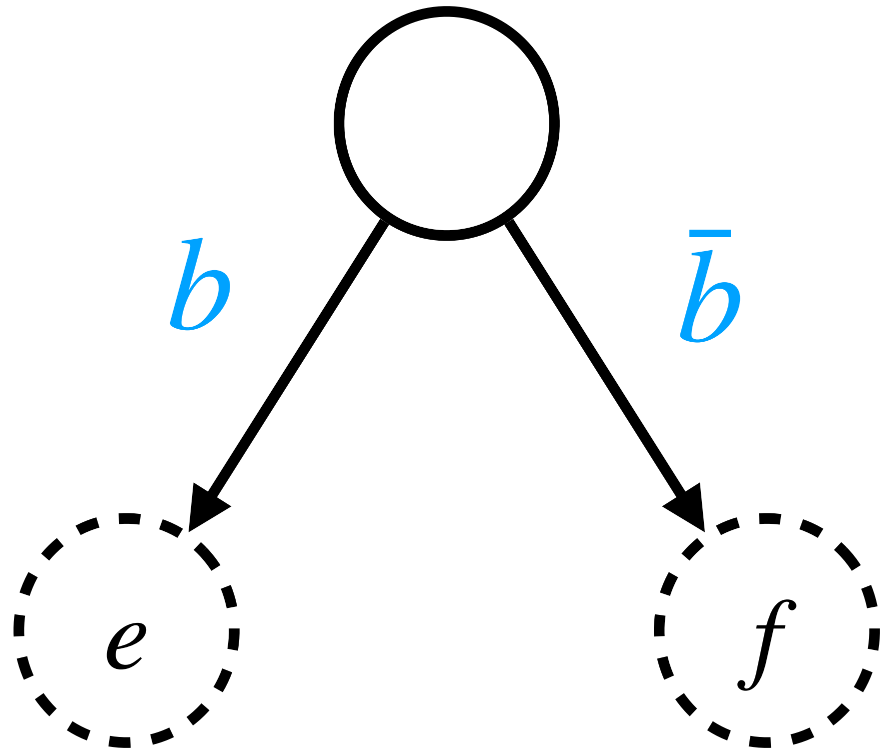
assert b



do p



if b then e else f



(Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)

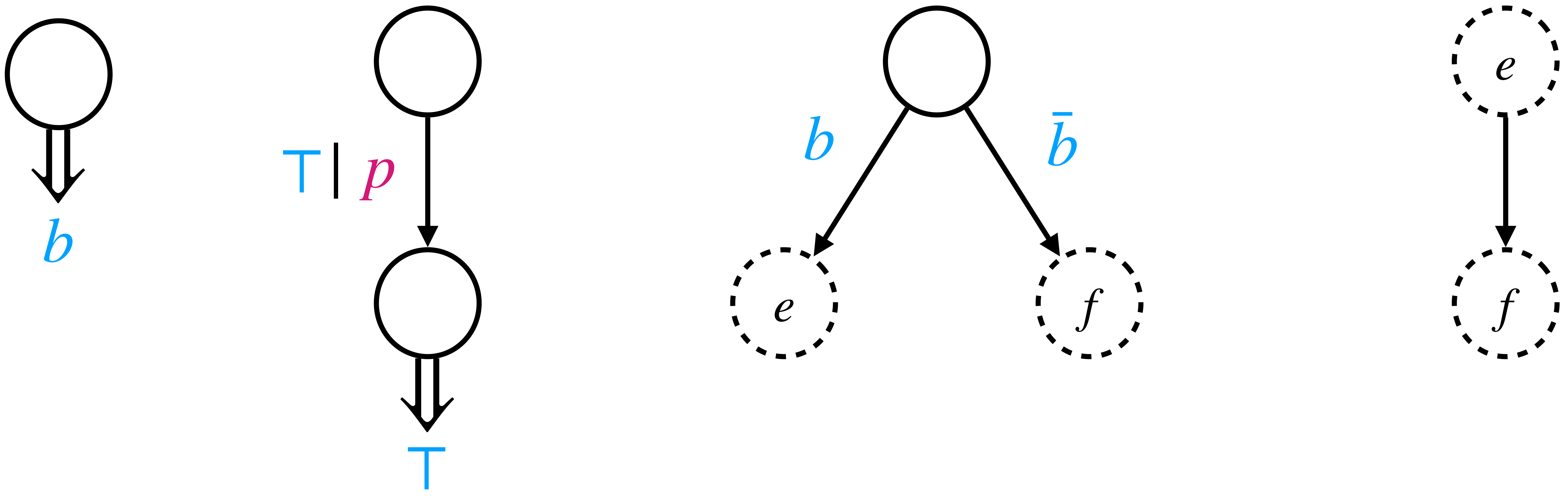
Guarded Kleene Algebra with Tests

Generates an atomic Boolean algebra with atoms $At = 2^T$.
$\mathrm{BExp}/ =_{\mathrm{BA}} \cong \mathscr{P}(2^T)$

$$\mathrm{BExp} \ni b, c ::= 0 \mid 1 \mid t \in T \mid b \vee c \mid b \wedge c \mid \bar{b}$$

$$\mathrm{GExp} \ni e, f ::= b \in \mathrm{BExp} \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

assert b | do p | if b then e else f | ef



(Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)

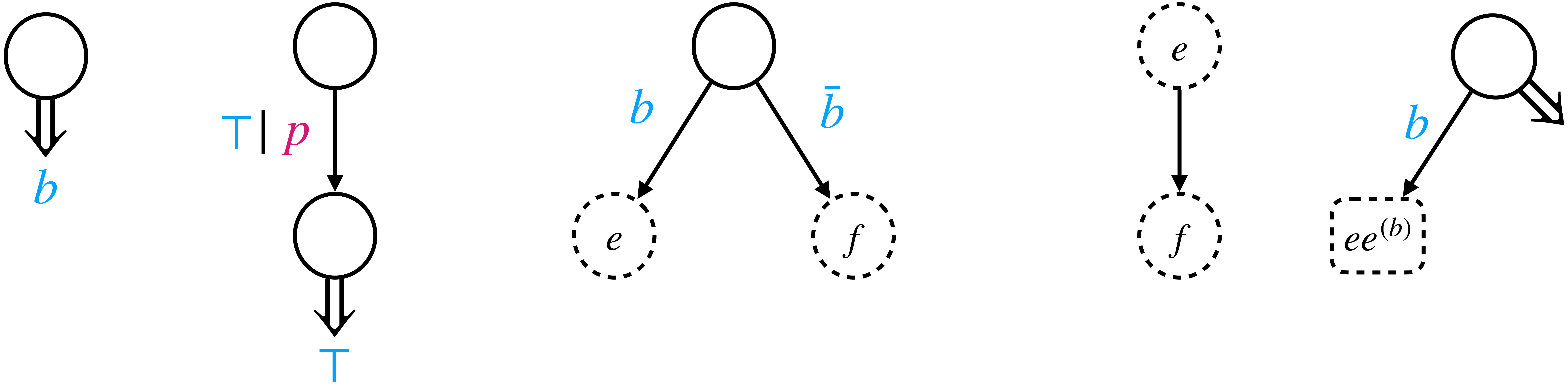
Guarded Kleene Algebra with Tests

Generates an atomic Boolean algebra with atoms $At = 2^T$.
$\mathrm{BExp}/ =_{\mathrm{BA}} \cong \mathscr{P}(2^T)$

$$\mathrm{BExp} \ni b, c ::= 0 \mid 1 \mid t \in T \mid b \vee c \mid b \wedge c \mid \bar{b}$$

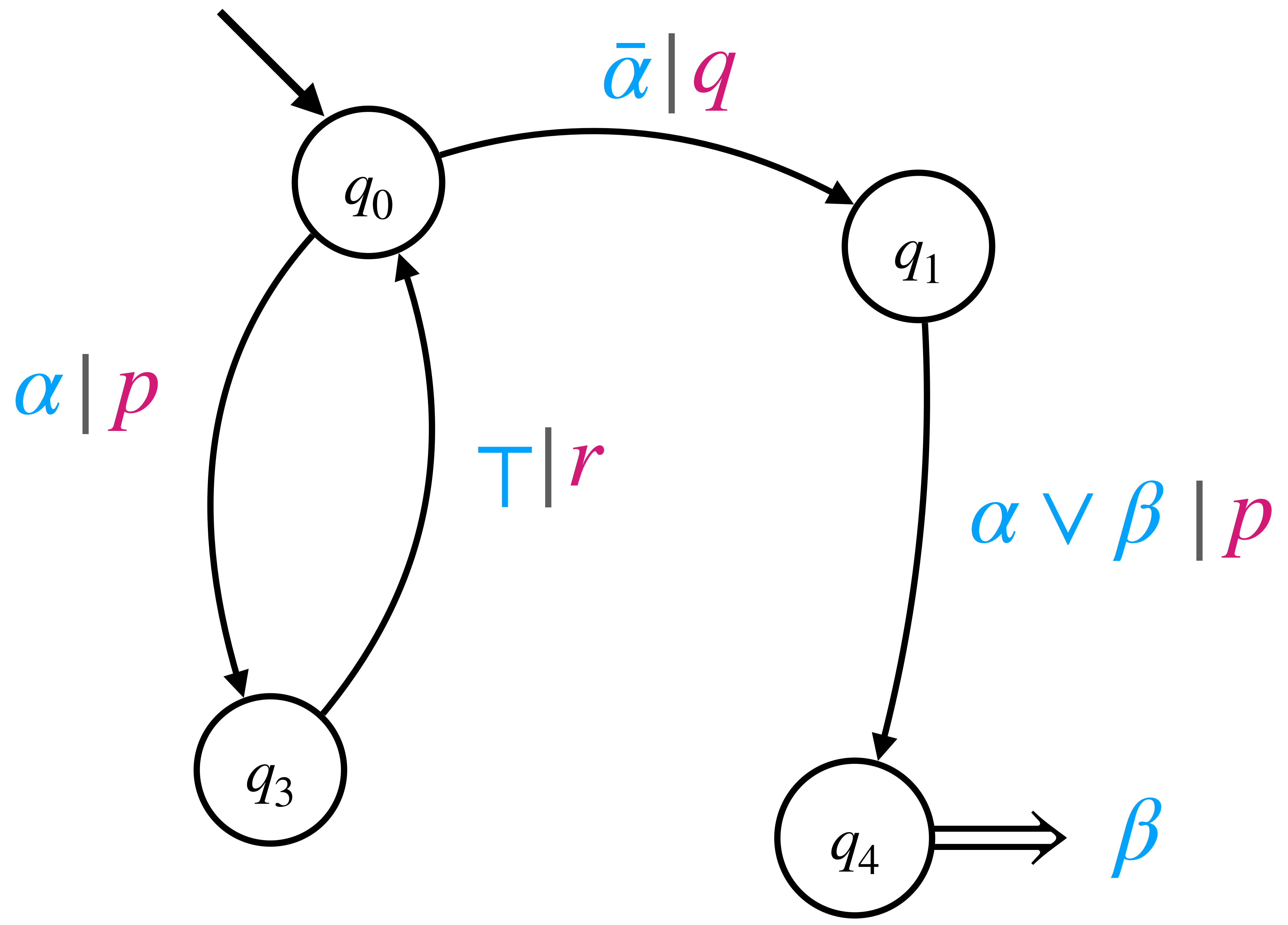
$$\mathrm{GExp} \ni e, f ::= b \in \mathrm{BExp} \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

assert b | do p | if b then e else f | ef | while b do e



(Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)

Example of a GKAT Automaton



$$X \longrightarrow (\{\bot, \top\} + \Sigma \times X)^{At}$$

$$(pr)^{(\alpha)} q (p\beta +_{\alpha \vee \beta} 0)$$

```
while α do
    p
    r
q
if α ∨ β then
    p
    assert β
else
    assert False
```

(Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)

Axiomatizing GKAT Programs up to Language Equivalence

(Smolka et al., 2019) Proposed the following axiomatization of GKAT

Guarded Union Axioms

U1.	$e +_b e \equiv e$	(idempotence)
U2.	$e +_b f \equiv f +_{\overline{b}} e$	(skew commut.)
U3.	$(e +_b f) +_c g \equiv e +_{bc} (f +_c g)$	(skew assoc.)
U4.	$e +_b f \equiv be +_b f$	(guardedness)
U5.	$eg +_b fg \equiv (e +_b f) \cdot g$	(right distrib.)

Sequence Axioms (inherited from KA)

S1.	$(e \cdot f) \cdot g \equiv e \cdot (f \cdot g)$	(associativity)
S2.	$0 \cdot e \equiv 0$	(absorbing left)
S3.	$e \cdot 0 \equiv 0$	(absorbing right)
S4.	$1 \cdot e \equiv e$	(neutral left)
S5.	$e \cdot 1 \equiv e$	(neutral right)

Guarded Loop Axioms

W1.	$e^{(b)} \equiv ee^{(b)} +_b 1$	(unrolling)
W2.	$(e +_c 1)^{(b)} \equiv (ce)^{(b)}$	(tightening)
W3.	$\dfrac{g \equiv eg +_b f}{g \equiv e^{(b)} f}$ if $E(e) \equiv 0$	(fixpoint)

(Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)

Axiomatizing GKAT Programs up to Language Equivalence

(Smolka et al., 2019) Proposed the following axiomatization of GKAT

Guarded Union Axioms

U1.	$e +_b e \equiv e$	(idempotence)
U2.	$e +_b f \equiv f +_{\bar{b}} e$	(skew commut.)
U3.	$(e +_b f) +_c g \equiv e +_{bc} (f +_c g)$	(skew assoc.)
U4.	$e +_b f \equiv be +_b f$	(guardedness)
U5.	$eg +_b fg \equiv (e +_b f) \cdot g$	(right distrib.)

Sequence Axioms (inherited from KA)

S1.	$(e \cdot f) \cdot g \equiv e \cdot (f \cdot g)$	(associativity)
S2.	$0 \cdot e \equiv 0$	(absorbing left)
S3.	$e \cdot 0 \equiv 0$	(absorbing right)
S4.	$1 \cdot e \equiv e$	(neutral left)
S5.	$e \cdot 1 \equiv e$	(neutral right)

Guarded Loop Axioms

W1.	$e^{(b)} \equiv ee^{(b)} +_b 1$	(unrolling)
W2.	$(e +_c 1)^{(b)} \equiv (ce)^{(b)}$	(tightening)
W3.	$\dfrac{g \equiv eg +_b f}{g \equiv e^{(b)} f}$ if $E(e) \equiv 0$	(fixpoint)

Open Problem: Are these axioms complete for language equivalence?

(Smolka, Foster, Hsu, Kappé, Kozen, Silva, 2019)

Axiomatizing GKAT Programs up to Bisimilarity

(S., Kappé, Kozen, Silva, 2021) Proposed the following axiomatization of GKAT/bisimilarity

Guarded Union Axioms

U1.	$e +_b e \equiv e$	(idempotence)
U2.	$e +_b f \equiv f +_{\overline{b}} e$	(skew commut.)
U3.	$(e +_b f) +_c g \equiv e +_{bc} (f +_c g)$	(skew assoc.)
U4.	$e +_b f \equiv be +_b f$	(guardedness)
U5.	$eg +_b fg \equiv (e +_b f) \cdot g$	(right distrib.)

Sequence Axioms (inherited from KA)

S1.	$(e \cdot f) \cdot g \equiv e \cdot (f \cdot g)$	(associativity)
S2.	$0 \cdot e \equiv 0$	(absorbing left)
~~S3.~~	~~$e \cdot 0 \equiv 0$~~	~~(absorbing right)~~
S4.	$1 \cdot e \equiv e$	(neutral left)
S5.	$e \cdot 1 \equiv e$	(neutral right)

Guarded Loop Axioms

W1.	$e^{(b)} \equiv ee^{(b)} +_b 1$	(unrolling)
W2.	$(e +_c 1)^{(b)} \equiv (ce)^{(b)}$	(tightening)
W3.	$\dfrac{g \equiv eg +_b f}{g \equiv e^{(b)} f}$ if $E(e) \equiv 0$	(fixpoint)

Axiomatizing GKAT Programs up to Bisimilarity

(S., Kappé, Kozen, Silva, 2021) Proposed the following axiomatization of GKAT/bisimilarity

Guarded Union Axioms

U1.	$e +_b e \equiv e$	(idempotence)
U2.	$e +_b f \equiv f +_{\bar{b}} e$	(skew commut.)
U3.	$(e +_b f) +_c g \equiv e +_{bc} (f +_c g)$	(skew assoc.)
U4.	$e +_b f \equiv be +_b f$	(guardedness)
U5.	$eg +_b fg \equiv (e +_b f) \cdot g$	(right distrib.)

Sequence Axioms (inherited from KA)

S1.	$(e \cdot f) \cdot g \equiv e \cdot (f \cdot g)$	(associativity)
S2.	$0 \cdot e \equiv 0$	(absorbing left)
~~S3.~~	~~$e \cdot 0 \equiv 0$~~	~~(absorbing right)~~
S4.	$1 \cdot e \equiv e$	(neutral left)
S5.	$e \cdot 1 \equiv e$	(neutral right)

Guarded Loop Axioms

W1. $e^{(b)} \equiv ee^{(b)} +_b 1$ (unrolling)

W2. $(e +_c 1)^{(b)} \equiv (ce)^{(b)}$ (tightening)

W3. $$\frac{g \equiv eg +_b f}{g \equiv e^{(b)} f} \text{ if } E(e) \equiv 0$$ (fixpoint)

Open Problem: Are these axioms complete for bisimilarity?

Axiomatizing GKAT Programs up to Bisimilarity

(S., Kappé, Kozen, Silva, 2021) Proposed the following axiomatization of GKAT/bisimilarity

Guarded Union Axioms

U1.	$e +_b e \equiv e$	(idempotence)
U2.	$e +_b f \equiv f +_{\bar{b}} e$	(skew commut.)
U3.	$(e +_b f) +_c g \equiv e +_{bc} (f +_c g)$	(skew assoc.)
U4.	$e +_b f \equiv be +_b f$	(guardedness)
U5.	$eg +_b fg \equiv (e +_b f) \cdot g$	(right distrib.)

Sequence Axioms (inherited from KA)

S1.	$(e \cdot f) \cdot g \equiv e \cdot (f \cdot g)$	(associativity)
S2.	$0 \cdot e \equiv 0$	(absorbing left)
~~S3.~~	~~$e \cdot 0 \equiv 0$~~	~~(absorbing right)~~
S4.	$1 \cdot e \equiv e$	(neutral left)
S5.	$e \cdot 1 \equiv e$	(neutral right)

Guarded Loop Axioms

W1.	$e^{(b)} \equiv ee^{(b)} +_b 1$	(unrolling)
W2.	$(e +_c 1)^{(b)} \equiv (ce)^{(b)}$	(tightening)
W3.	$\dfrac{g \equiv eg +_b f}{g \equiv e^{(b)} f}$ if $E(e) \equiv 0$	(fixpoint)

Open Problem: Are these axioms complete for bisimilarity?

Completeness here implies completeness for language equivalence

Massaging the Syntax to Fit the Mould

$b \in \text{BExp}$ interpreted as **assert** b

b

Massaging the Syntax to Fit the Mould

$b \in$ BExp interpreted as **assert** b

The test 1 is interpreted as **assert** True

b

Massaging the Syntax to Fit the Mould

$b \in \text{BExp}$ interpreted as **assert** b

The test 1 is interpreted as **assert** True
and the test 0 is interpreted as **assert** False

b

Massaging the Syntax to Fit the Mould

$b \in \mathrm{BExp}$ interpreted as **assert** b

The test 1 is interpreted as **assert** True
and the test 0 is interpreted as **assert** False

b

Massaging the Syntax to Fit the Mould

$b \in$ BExp interpreted as **assert** b

The test 1 is interpreted as **assert** True
and the test 0 is interpreted as **assert** False

assert True is equivalent to simply **skip**

b

Massaging the Syntax to Fit the Mould

$b \in$ BExp interpreted as **assert** b

The test 1 is interpreted as **assert** True
and the test 0 is interpreted as **assert** False

assert True is equivalent to simply **skip**
assert False is equivalent to simply **crash**

b

Massaging the Syntax to Fit the Mould

$b \in \mathrm{BExp}$ interpreted as **assert** b

The test 1 is interpreted as **assert** True
and the test 0 is interpreted as **assert** False

assert True is equivalent to simply **skip**
assert False is equivalent to simply **crash**

$\mathrm{GKAT} \vdash b = 1 +_b 0$ or **if** b **then skip else crash**

Guarded Kleene Algebra with Tests modulo Bisimulation

$$\mathsf{GExp}_{\mathsf{ts}} \ni e, f ::= 0 \mid 1 \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

$$e = e +_\top f$$
$$e = e +_b e$$
$$e +_b f = f +_{\bar{b}} e$$
$$e +_b (f +_c g) = (e +_b f) +_{b \vee c} g$$

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e +_b f)g = eg +_b fg$$

$$(1 +_c e)^{(b)} = (0 +_c e)^{(b)}$$

$$e^{(b)} = ee^{(b)} +_{(b)} 1$$

$$\frac{g = eg +_{(b)} f \quad e \text{ guarded}}{g = e^{(b)} f}$$

Guarded Kleene Algebra with Tests modulo Bisimulation

$$\mathsf{GExp}_{\mathsf{ts}} \ni e, f ::= 0 \mid 1 \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

$$e = e +_{\top} f$$
$$e = e +_b e$$
$$e +_b f = f +_{\bar{b}} e$$
$$e +_b (f +_c g) = (e +_b f) +_{b \vee c} g$$

Equational Branching Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e +_b f)g = eg +_b fg$$

$$(1 +_c e)^{(b)} = (0 +_c e)^{(b)}$$

$$e^{(b)} = ee^{(b)} +_{(b)} 1$$

$$\frac{g = eg +_{(b)} f \quad e \text{ guarded}}{g = e^{(b)} f}$$

Guarded Kleene Algebra with Tests modulo Bisimulation

$$\mathsf{GExp}_{\mathsf{ts}} \ni e, f ::= 0 \mid 1 \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

$$e = e +_\top f$$
$$e = e +_b e$$
$$e +_b f = f +_{\bar{b}} e$$
$$e +_b (f +_c g) = (e +_b f) +_{b \vee c} g$$

Equational Branching Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e +_b f)g = eg +_b fg$$

Sequencing Axioms

$$(1 +_c e)^{(b)} = (0 +_c e)^{(b)}$$

$$e^{(b)} = ee^{(b)} +_{(b)} 1$$

$$\frac{g = eg +_{(b)} f \quad e \text{ guarded}}{g = e^{(b)} f}$$

Guarded Kleene Algebra with Tests modulo Bisimulation

$$\mathsf{GExp}_{\mathsf{ts}} \ni e, f ::= 0 \mid 1 \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

$$e = e +_\top f$$
$$e = e +_b e$$
$$e +_b f = f +_{\bar{b}} e$$
$$e +_b (f +_c g) = (e +_b f) +_{b \vee c} g$$

Equational Branching Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e +_b f)g = eg +_b fg$$

Sequencing Axioms

$$(1 +_c e)^{(b)} = (0 +_c e)^{(b)}$$

$$e^{(b)} = ee^{(b)} +_{(b)} 1$$

$$\frac{g = eg +_{(b)} f \quad e \text{ guarded}}{g = e^{(b)} f}$$

Unique Guarded Fixed-point Axioms

Guarded Kleene Algebra with Tests modulo Bisimulation

$$\mathsf{GExp}_{\mathsf{ts}} \ni e, f ::= 0 \mid 1 \mid p \in \Sigma \mid e +_b f \mid ef \mid e^{(b)}$$

Equational Branching Axioms

$$e = e +_\top f$$
$$e = e +_b e$$
$$e +_b f = f +_{\bar{b}} e$$
$$e +_b (f +_c g) = (e +_b f) +_{b \vee c} g$$

Sequencing Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e +_b f)g = eg +_b fg$$

Unguarded Fixed-point Axiom

$$(1 +_c e)^{(b)} = (0 +_c e)^{(b)}$$

Unique Guarded Fixed-point Axioms

$$e^{(b)} = ee^{(b)} +_{(b)} 1$$
$$\frac{g = eg +_{(b)} f \quad e \text{ guarded}}{g = e^{(b)} f}$$

How to distinguish the examples!

EQUATIONAL THEORY

$$e = e +_{\top} f$$
$$e = e +_b e$$
$$e +_b f = f +_{\bar{b}} e$$
$$e +_b (f +_c g) = (e +_b f) +_{b \vee c} g$$

Equational Branching Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e +_b f)g = eg +_b fg$$

Sequencing Axioms

Unguarded Fixed-point Axioms

FIXED POINT EQUATIONS

$$e^{(b)} = ee^{(b)} +_{(b)} 1$$

$$\frac{g = eg +_{(b)} f \quad e \text{ guarded}}{g = e^{(b)} f}$$

Unique Guarded Fixed-point Axioms

How to distinguish the examples!

EQUATIONAL THEORY

$$e = e +_{\top} f$$
$$e = e +_b e$$
$$e +_b f = f +_{\bar{b}} e$$
$$e +_b (f +_c g) = (e +_b f) +_{b \vee c} g$$

Equational Branching Axioms

$$0e = 0$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e +_b f)g = eg +_b fg$$

Sequencing Axioms

Unguarded Fixed-point Axioms

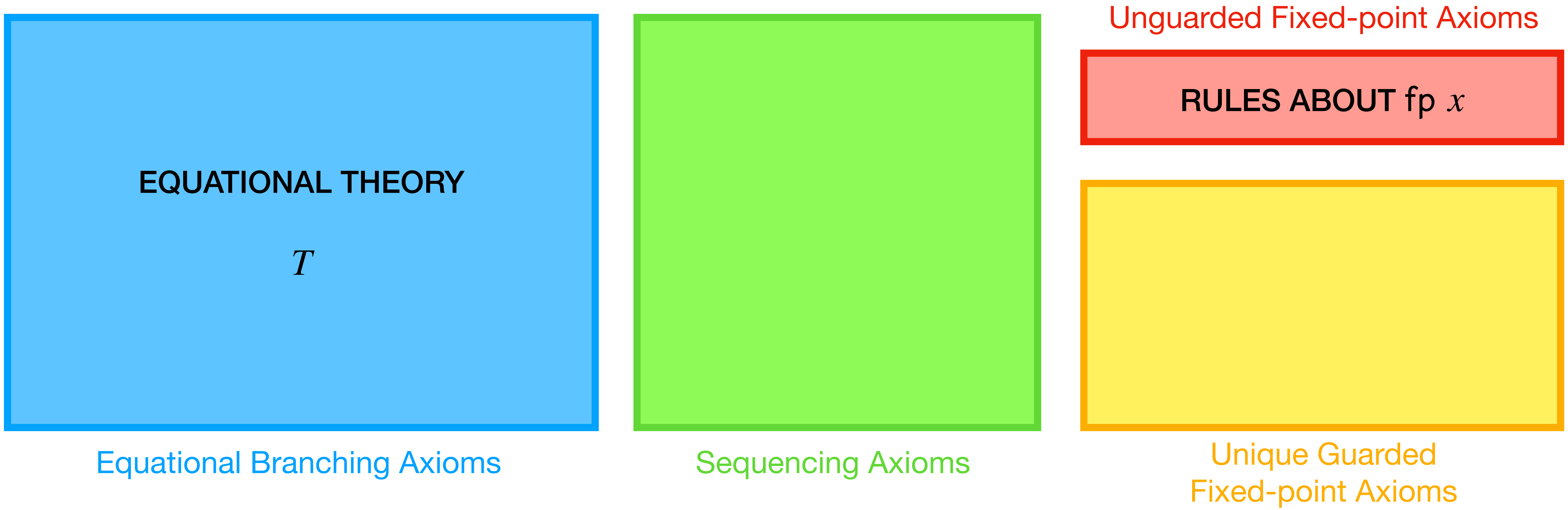
FIXED POINT EQUATIONS

$$e^{(b)} = ee^{(b)} +_{(b)} 1$$
$$\frac{g = eg +_{(b)} f \quad e \text{ guarded}}{g = e^{(b)} f}$$

Unique Guarded Fixed-point Axioms

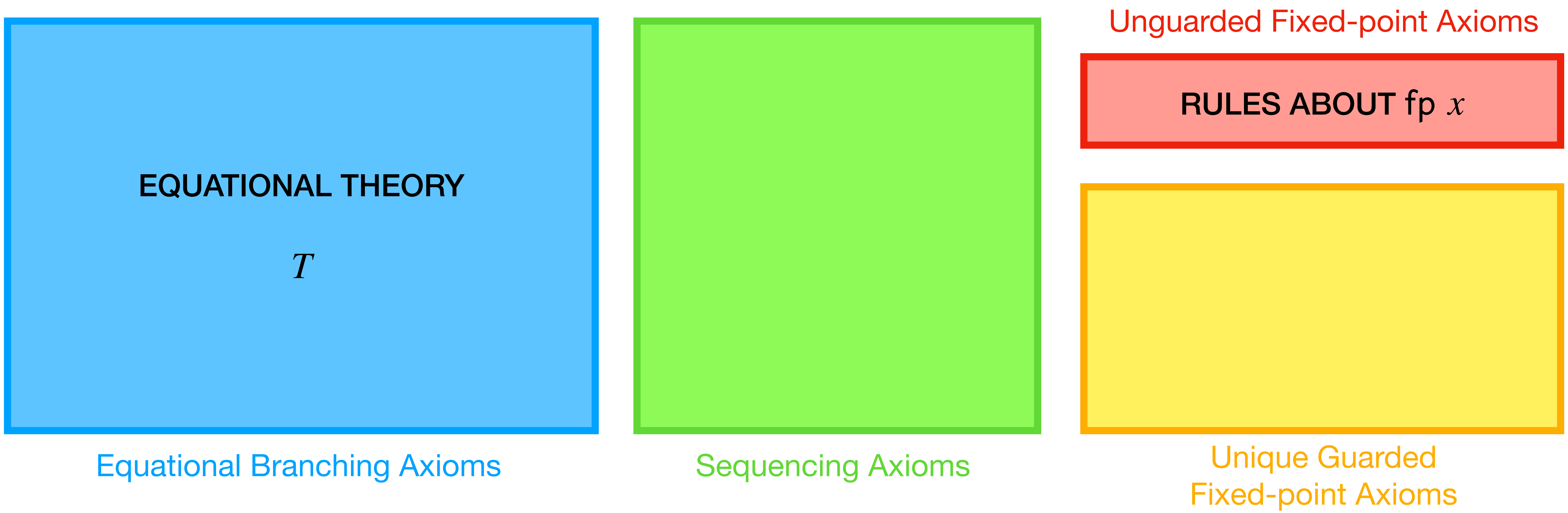
Together, this data comprises a *branching theory.*

A Recipe


Unguarded Fixed-point Axioms
RULES ABOUT fp x
EQUATIONAL THEORY
T
Equational Branching Axioms
Sequencing Axioms
Unique Guarded
Fixed-point Axioms

A Recipe

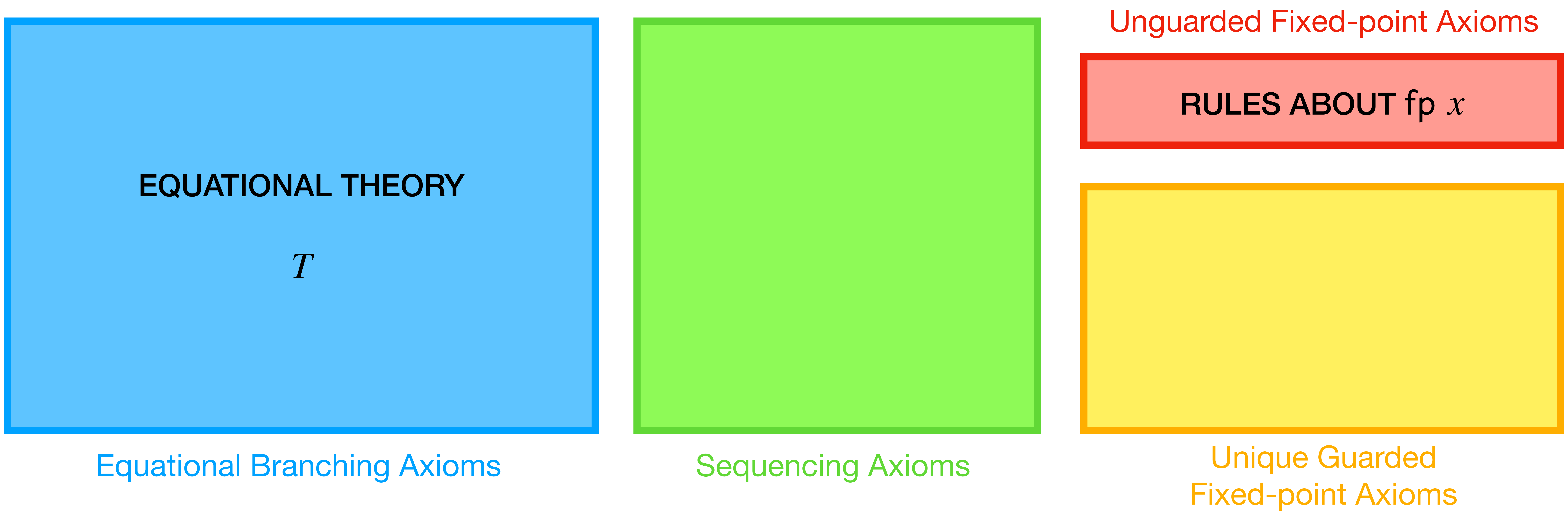
Definition. A *branching theory* consists of a



A Recipe

Definition. A *branching theory* consists of a

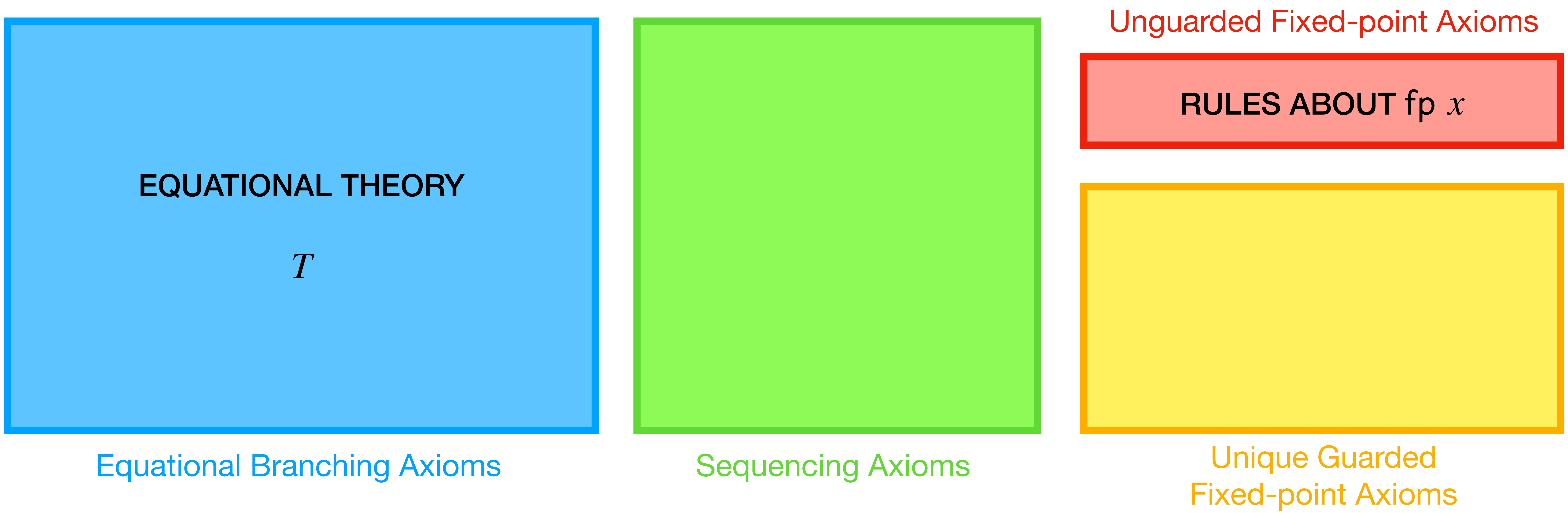
1. An algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ consisting of **constants** and **binary operations**



A Recipe

Definition. A *branching theory* consists of a

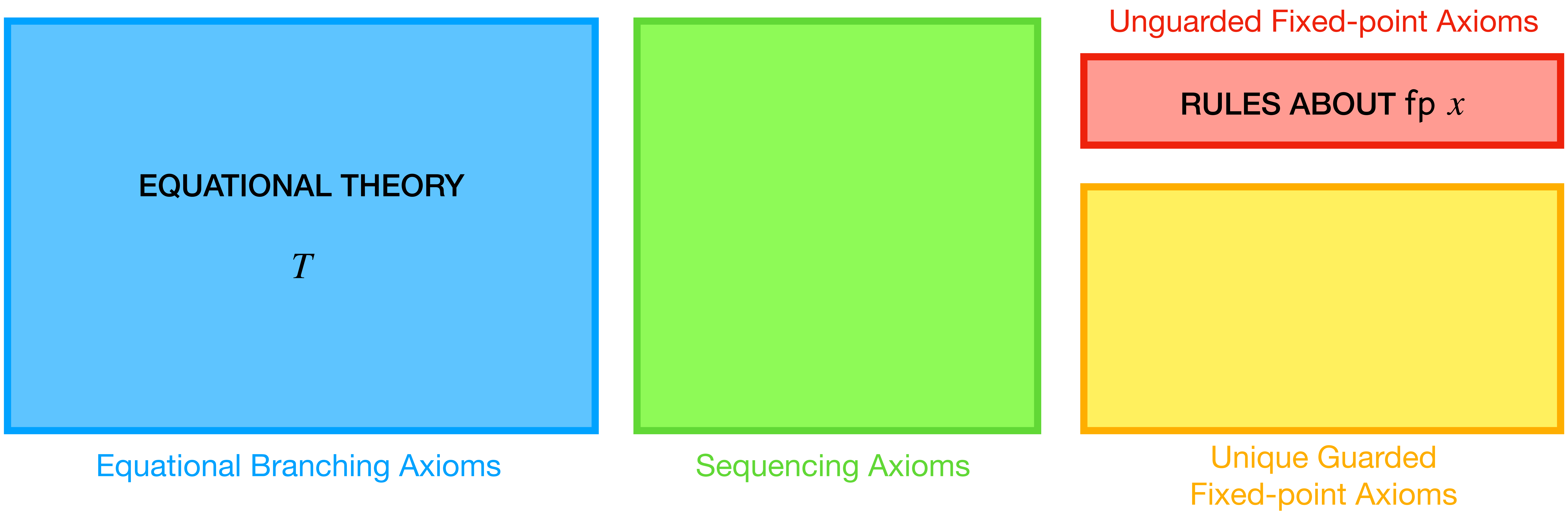
1. An algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ consisting of **constants** and **binary operations**
2. A set $T \subseteq S^*(Var) \times S^*(Var)$ of equations between S-terms



A Recipe

Definition. A *branching theory* consists of a

1. An algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ consisting of **constants** and **binary operations**
2. A set $T \subseteq S^*(Var) \times S^*(Var)$ of equations between S-terms
3. A **fixed-point operator** on S-terms $\mathsf{fp}\ x \colon S^*(\{x\} + Y) \to S^*(Y)$ (natural in Y) satisfying

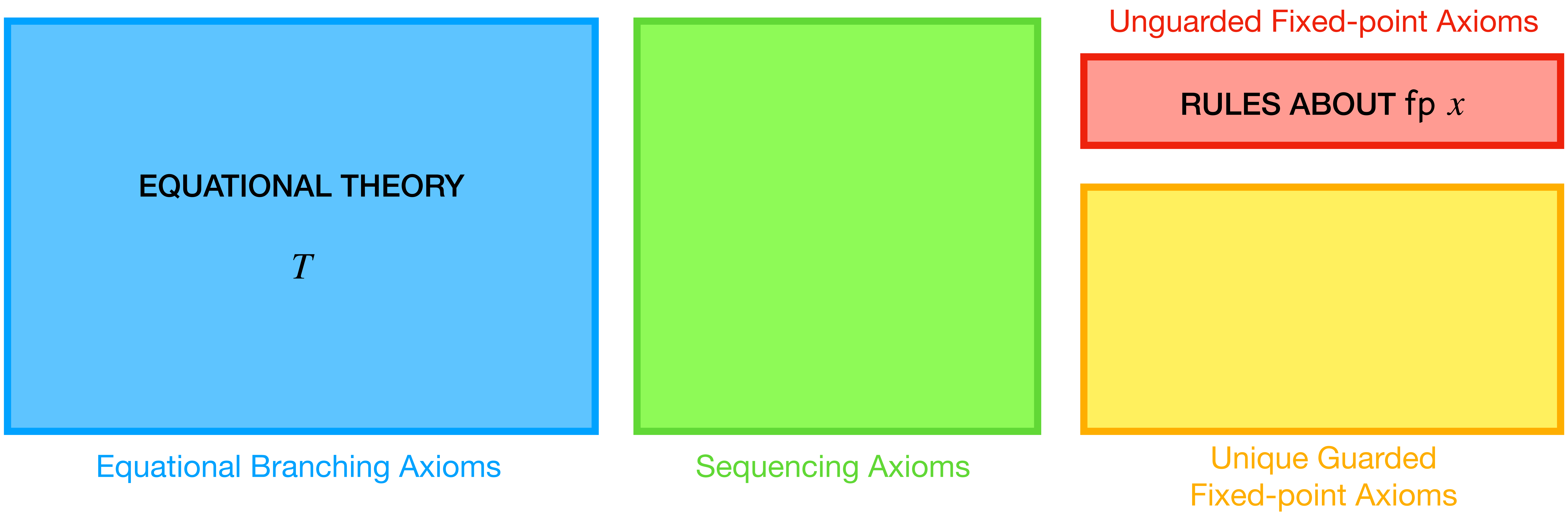


A Recipe

Definition. A *branching theory* consists of a

1. An algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ consisting of **constants** and **binary operations**
2. A set $T \subseteq S^*(Var) \times S^*(Var)$ of equations between S-terms
3. A **fixed-point operator** on S-terms fp $x\colon S^*(\{x\} + Y) \to S^*(Y)$ (natural in Y) satisfying

$$T \vdash \mathsf{fp}\ x\ t(x, \vec{y}) = t(\mathsf{fp}\ x\ t(x, \vec{y}), \vec{y})$$



Introducing: *Star Fragments!*

Definition. For a given branching theory $(S, T, \mathtt{fp})$, the set of *star expressions* is given by

Eg.

$$S_0 = \{0\}$$

$$S_2 = \{\, +_b \mid b \in \mathsf{BExp}\}$$

Introducing: *Star Fragments!*

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$$\mathsf{StExp} \ni e, f ::= c \in S_0 \qquad \textbf{raise}\ c$$

Eg.

$S_0 = \{0\}$

$S_2 = \{\ +_b \mid b \in \mathsf{BExp}\}$

Introducing: *Star Fragments!*

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$$\mathsf{StExp} \ni e, f ::= c \in S_0 \qquad \textbf{raise } c$$
$$\mid 1 \qquad \textbf{skip}$$

Eg.

$S_0 = \{0\}$

$S_2 = \{ +_b \mid b \in \mathsf{BExp}\}$

Introducing: ***Star Fragments!***

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$$\mathsf{StExp} \ni e, f ::= c \in S_0 \qquad \textbf{raise } c$$

$$\mid 1 \qquad \textbf{skip}$$

$$\mid e +_\sigma f \qquad \textbf{branch into } \sigma(e, f)\text{, where } \sigma \in S_2$$

Eg.

$S_0 = \{0\}$

$S_2 = \{ +_b \mid b \in \mathsf{BExp}\}$

Introducing: *Star Fragments!*

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$\mathsf{StExp} \ni e, f ::= c \in S_0$	**raise** c
$\mid 1$	**skip**
$\mid e +_\sigma f$	**branch into** $\sigma(e, f)$, where $\sigma \in S_2$
$\mid ef$	$e;f$

Eg.

$S_0 = \{0\}$

$S_2 = \{ +_b \mid b \in \mathsf{BExp}\}$

Introducing: *Star Fragments!*

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$$\mathsf{StExp} \ni e, f ::= c \in S_0 \qquad \textbf{raise } c$$
$$\mid 1 \qquad \textbf{skip}$$
$$\mid e +_\sigma f \qquad \textbf{branch into } \sigma(e, f)\text{, where } \sigma \in S_2$$
$$\mid ef \qquad e;f$$
$$\mid e^{(\sigma)} \qquad \textbf{recurse in } x = \sigma(e; x, \top\ )$$

Eg.

$S_0 = \{0\}$

$S_2 = \{\ +_b \mid b \in \mathsf{BExp}\}$

Introducing: ***Star Fragments!***

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$\mathsf{StExp} \ni e, f ::= c \in S_0$	**raise** c
$\mid 1$	**skip**
$\mid e +_\sigma f$	**branch into** $\sigma(e, f)$, where $\sigma \in S_2$
$\mid ef$	$e;f$
$\mid e^{(\sigma)}$	**recurse in** $x = \sigma(e; x, \top)$

Eg.

$S_0 = \{0\}$

$S_2 = \{ +_b \mid b \in \mathsf{BExp}\}$

Introducing: *Star Fragments!*

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$\mathsf{StExp} \ni e, f ::= c \in S_0$	**raise** c
$\mid 1$	**skip**
$\mid e +_\sigma f$	**branch into** $\sigma(e, f)$, where $\sigma \in S_2$
$\mid ef$	$e;f$
$\mid e^{(\sigma)}$	**recurse in** $x = \sigma(e; x, \top)$

Eg. $\mathsf{GExp}_{ts} \ni e, f ::= 0$ **crash** $S_0 = \{0\}$

$S_2 = \{ +_b \mid b \in \mathsf{BExp}\}$

Introducing: *Star Fragments!*

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$\mathsf{StExp} \ni e, f ::= c \in S_0$	**raise** c
$\mid 1$	**skip**
$\mid e +_\sigma f$	**branch into** $\sigma(e, f)$, where $\sigma \in S_2$
$\mid ef$	$e;f$
$\mid e^{(\sigma)}$	**recurse in** $x = \sigma(e; x, \top)$

Eg.

$\mathsf{GExp}_{ts} \ni e, f ::= 0$	**crash**	$S_0 = \{0\}$
$\mid 1$	**skip**	
		$S_2 = \{ +_b \mid b \in \mathsf{BExp}\}$

Introducing: *Star Fragments!*

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$\mathsf{StExp} \ni e, f ::= c \in S_0$	**raise** c
$\mid 1$	**skip**
$\mid e +_\sigma f$	**branch into** $\sigma(e, f)$, where $\sigma \in S_2$
$\mid ef$	$e;f$
$\mid e^{(\sigma)}$	**recurse in** $x = \sigma(e; x, \top)$

Eg.

$\mathsf{GExp}_{ts} \ni e, f ::= 0$	**crash**	$S_0 = \{0\}$
$\mid 1$	**skip**	
$\mid e +_b f$	**if** b **then** e **else** f	$S_2 = \{ +_b \mid b \in \mathsf{BExp}\}$

Introducing: *Star Fragments!*

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$\mathsf{StExp} \ni e, f ::= c \in S_0$	**raise** c	
$\mid 1$	**skip**	
$\mid e +_\sigma f$	**branch into** $\sigma(e, f)$, where $\sigma \in S_2$	
$\mid ef$	$e;f$	
$\mid e^{(\sigma)}$	**recurse in** $x = \sigma(e; x, \top)$	

Eg.

$\mathsf{GExp}_{ts} \ni e, f ::= 0$	**crash**	$S_0 = \{0\}$
$\mid 1$	**skip**	
$\mid e +_b f$	**if** b **then** e **else** f	$S_2 = \{ +_b \mid b \in \mathsf{BExp}\}$
$\mid ef$	$e;f$	

Introducing: *Star Fragments!*

Definition. For a given branching theory (S, T, fp), the set of *star expressions* is given by

$\mathsf{StExp} \ni e,f ::= c \in S_0$	**raise** c
$\mid 1$	**skip**
$\mid e +_\sigma f$	**branch into** $\sigma(e,f)$, where $\sigma \in S_2$
$\mid ef$	$e;f$
$\mid e^{(\sigma)}$	**recurse in** $x = \sigma(e; x, \top)$

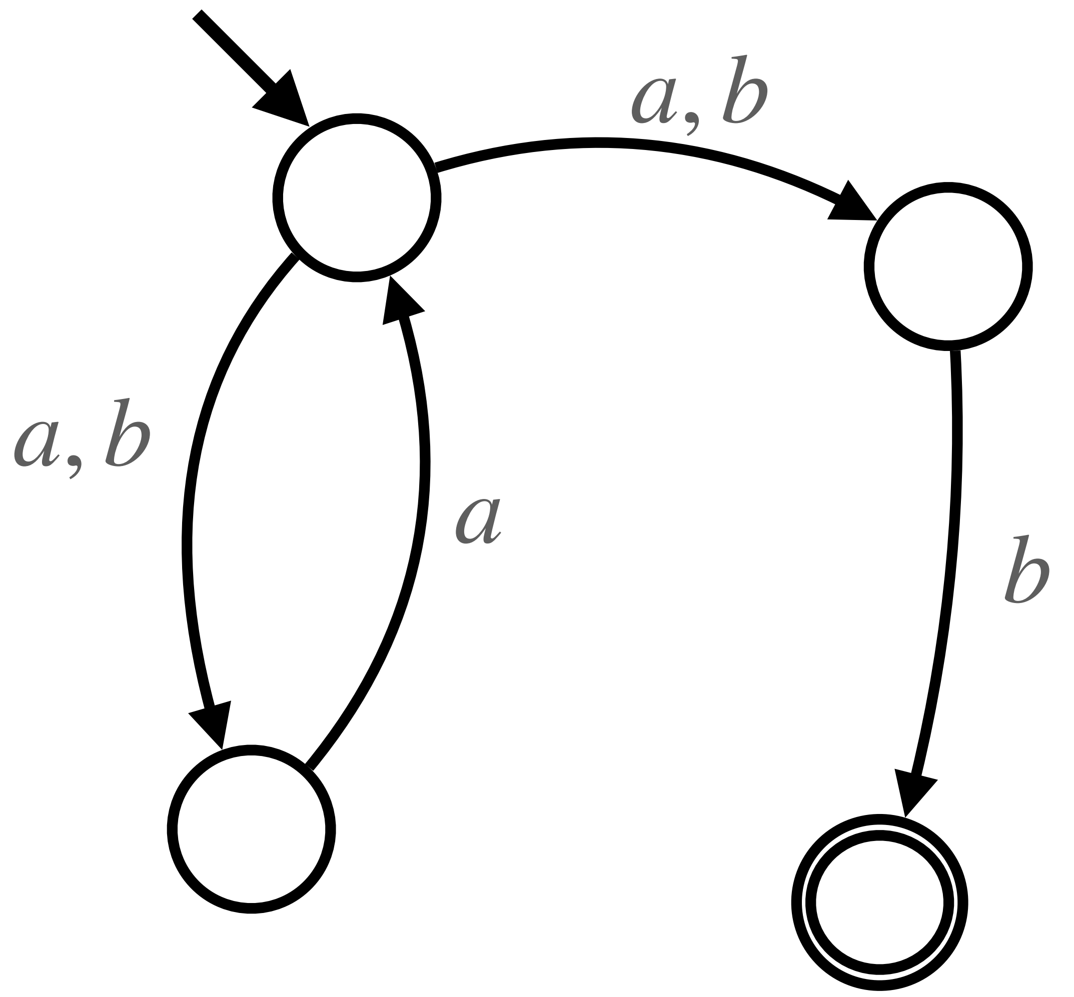
Eg.

$\mathsf{GExp}_{ts} \ni e,f ::= 0$	**crash**	$S_0 = \{0\}$
$\mid 1$	**skip**	
$\mid e +_b f$	**if** b **then** e **else** f	$S_2 = \{ +_b \mid b \in \mathsf{BExp}\}$
$\mid ef$	$e;f$	
$\mid e^{(b)}$	**while** b **do** e	

Star Fragment Semantics

Operational semantics of regular expressions modulo bisimilarity:

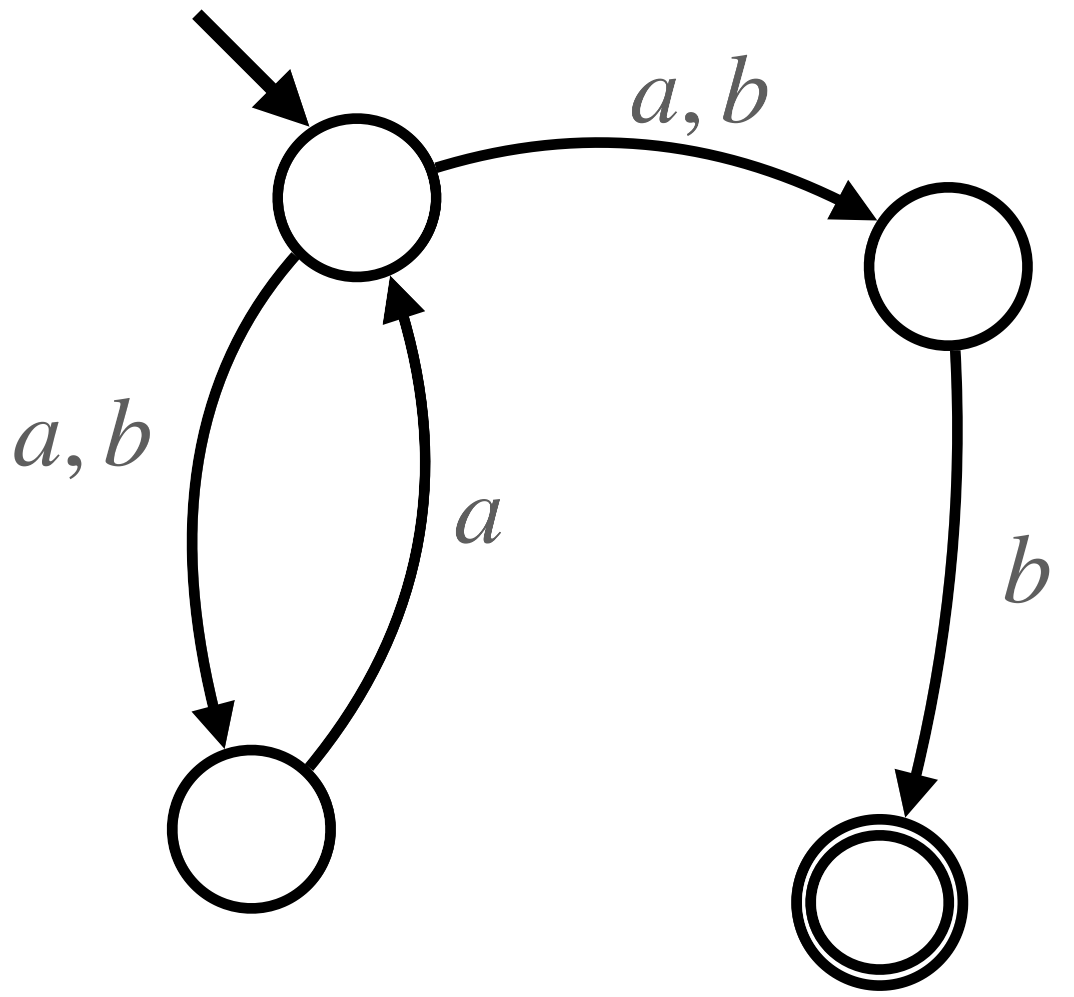
$$\mathsf{Exp} \longrightarrow \{\ \bot\ ,\ \top\ \} \times \mathscr{P}_{fin}(\mathsf{Exp})^A$$



Star Fragment Semantics

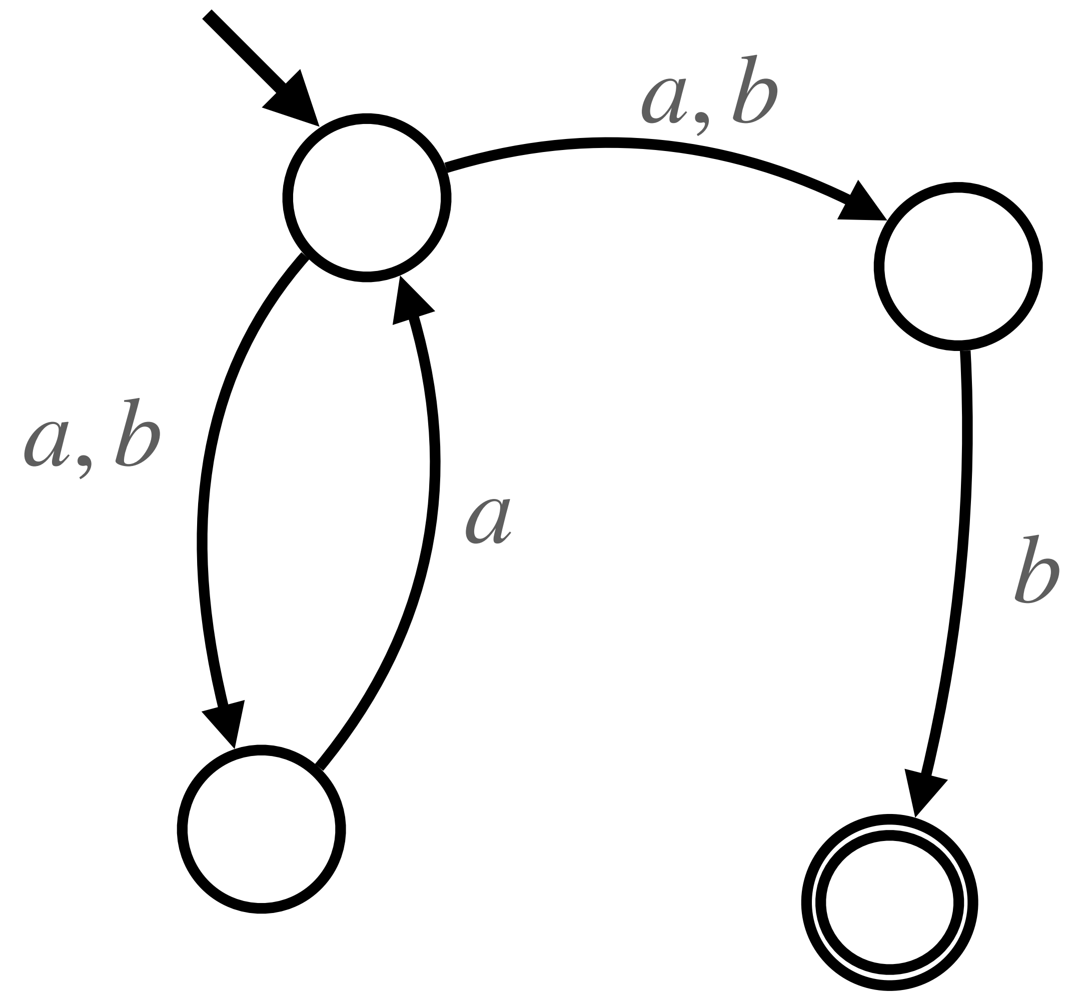
Operational semantics of regular expressions modulo bisimilarity:

$$\ell : \mathsf{Exp} \longrightarrow \mathcal{P}_{fin}(\top + A \times \mathsf{Exp})$$



Star Fragment Semantics

Operational semantics of regular expressions modulo bisimilarity:

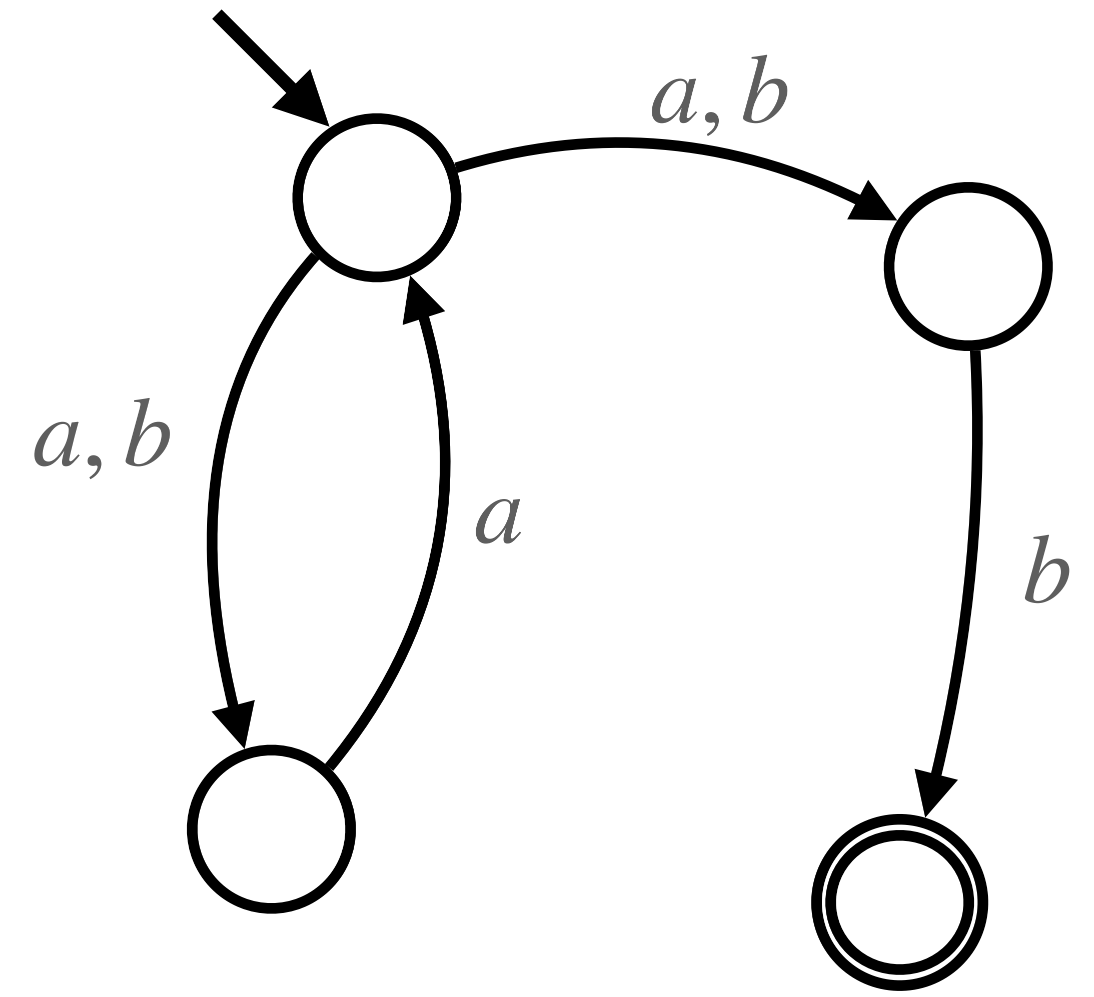


$$\ell : \mathsf{Exp} \longrightarrow \mathscr{P}_{fin}(\top + A \times \mathsf{Exp})$$

$$\ell(0) = \emptyset \qquad \ell(1) = \{\top\} \qquad \ell(a) = \{(a, 1)\} \qquad \ell(e + f) = \ell(e) \cup \ell(f)$$

Star Fragment Semantics

Operational semantics of regular expressions modulo bisimilarity:



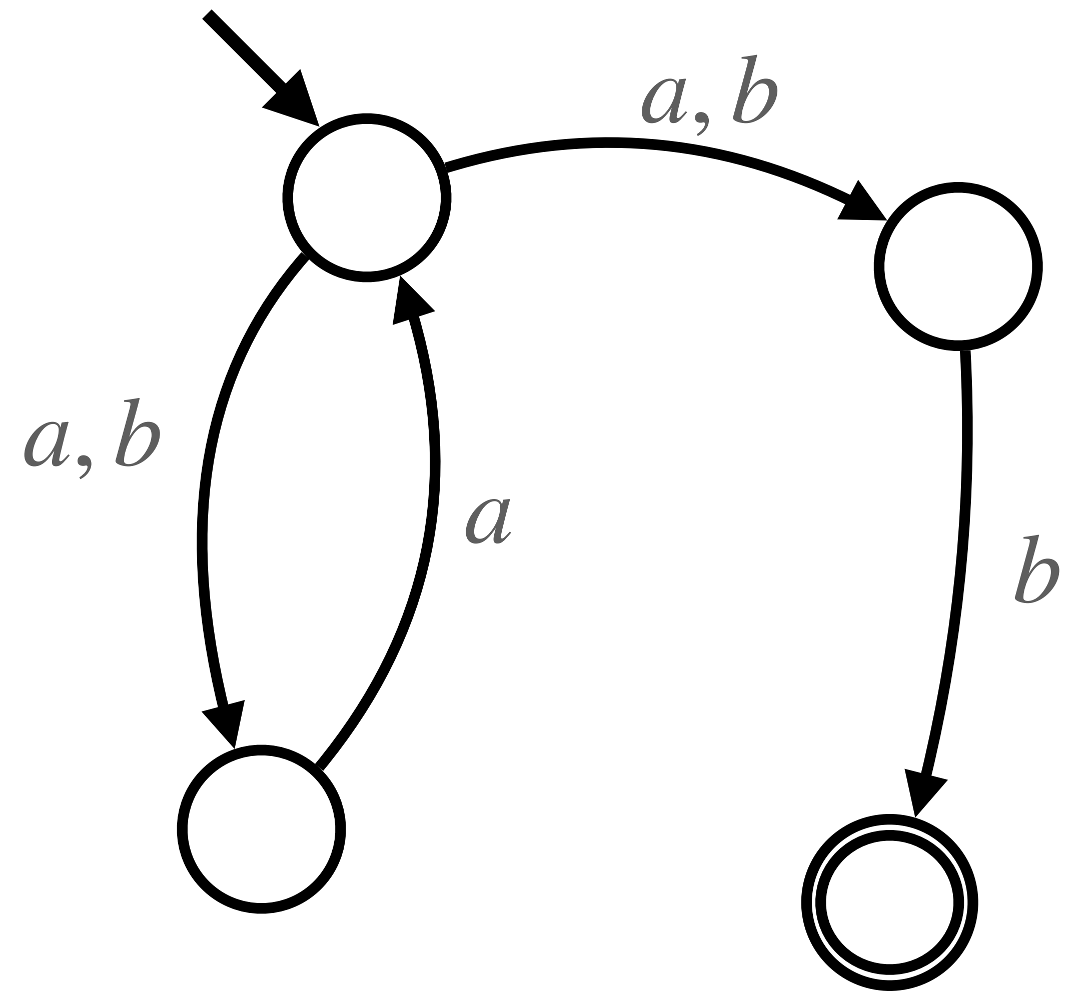
$$\ell : \mathsf{Exp} \longrightarrow \mathcal{P}_{fin}(\top + A \times \mathsf{Exp})$$

$$\ell(0) = \emptyset \qquad \ell(1) = \{\top\} \qquad \ell(a) = \{(a, 1)\} \qquad \ell(e+f) = \ell(e) \cup \ell(f)$$

and if $\ell(e) = \{\top, (a_1, e_1), \ldots, (a_n, e_n)\}$, then

Star Fragment Semantics

Operational semantics of regular expressions modulo bisimilarity:



$$\ell : \text{Exp} \longrightarrow \mathcal{P}_{fin}(\top + A \times \text{Exp})$$

$$\ell(0) = \varnothing \qquad \ell(1) = \{\top\} \qquad \ell(a) = \{(a,1)\} \qquad \ell(e+f) = \ell(e) \cup \ell(f)$$

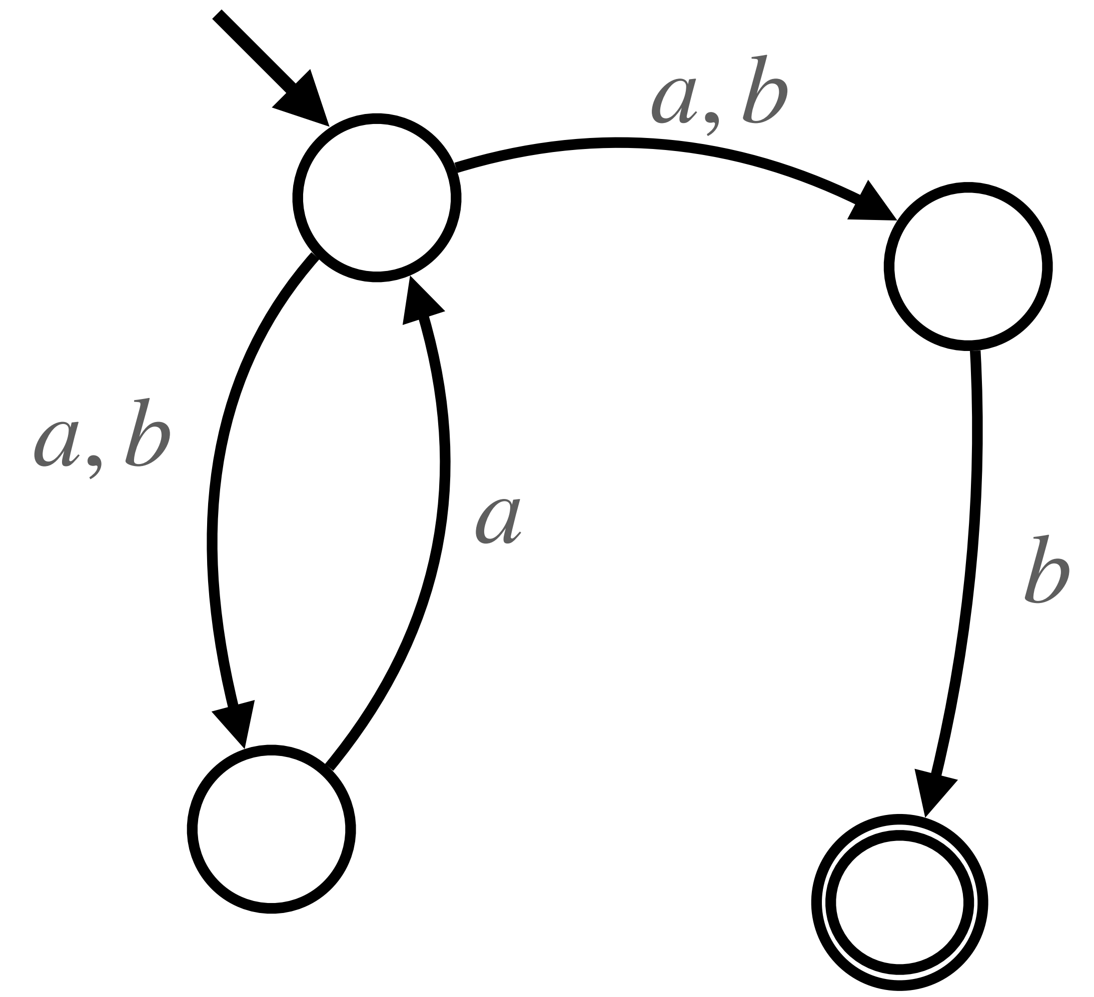
and if $\ell(e) = \{\top, (a_1, e_1), \ldots, (a_n, e_n)\}$, then

$$\ell(ef) = \ell(f) \cup \{(a_1, e_1 f), \ldots, (a_n, e_n f)\} \quad \text{and} \quad \ell(e^*) = \{\top, (a_1, e_1 e^*), \ldots, (a_n, e_n e^*)\}$$

Star Fragment Semantics

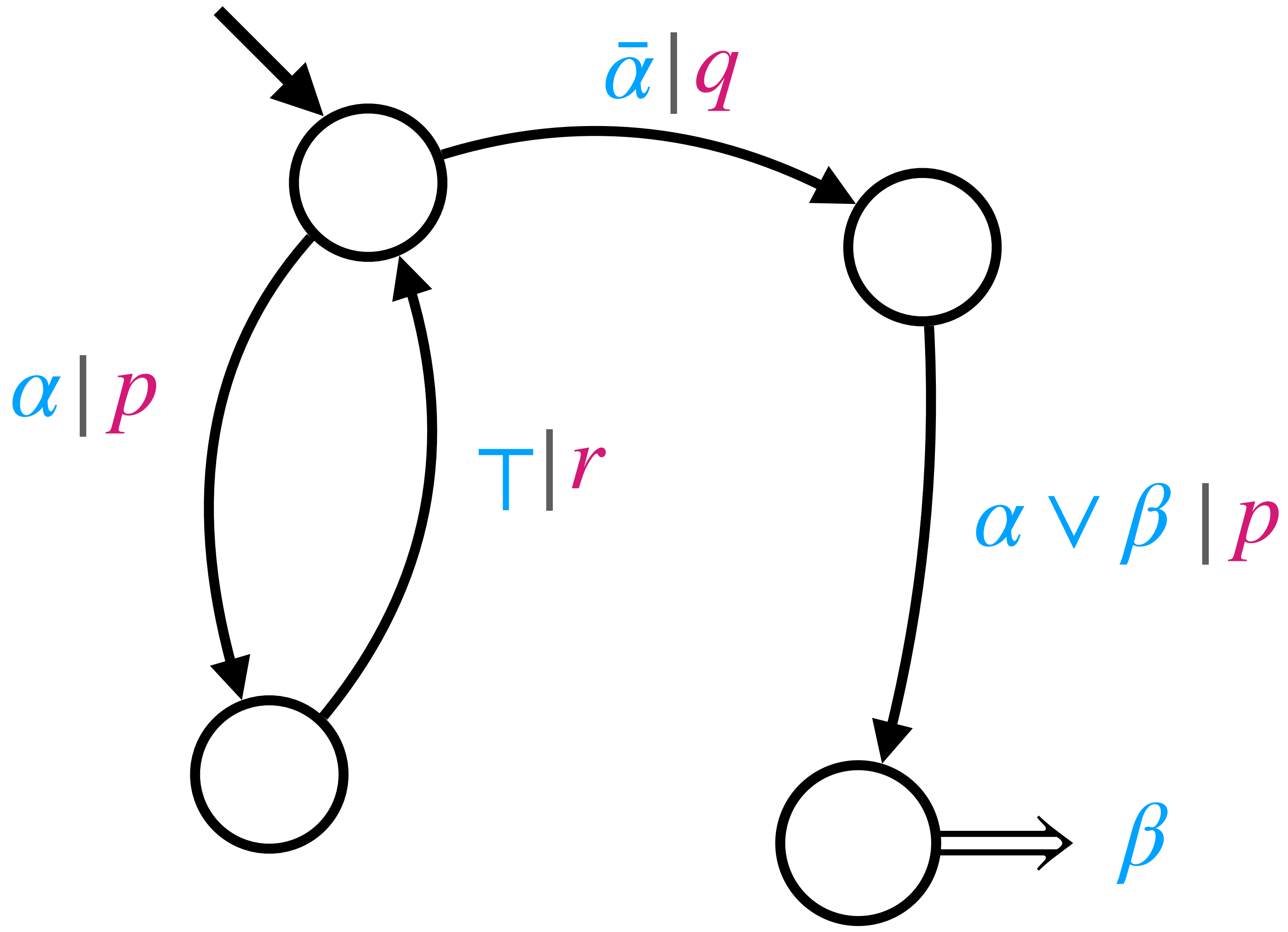
Operational semantics of regular expressions modulo bisimilarity:

$$\mathrm{Exp} \longrightarrow \mathscr{P}_{fin}(\top + A \times \mathrm{Exp})$$



Operational semantics of GKAT expressions modulo bisimilarity:

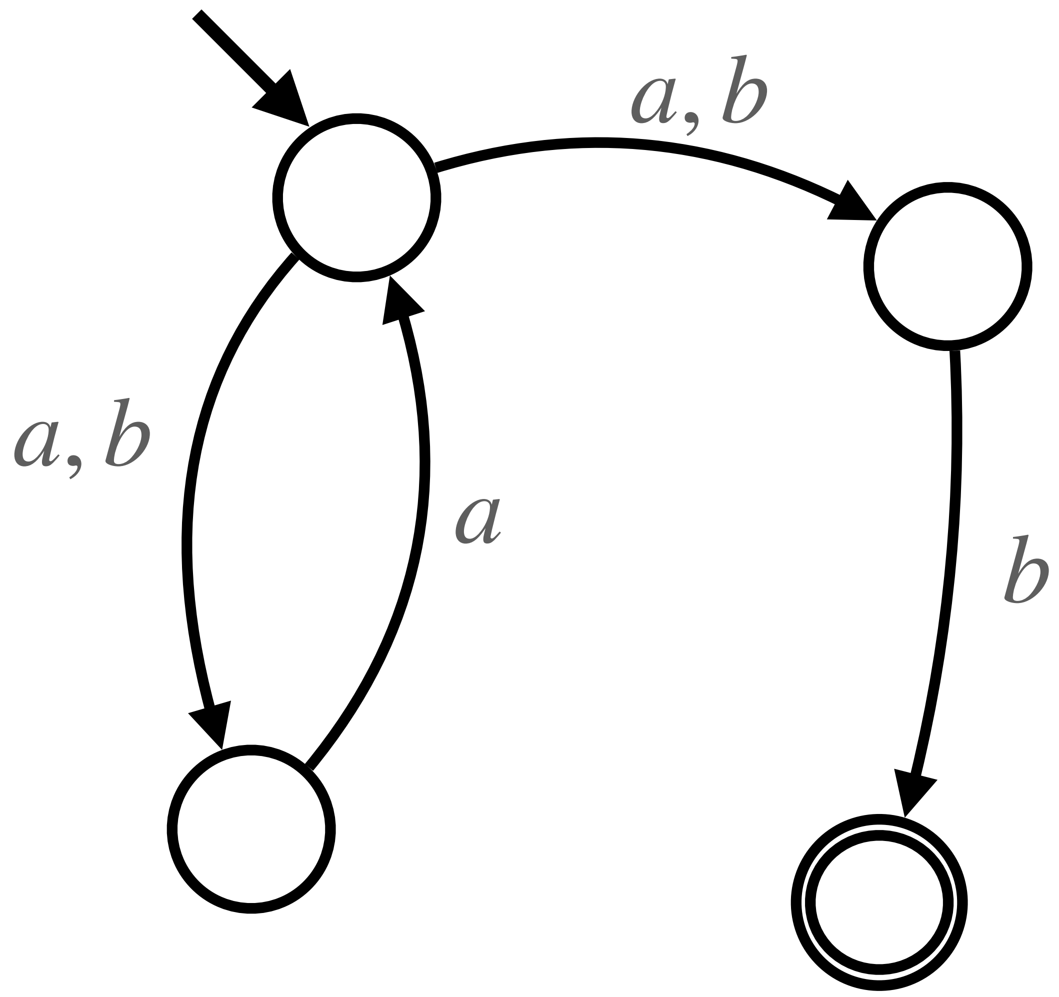
$$\mathrm{GExp} \longrightarrow (\{\perp, \top\} + \Sigma \times \mathrm{GExp})^{At}$$



Star Fragment Semantics

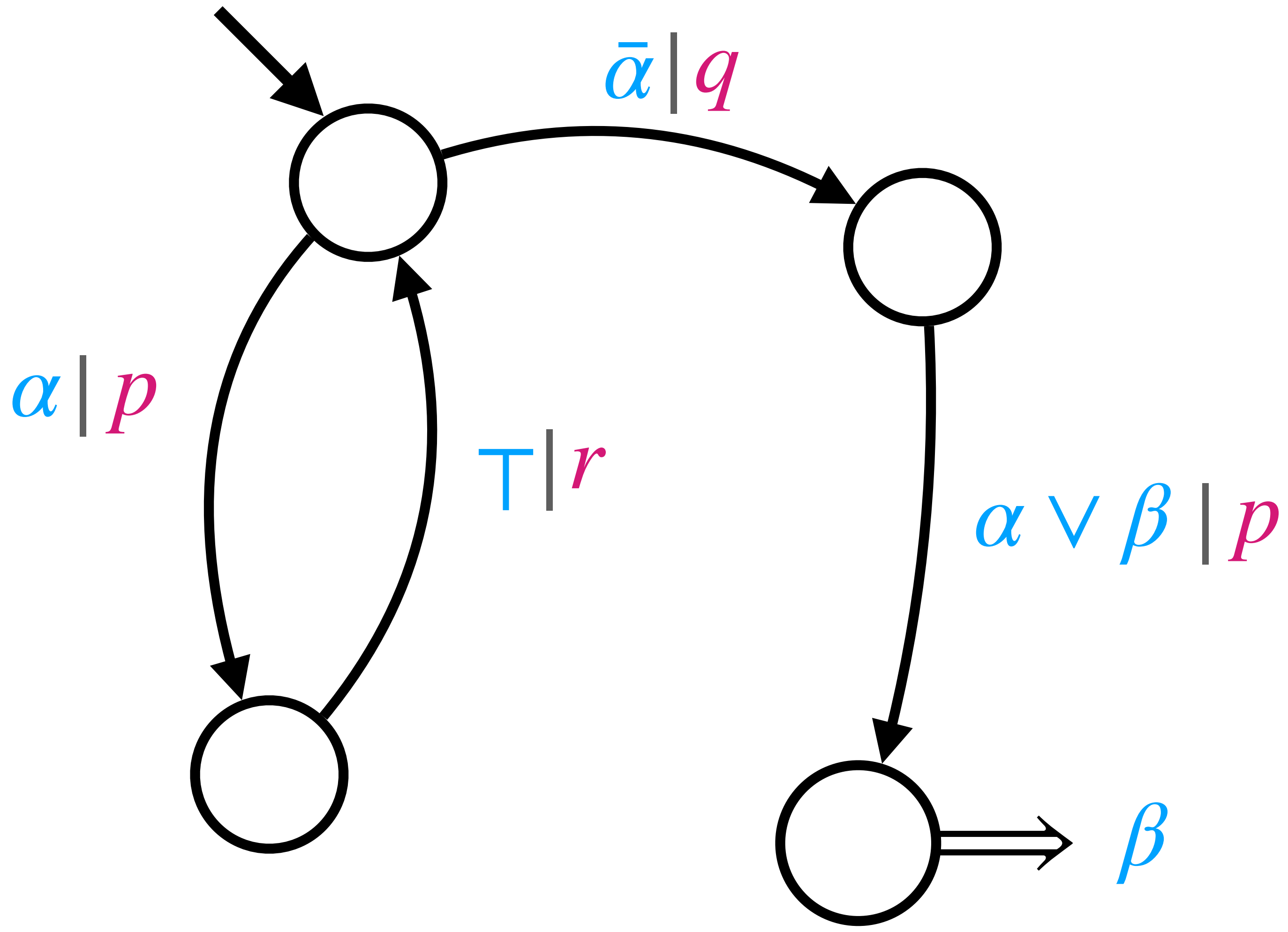
Operational semantics of regular expressions modulo bisimilarity:

$$\mathsf{Exp} \longrightarrow \mathscr{P}_{fin}(\top + A \times \mathsf{Exp})$$



Operational semantics of GKAT expressions modulo bisimilarity:

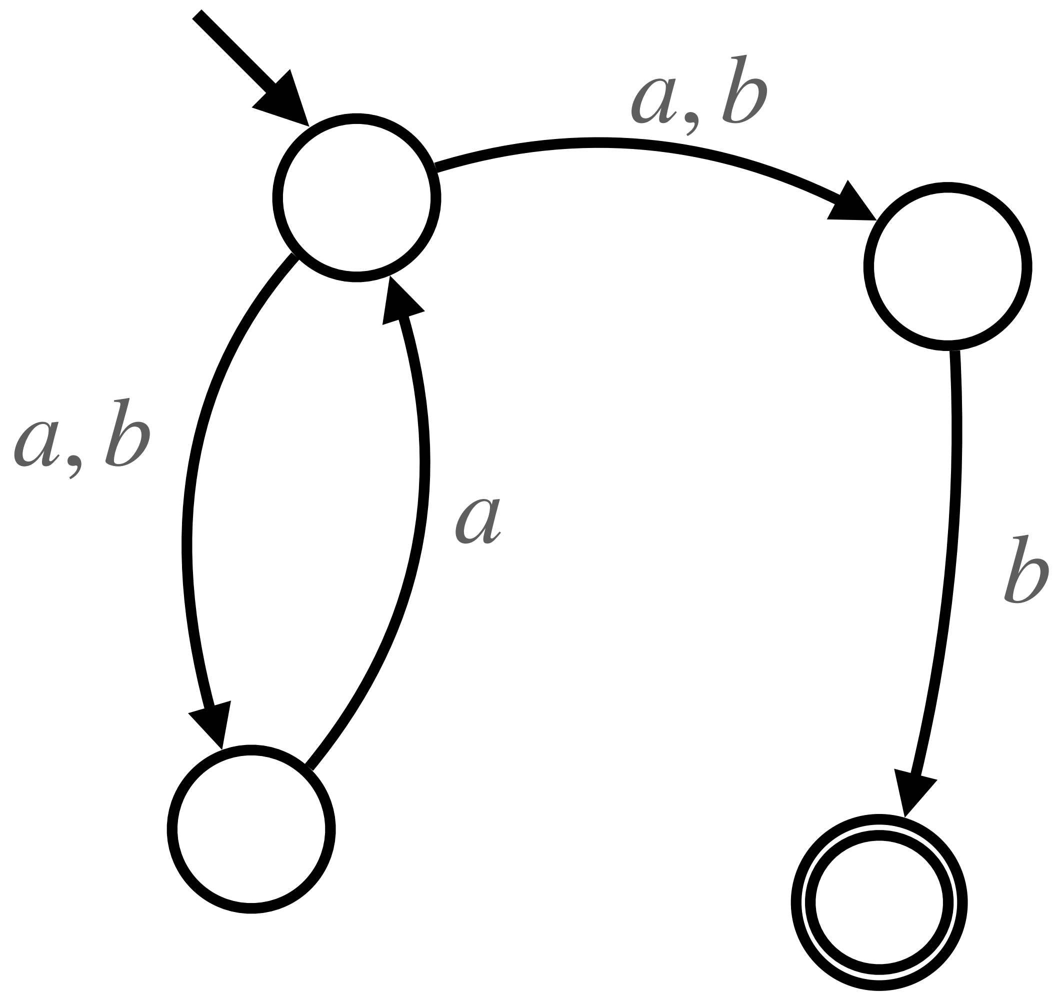
$$\mathsf{GExp} \longrightarrow (\bot + (\top + \Sigma \times \mathsf{GExp}))^{At}$$



Star Fragment Semantics

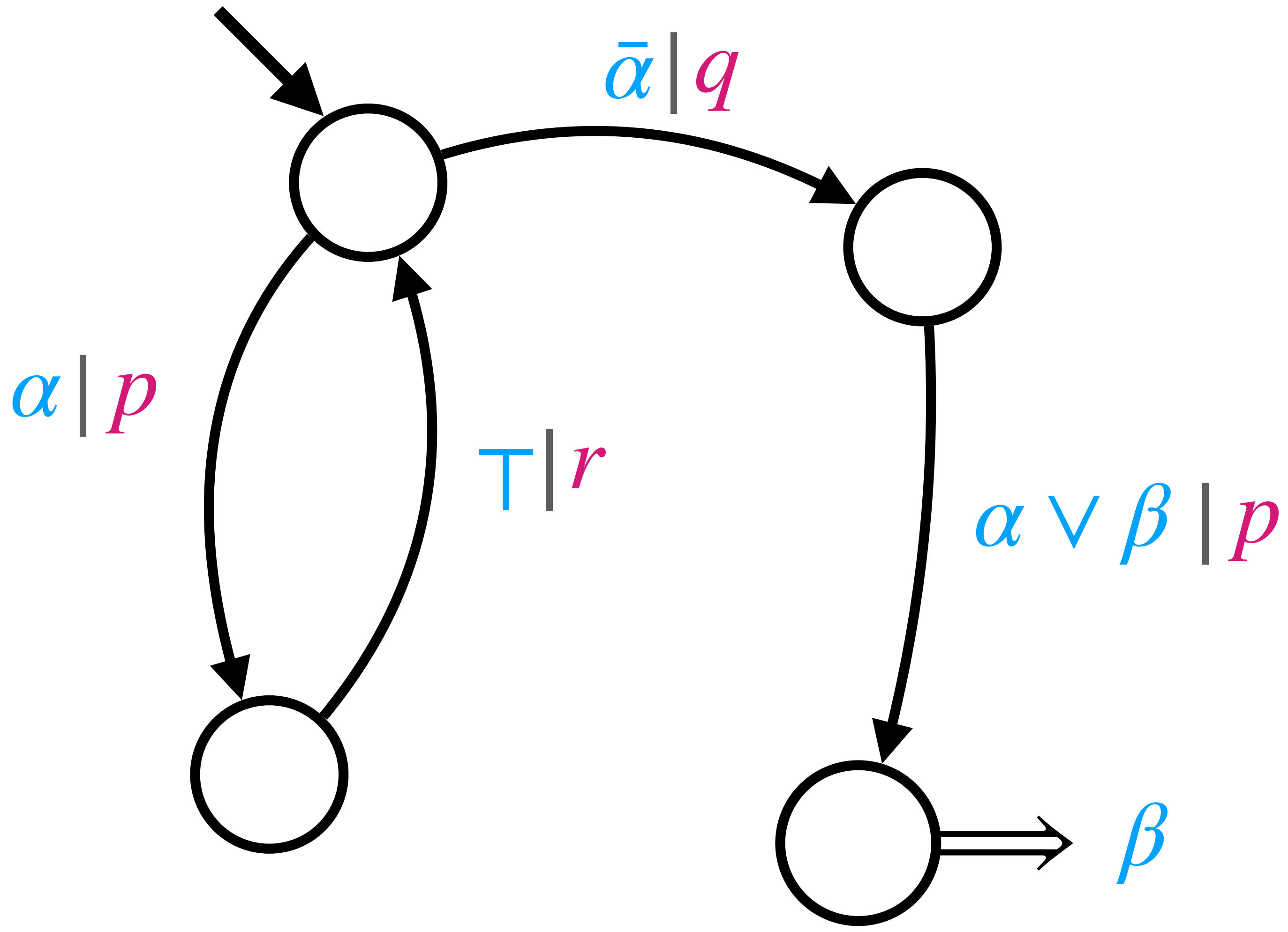
Operational semantics of regular expressions modulo bisimilarity:

$$\mathsf{Exp} \longrightarrow \mathscr{P}_{fin}(\top + Act \times \mathsf{Exp})$$



Operational semantics of GKAT expressions modulo bisimilarity:

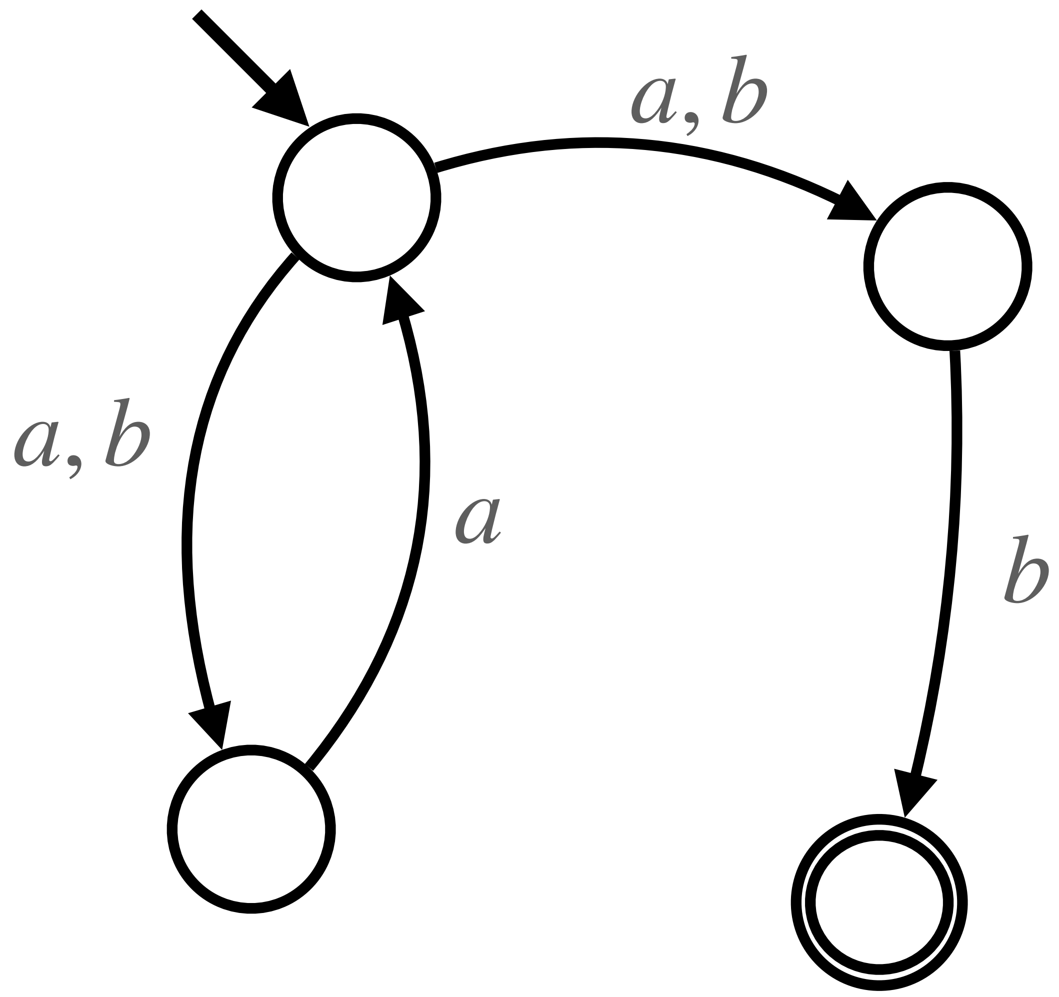
$$\mathsf{GExp} \longrightarrow (\bot + (\top + Act \times \mathsf{GExp}))^{At}$$



Star Fragment Semantics

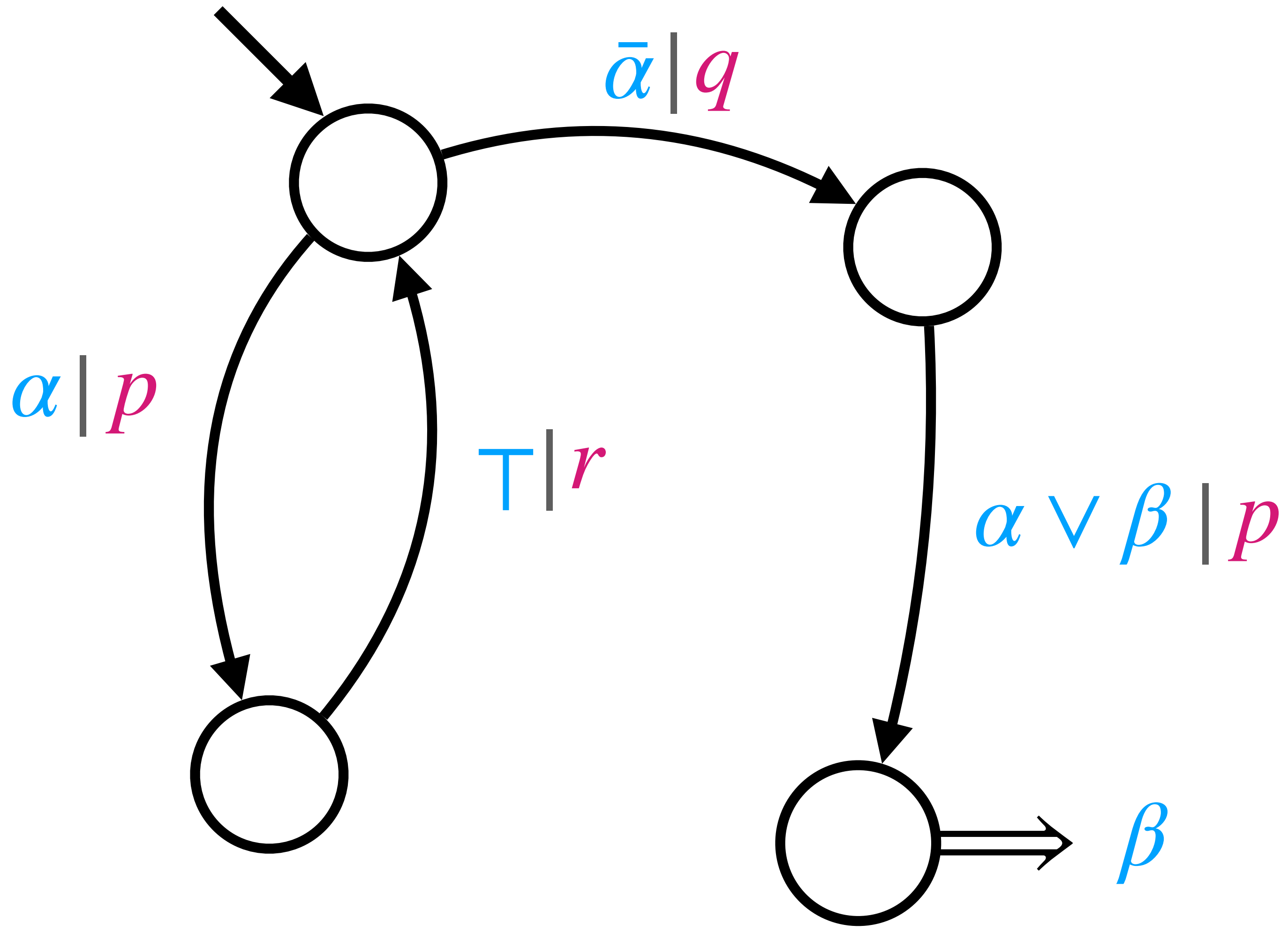
Operational semantics of regular expressions modulo bisimilarity:

$$\mathsf{Exp} \longrightarrow \mathscr{P}_{fin}(\top + Act \times \mathsf{Exp})$$



Operational semantics of GKAT expressions modulo bisimilarity:

$$\mathsf{GExp} \longrightarrow (\bot + (\top + Act \times \mathsf{GExp}))^{At}$$



Observe: Format is $\top + Act \times (-)$ **wrapped in** $M(-)$**.**

$\mathscr{P}_{fin}(-)$ **— the finite powerset monad**

$(\bot + (-))^{At}$ **— the partial functions monad**

Star Fragment Semantics: Branching Types

Fix an algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ and a set of equations $T \subseteq S^*(V) \times S^*(V)$.

Star Fragment Semantics: Branching Types

Fix an algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ and a set of equations $T \subseteq S^*(V) \times S^*(V)$.

Definition. A monad is M *presented* by the equational theory (S, T) if there is an isomorphism

$$M \cong S^*(-)/=_T$$

i.e., the monad M is a *free-algebra construction* for (S, T).

Star Fragment Semantics: Branching Types

Fix an algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ and a set of equations $T \subseteq S^*(V) \times S^*(V)$.

Definition. A monad is M *presented* by the equational theory (S, T) if there is an isomorphism

$$M \cong S^*(\,-\,)/ =_T$$

i.e., the monad M is a *free-algebra construction* for (S, T).

Example. The equational theory in Salomaa/Milner's axioms captures *semilattices with bottom*.

$$\begin{aligned} e &= e + 0 \\ e &= e + e \\ f + e &= e + f \\ e + (f + g) &= (e + f) + g \end{aligned}$$

Star Fragment Semantics: Branching Types

Fix an algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ and a set of equations $T \subseteq S^*(V) \times S^*(V)$.

Definition. A monad is M *presented* by the equational theory (S, T) if there is an isomorphism

$$M \cong S^*(-)/=_T$$

i.e., the monad M is a *free-algebra construction* for (S, T).

Example. The equational theory in Salomaa/Milner's axioms captures *semilattices with bottom*.

$$\begin{aligned} e &= e + 0 \\ e &= e + e \\ f + e &= e + f \\ e + (f + g) &= (e + f) + g \end{aligned}$$

presents

Star Fragment Semantics: Branching Types

Fix an algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ and a set of equations $T \subseteq S^*(V) \times S^*(V)$.

Definition. A monad is M *presented* by the equational theory (S, T) if there is an isomorphism

$$M \cong S^*(-)/=_T$$

i.e., the monad M is a *free-algebra construction* for (S, T).

Example. The equational theory in Salomaa/Milner's axioms captures *semilattices with bottom*.

$$\begin{aligned} e &= e + 0 \\ e &= e + e \\ f + e &= e + f \\ e + (f + g) &= (e + f) + g \end{aligned}$$

presents →

Finite Powerset Monad $\mathscr{P}_{fin}$

Star Fragment Semantics: Branching Types

Fix an algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ and a set of equations $T \subseteq S^*(V) \times S^*(V)$.

Definition. A monad is M *presented* by the equational theory (S, T) if there is an isomorphism

$$M \cong S^*(\,-\,)/=_T$$

i.e., the monad M is a *free-algebra construction* for (S, T).

Example. The equational theory in Salomaa/Milner's axioms captures *semilattices with bottom*.

$$e = e + 0$$
$$e = e + e$$
$$f + e = e + f$$
$$e + (f + g) = (e + f) + g$$

presents →

Finite Powerset Monad $\mathscr{P}_{fin}$

$$U_1, U_2 \in \mathscr{P}_{fin}(X)$$

Star Fragment Semantics: Branching Types

Fix an algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ and a set of equations $T \subseteq S^*(V) \times S^*(V)$.

Definition. A monad is M *presented* by the equational theory (S, T) if there is an isomorphism

$$M \cong S^*(-)/=_T$$

i.e., the monad M is a *free-algebra construction* for (S, T).

Example. The equational theory in Salomaa/Milner's axioms captures *semilattices with bottom*.

$$e = e + 0$$
$$e = e + e$$
$$f + e = e + f$$
$$e + (f + g) = (e + f) + g$$

presents →

Finite Powerset Monad $\mathcal{P}_{fin}$

$$U_1, U_2 \in \mathcal{P}_{fin}(X)$$
$$U_1 + U_2 = U_1 \cup U_2$$

Star Fragment Semantics: Branching Types

Fix an algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ and a set of equations $T \subseteq S^*(V) \times S^*(V)$.

Definition. A monad is M *presented* by the equational theory (S, T) if there is an isomorphism

$$M \cong S^*(-)/=_T$$

i.e., the monad M is a *free-algebra construction* for (S, T).

Example. The equational theory in Salomaa/Milner's axioms captures *semilattices with bottom*.

$$e = e + 0$$
$$e = e + e$$
$$f + e = e + f$$
$$e + (f + g) = (e + f) + g$$

presents →

Finite Powerset Monad $\mathscr{P}_{fin}$

$$U_1, U_2 \in \mathscr{P}_{fin}(X)$$
$$U_1 + U_2 = U_1 \cup U_2$$
$$0 = \emptyset$$

Star Fragment Semantics: Branching Types

Fix an algebraic signature $S = S_0 + S_2 \times \mathrm{Id}^2$ and a set of equations $T \subseteq S^*(V) \times S^*(V)$.

Definition. A monad is M *presented* by the equational theory (S, T) if there is an isomorphism

$$M \cong S^*(-)/=_T$$

i.e., the monad M is a *free-algebra construction* for (S, T).

Example. The equational theory in Salomaa/Milner's axioms captures *semilattices with bottom*.

$$e = e + 0$$
$$e = e + e$$
$$f + e = e + f$$
$$e + (f + g) = (e + f) + g$$

presents →

Finite Powerset Monad $\mathscr{P}_{fin}$

$$U_1, U_2 \in \mathscr{P}_{fin}(X)$$
$$U_1 + U_2 = U_1 \cup U_2$$
$$0 = \emptyset$$

Definition. A monad that is presented by (S, T) is a *branching type* of the branching theory.

Star Fragment Semantics: Unguarded Fixed-points

Last ingredient of a branching theory is the **fixed-point operator** $\mathsf{fp}\ x \colon S^*(\{x\} + Y) \to S^*(Y)$

$$T \vdash \mathsf{fp}\ x\ t(x, \vec{y}) = t(\mathsf{fp}\ x\ t(x, \vec{y}), \vec{y})$$

Star Fragment Semantics: Unguarded Fixed-points

Last ingredient of a branching theory is the **fixed-point operator** $\mathsf{fp}\ x : S^*(\{x\} + Y) \to S^*(Y)$

$$T \vdash \mathsf{fp}\ x\ t(x, \vec{y}) = t(\ \mathsf{fp}\ x\ t(x, \vec{y})\ , \vec{y})$$

We obtain an operator on M that performs a type of iteration determined by $\mathsf{fp}\ x$

$$\begin{array}{ccc} S^*(\{x\}+Y)/{=_T} & \xrightarrow{\mathsf{fp}\ x} & S^*(Y)/{=_T} \\ \cong \big\uparrow & & \big\downarrow \cong \\ M(\{x\}+Y) & \overset{\mathsf{fp}\ x}{\dashrightarrow} & M(Y) \end{array}$$

Star Fragment Semantics: Unguarded Fixed-points

Last ingredient of a branching theory is the **fixed-point operator** fp $x\colon S^*(\{x\} + Y) \to S^*(Y)$

$$T \vdash \ \mathsf{fp}\ x\ t(x, \vec{y}) = t(\ \mathsf{fp}\ x\ t(x, \vec{y})\ , \vec{y})$$

We obtain an operator on M that performs a type of iteration determined by fp x

$$\begin{array}{ccc} S^*(\{x\}+Y)/{=_T} & \xrightarrow{\ \mathsf{fp}\ x\ } & S^*(Y)/{=_T} \\ \cong \uparrow & & \downarrow \cong \\ M(\{x\}+Y) & \overset{\mathsf{fp}\ x}{\dashrightarrow} & M(Y) \end{array}$$

Example. The operator fp $x\ t(x, \vec{y}) = t(0, \vec{y})$ on semilattice terms is a fixed-point operator:

Star Fragment Semantics: Unguarded Fixed-points

Last ingredient of a branching theory is the **fixed-point operator** $\mathsf{fp}\ x\colon S^*(\{x\} + Y) \to S^*(Y)$

$$T \vdash \mathsf{fp}\ x\ t(x, \vec{y}) = t(\mathsf{fp}\ x\ t(x, \vec{y}), \vec{y})$$

We obtain an operator on M that performs a type of iteration determined by $\mathsf{fp}\ x$

$$\begin{array}{ccc} S^*(\{x\}+Y)/{=_T} & \xrightarrow{\mathsf{fp}\ x} & S^*(Y)/{=_T} \\ \cong\big\uparrow & & \big\downarrow\cong \\ M(\{x\}+Y) & \dashrightarrow^{\mathsf{fp}\ x} & M(Y) \end{array}$$

Example. The operator $\mathsf{fp}\ x\ t(x, \vec{y}) = t(0, \vec{y})$ on semilattice terms is a fixed-point operator:

$$T_{\mathrm{SL}} \vdash \mathsf{fp}\ x\ (x + y) = 0 + y = y = y + y = (\mathsf{fp}\ x\ (x + y)) + y$$

Star Fragment Semantics: Unguarded Fixed-points

Last ingredient of a branching theory is the **fixed-point operator** fp $x\colon S^*(\{x\} + Y) \to S^*(Y)$

$$T \vdash \text{fp } x\ t(x, \vec{y}) = t(\text{fp } x\ t(x, \vec{y})\ , \vec{y})$$

We obtain an operator on M that performs a type of iteration determined by fp x

$$\begin{array}{ccc} S^*(\{x\} + Y)/{=_T} & \xrightarrow{\text{fp } x} & S^*(Y)/{=_T} \\ \cong\uparrow & & \downarrow\cong \\ M(\{x\} + Y) & \overset{\text{fp } x}{\dashrightarrow} & M(Y) \end{array}$$

Example. The operator fp $x\ t(x, \vec{y}) = t(0, \vec{y})$ on semilattice terms is a fixed-point operator:

$$T_{\mathrm{SL}} \vdash \text{fp } x\ (x + y) = 0 + y = y = y + y = (\text{fp } x\ (x + y)) + y$$

Given $U \subseteq \{x\} + Y$, this corresponds to

Star Fragment Semantics: Unguarded Fixed-points

Last ingredient of a branching theory is the **fixed-point operator** $\mathsf{fp}\ x\colon S^*(\{x\} + Y) \to S^*(Y)$

$$T \vdash \ \mathsf{fp}\ x\ t(x, \vec{y}) = t(\ \mathsf{fp}\ x\ t(x, \vec{y})\ , \vec{y})$$

We obtain an operator on M that performs a type of iteration determined by $\mathsf{fp}\ x$

$$\begin{array}{ccc} S^*(\{x\} + Y)/{=_T} & \xrightarrow{\mathsf{fp}\ x} & S^*(Y)/{=_T} \\ \cong \uparrow & & \downarrow \cong \\ M(\{x\} + Y) & \overset{\mathsf{fp}\ x}{\dashrightarrow} & M(Y) \end{array}$$

Example. The operator $\mathsf{fp}\ x\ t(x, \vec{y}) = t(0, \vec{y})$ on semilattice terms is a fixed-point operator:

$$T_{\mathrm{SL}} \vdash \mathsf{fp}\ x\ (x + y) = 0 + y = y = y + y = (\mathsf{fp}\ x\ (x + y)) + y$$

Given $U \subseteq \{x\} + Y$, this corresponds to

$$\mathsf{fp}\ x\ (U) = U - \{x\}$$

Star Fragment Semantics

Operational semantics is given by a map

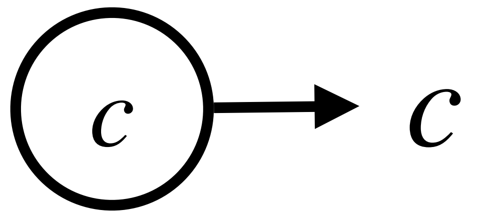
$$\ell : \mathsf{StExp} \longrightarrow M(\top + Act \times \mathsf{StExp})$$

Star Fragment Semantics

Operational semantics is given by a map

$$\ell : \mathsf{StExp} \longrightarrow M(\top + Act \times \mathsf{StExp})$$

$\ell(c) = c$

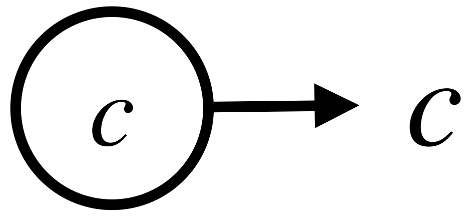


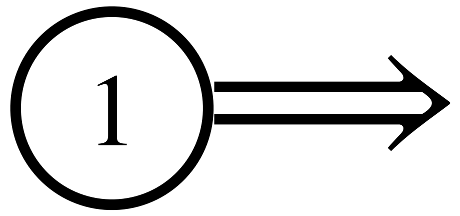
Star Fragment Semantics

Operational semantics is given by a map

$$\ell : \mathsf{StExp} \longrightarrow M(\top + Act \times \mathsf{StExp})$$

$\ell(c) = c$ $\qquad\qquad$ $\ell(1) = \top$



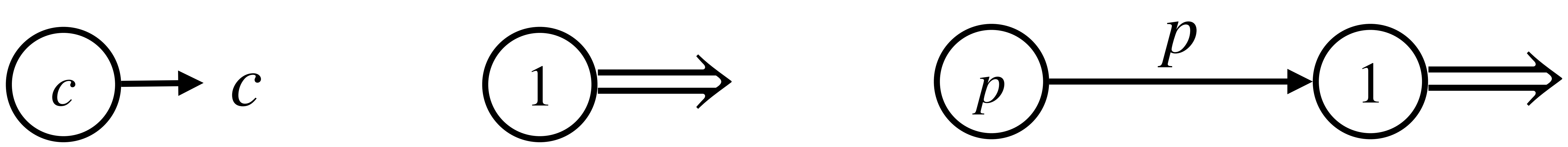


Star Fragment Semantics

Operational semantics is given by a map

$$\ell : \mathsf{StExp} \longrightarrow M(\top + Act \times \mathsf{StExp})$$

$\ell(c) = c$ $\qquad \ell(1) = \top$ $\qquad \ell(p) = (p, 1)$

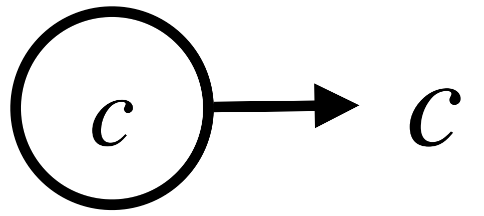


Star Fragment Semantics

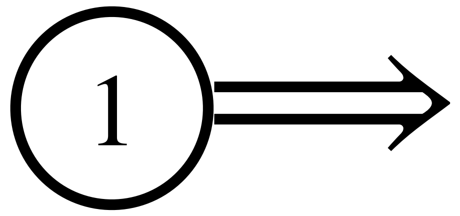
Operational semantics is given by a map

$$\ell : \mathsf{StExp} \longrightarrow M(\top + Act \times \mathsf{StExp})$$

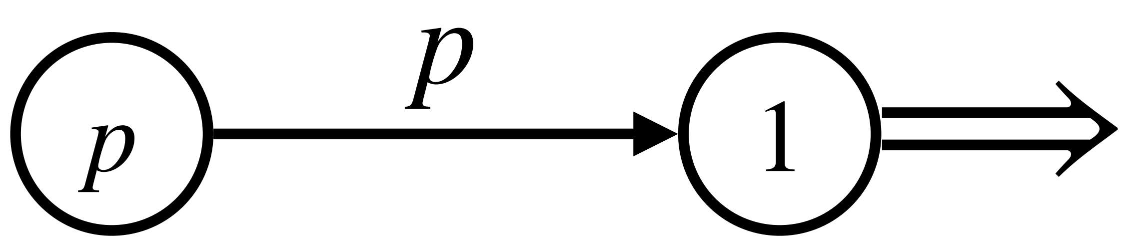
$\ell(c) = c$



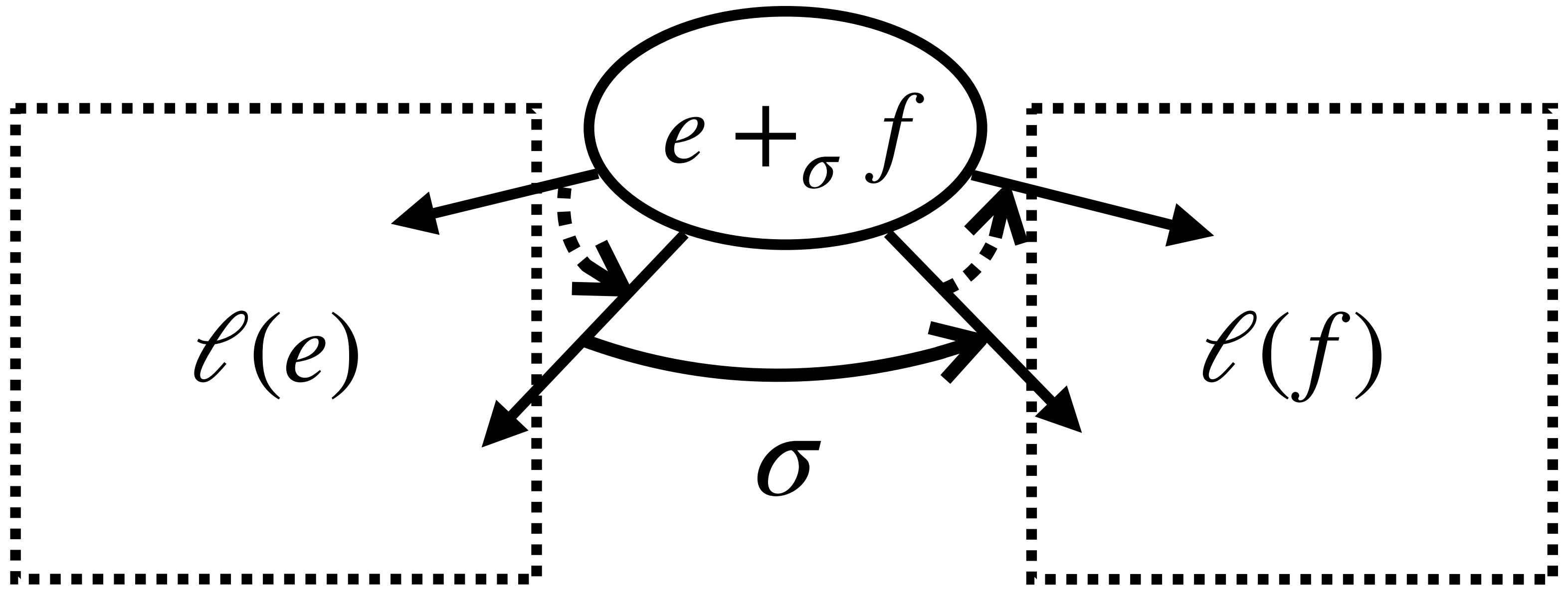
$\ell(1) = \top$



$\ell(p) = (p, 1)$



$\ell(e +_\sigma f) = \sigma(\ell(e), \ell(f))$

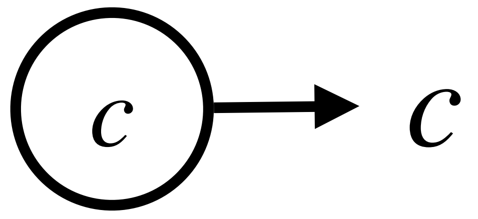


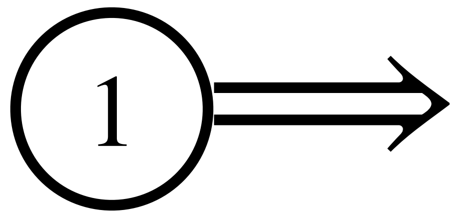
Star Fragment Semantics

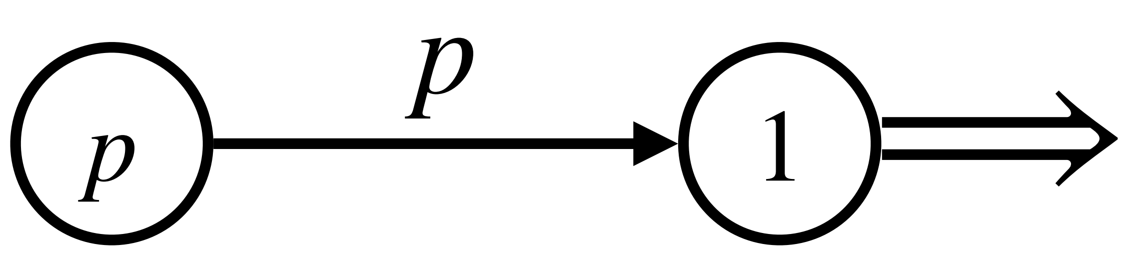
Operational semantics is given by a map

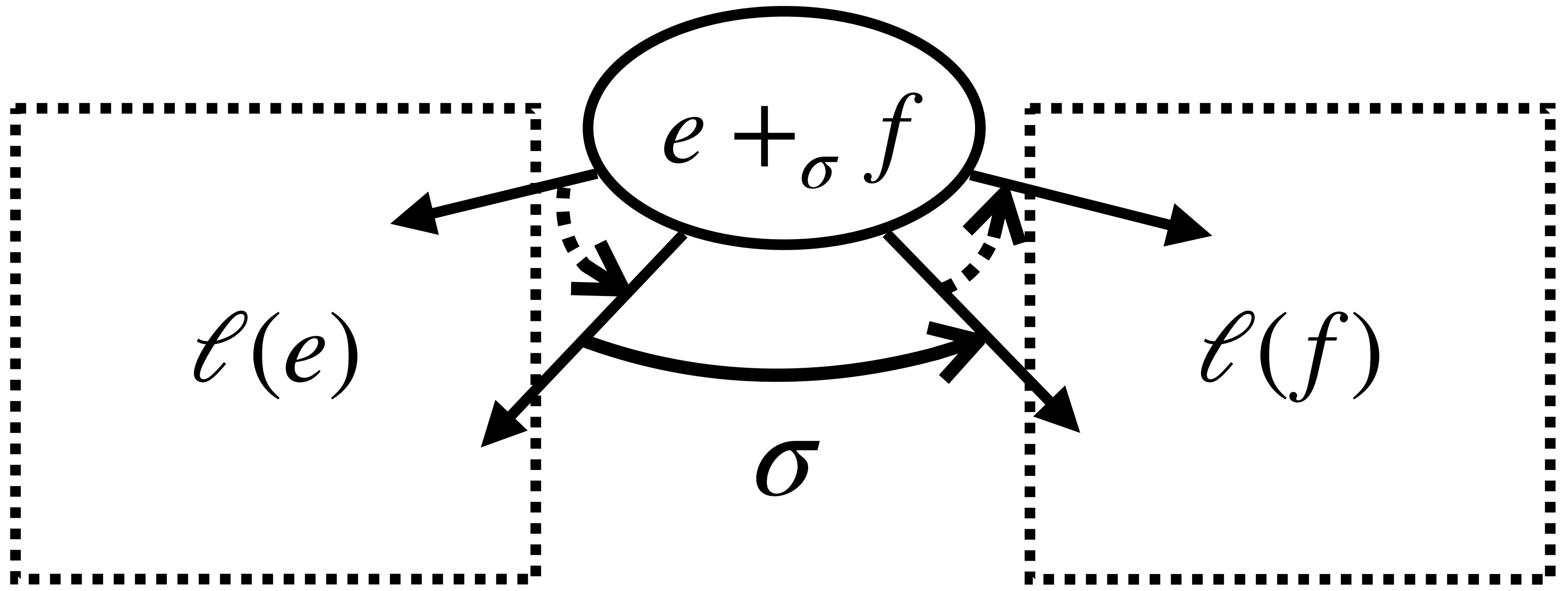
$$\ell : \mathsf{StExp} \longrightarrow M(\top + Act \times \mathsf{StExp})$$

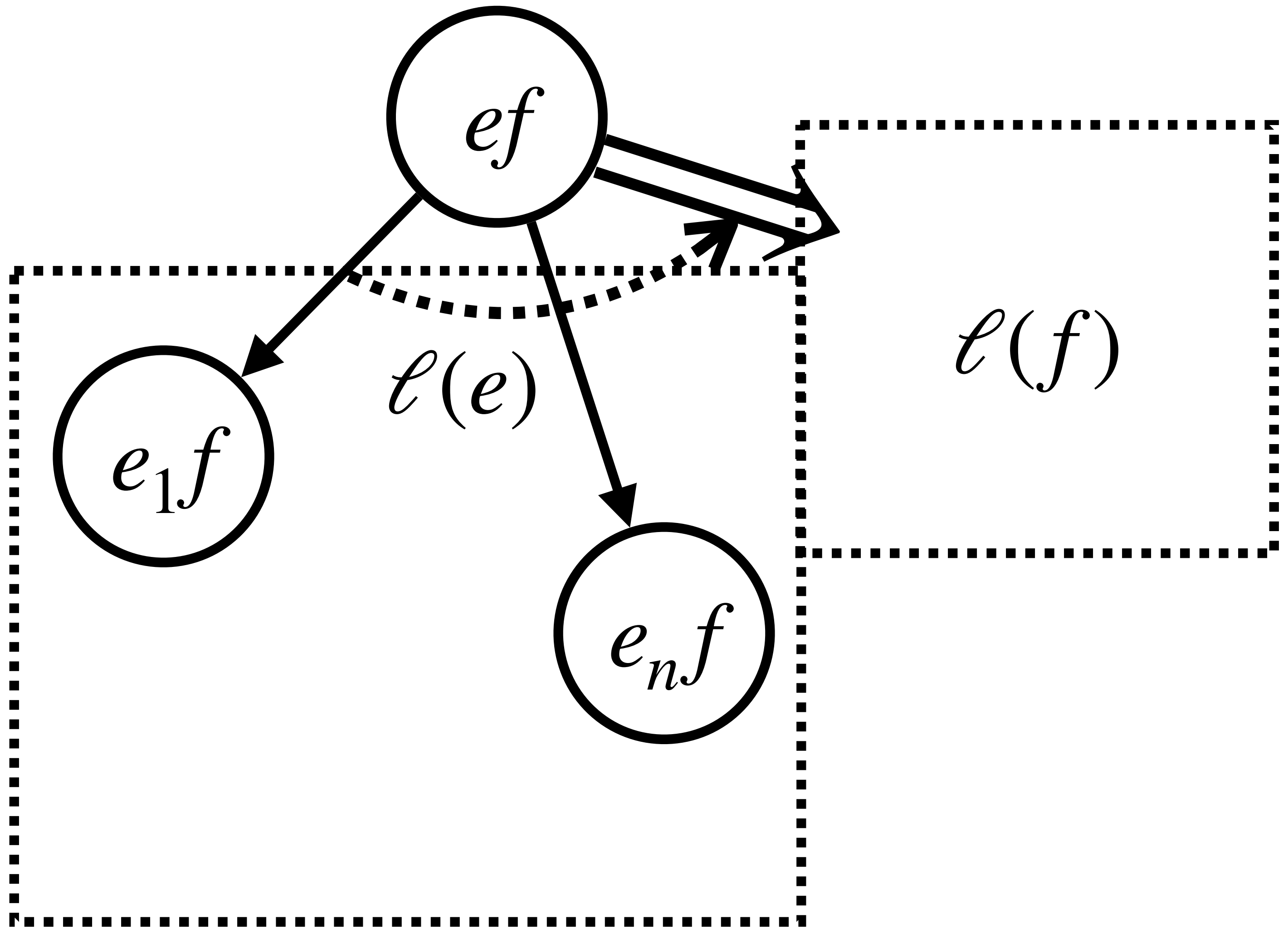
$\ell(c) = c$ $\quad$ $\ell(1) = \top$ $\quad$ $\ell(p) = (p, 1)$ $\quad$ $\ell(e +_\sigma f) = \sigma(\ell(e), \ell(f))$











If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_n, e_n))$, then

$$\ell(ef) = t(\ell(f), (p_1, e_1 f), \ldots, (p_n, e_n f))$$

Star Fragment Semantics

If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_1, e_1))$, then

$$\ell(e^{(\sigma)}) = \text{fp } x \ \sigma(t(x, (p_1, e_1 e^{(\sigma)}), \ldots, (p_1, e_1 e^{(\sigma)})), \top)$$

Star Fragment Semantics

If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_1, e_1))$, then

$$\ell(e^{(\sigma)}) = \mathsf{fp}\ x\ \sigma(t(x, (p_1, e_1 e^{(\sigma)}), \ldots, (p_1, e_1 e^{(\sigma)})), \top)$$

Star Fragment Semantics

If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_1, e_1))$, then

$$\ell(e^{(\sigma)}) = \text{fp } x \ \sigma(t(x, (p_1, e_1 e^{(\sigma)}), \ldots, (p_1, e_1 e^{(\sigma)})), \top\ )$$

Example. For regular expressions, if $p \in Act$, then

Star Fragment Semantics

If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_1, e_1))$, then

$$\ell(e^{(\sigma)}) = \text{fp } x \ \sigma(t(x, (p_1, e_1 e^{(\sigma)}), \ldots, (p_1, e_1 e^{(\sigma)})), \top\ )$$

Example. For regular expressions, if $p \in Act$, then

Star Fragment Semantics

If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_1, e_1))$, then

$$\ell(e^{(\sigma)}) = \mathsf{fp}\ x\ \sigma(t(x, (p_1, e_1 e^{(\sigma)}), \ldots, (p_1, e_1 e^{(\sigma)})), \top\ )$$

Example. For regular expressions, if $p \in Act$, then

$$\ell(1 + p) = \{\ \top\ , (p, 1)\}$$

Star Fragment Semantics

If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_1, e_1))$, then

$$\ell(e^{(\sigma)}) = \mathsf{fp}\ x\ \sigma(t(x, (p_1, e_1 e^{(\sigma)}), \ldots, (p_1, e_1 e^{(\sigma)})), \top\ )$$

Example. For regular expressions, if $p \in Act$, then

$\ell(1 + p) = \{\ \top\ , (p,1)\}$

Star Fragment Semantics

If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_1, e_1))$, then

$$\ell(e^{(\sigma)}) = \text{fp } x \ \sigma(t(x, (p_1, e_1 e^{(\sigma)}), \ldots, (p_1, e_1 e^{(\sigma)})), \top)$$

Example. For regular expressions, if $p \in Act$, then

$$\ell(1 + p) = \{ \top, (p,1)\}$$

$$\ell((1 + p)^*) = \text{fp } x \ \{x, (p, 1e^{(\sigma)})\} \cup \{ \top \}$$

Star Fragment Semantics

If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_1, e_1))$, then

$$\ell(e^{(\sigma)}) = \text{fp } x \ \sigma(t(x, (p_1, e_1 e^{(\sigma)}), \ldots, (p_1, e_1 e^{(\sigma)})), \top \ )$$

Example. For regular expressions, if $p \in Act$, then

$$\ell(1 + p) = \{ \ \top \ , (p, 1)\}$$

$$\ell((1 + p)^*) = \text{fp } x \ \{x, (p, 1e^{(\sigma)})\} \cup \{ \ \top \ \}$$
$$= \{(p, 1e^{(\sigma)}), \ \top \ \}$$

Star Fragment Semantics

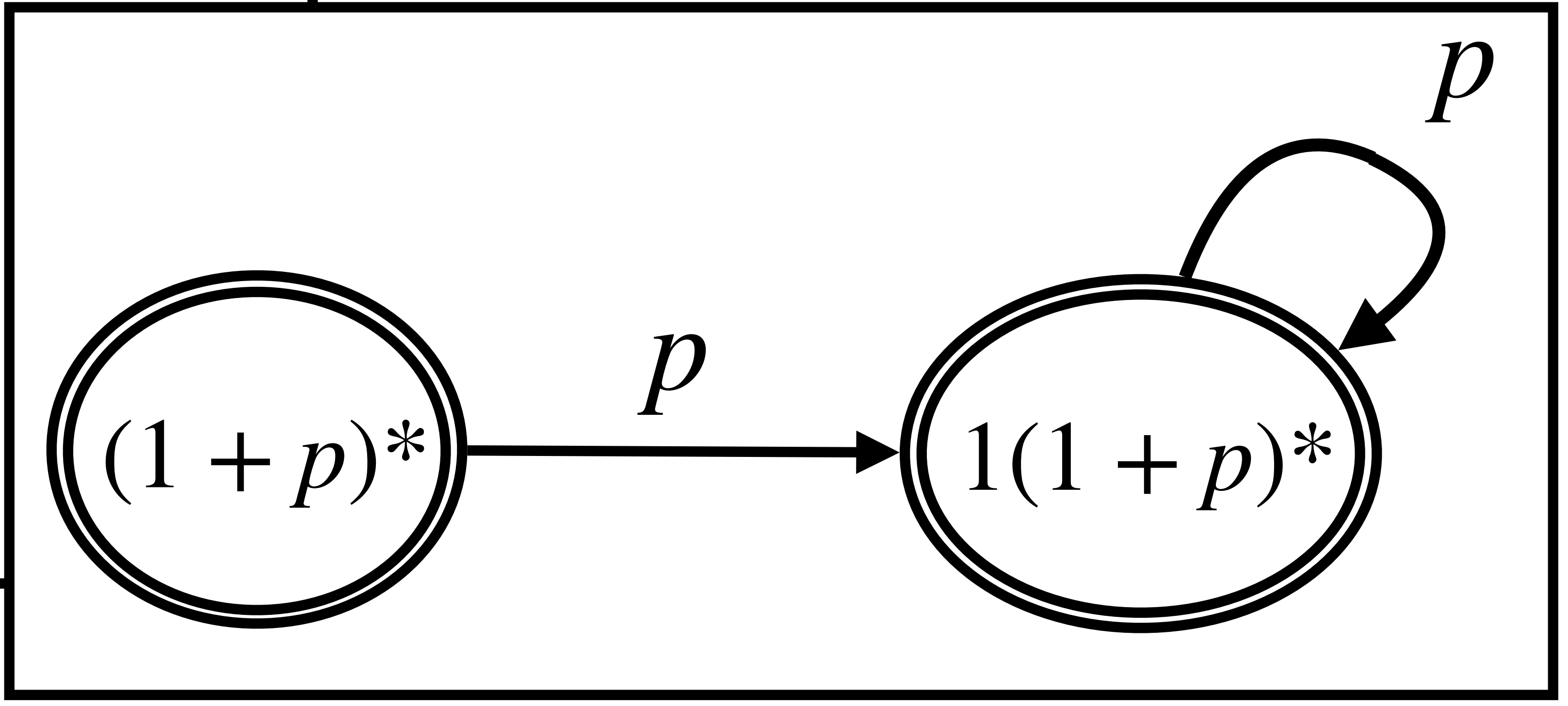
If $\ell(e) = t(\top, (p_1, e_1), \ldots, (p_1, e_1))$, then

$$\ell(e^{(\sigma)}) = \text{fp } x \ \sigma(t(x, (p_1, e_1 e^{(\sigma)}), \ldots, (p_1, e_1 e^{(\sigma)})), \top\ )$$

Example. For regular expressions, if $p \in Act$, then

$\ell(1 + p) = \{\ \top\ , (p, 1)\}$

$\ell((1 + p)^*) = \text{fp } x\ \{x, (p, 1e^{(\sigma)})\} \cup \{\ \top\ \}$
$= \{(p, 1e^{(\sigma)}), \top\ \}$



An Axiomatization of Star Fragments modulo Bisimilarity?

$$T$$

Equational Branching Axioms

$$ce = c$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e +_\sigma f)g = eg +_\sigma fg$$

Sequencing Axioms

General Unguarded Fixed-point Axiom

$$t(1, \vec{g})^{(\sigma)} = \mathsf{fp}\ x\ (t(x, \vec{g}t(1, \vec{g})^{(\sigma)}) +_\sigma 1)$$

(Above, $\vec{g} = (g_1, \ldots, g_n)$ are guarded)

$$e^{(\sigma)} = ee^{(\sigma)} +_\sigma 1$$

$$\frac{g = eg +_\sigma f \quad e \text{ guarded}}{g = e^{(\sigma)} f}$$

Unique Guarded Fixed-point Axioms

An Axiomatization of Star Fragments modulo Bisimilarity?

$$T$$

Equational Branching Axioms

$$ce = c$$
$$1e = e$$
$$e = e1$$
$$e(fg) = (ef)g$$
$$(e +_\sigma f)g = eg +_\sigma fg$$

Sequencing Axioms

General Unguarded Fixed-point Axiom

$$t(1, \vec{g})^{(\sigma)} = \mathsf{fp}\ x\ (t(x, \vec{g}t(1, \vec{g})^{(\sigma)}) +_\sigma 1)$$

(Above, $\vec{g} = (g_1, \ldots, g_n)$ are guarded)

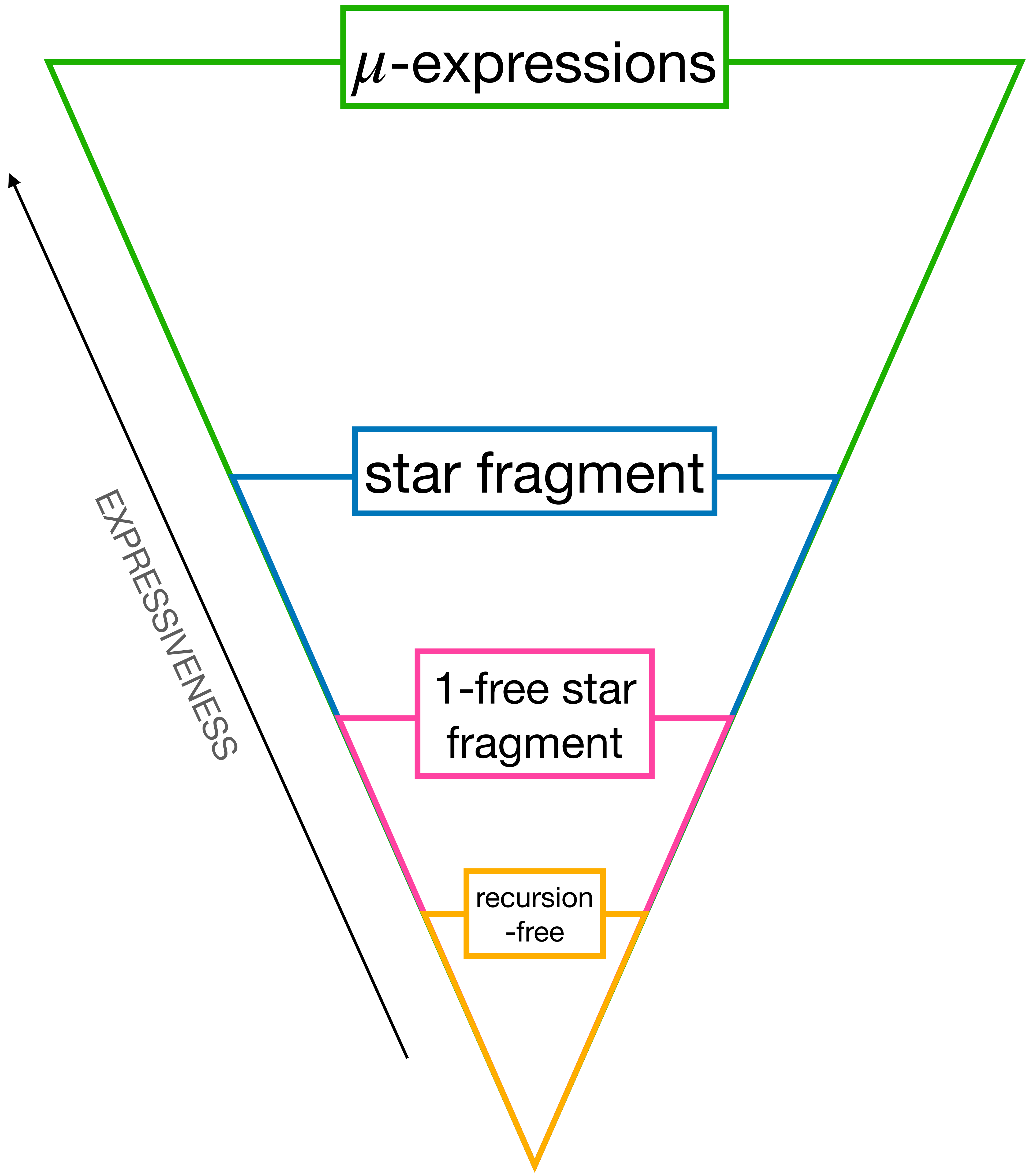
$$e^{(\sigma)} = ee^{(\sigma)} +_\sigma 1$$

$$\frac{g = eg +_\sigma f \quad e \text{ guarded}}{g = e^{(\sigma)} f}$$

Unique Guarded Fixed-point Axioms

Generalized Milner's Completeness Problem:
Is this axiomatization of bisimulation complete for *every* star fragment?

Known & Unknown Completeness Theorems



	Regex mod bisimilarity	GKAT mod bisimilarity	ProbRegex mod bisim.	ProbGKAT mod bisim.
μ -exp	complete	complete	complete	complete
star fragment	complete (Grabmayer, 2022)	Unkown	Unkown	Unknown
1-free star fragment	complete (Grabmayer, Fokkink, 2019)	complete (Kappé, S., Silva, 2023)	complete (unpublished)	Unknown
recursion-free	complete	complete	complete	complete

Summary

- Star fragments arise from *branching theories*, (S, T, fp) consisting of an algebraic theory and a fixed-point operator that determines behaviour of unguarded fixed-points
- Milner's regular expressions mod bisimilarity = *semilattices with bottom* star fragment
- GKAT/bisimilarity = **if-then-else** with **crash** star fragment
- Further examples:
  - (Rozowski, Kappé, Kozen, Schmid, Silva, 2023) ProbGKAT mod bisimilarity = GKAT + $\bigoplus_p$
  - Probabilistic regular expressions mod bisimilarity = $\bigoplus_p$ instead of $+$
  - Regex mixing nondeterminism and probability = Regular expressions + $\bigoplus_p$

Generalized Milner's Completeness Problem: Is this axiomatization of bisimulation complete for *every* star fragment?

$$T$$

Equational Branching Axioms

$$ce = c \qquad 1e = e \qquad e = e1 \qquad e(fg) = (ef)g \qquad (e +_\sigma f)g = eg +_\sigma fg$$

Sequencing Axioms

General Unguarded Fixed-point Axiom

$$t(1, \vec{g})^{(\sigma)} = \mathsf{fp}\ x\ (t(x, \vec{g}t(1, \vec{g})^{(\sigma)}) +_\sigma 1)$$

(Above, $\vec{g} = (g_1, \ldots, g_n)$ are guarded)

$$e^{(\sigma)} = ee^{(\sigma)} +_\sigma 1$$

$$\frac{g = eg +_\sigma f \quad e \text{ guarded}}{g = e^{(\sigma)} f}$$

Unique Guarded Fixed-point Axioms