State Complexity of Simple Splicing

Lila Kari and Timothy Ng
School of Computer Science, University of Waterloo
DCFS 2019, Košice, Slovakia

ATCCTCACGCCGTAG

CATTTCCGTGTATGA

ACG	CCG
TTC	CGT

ATCCTCACGCGTGTATGA

Splicing systems

T. Head (1987)

A **splicing system** is a 3-tuple $H = (\Sigma, R, L)$ where

- Σ is a finite alphabet
- R is a set of rules
- L is the initial language

A **splicing rule** is a 4-tuple $r = (u_1,\ u_2;\ u_3,\ u_4)$ such that if $x = x_1 u_1 u_2 x_2$ and $y = y_1 u_3 u_4 y_2$, then

$$x \vdash_r y = x_1 u_1 u_4 y_2$$

The **language** of a splicing system $H = (\Sigma, R, L)$ is $R^*(L)$ where

$$R(L) = \{w \in \Sigma \mid (\exists\ x, y \in L,\ r \in R) \text{ such that } x \vdash_r y\}$$

and

- $R^0(L) = L$
- $R^i(L) = R^{\mathrm{i}}(L) \cup R(R^{i-1}(L))$
- $R^*(L) = \bigcup_{i \geq 0} R^i(L)$

Complexity of splicing systems

	Finite Ruleset	Regular Ruleset
Finite Initial Language	**Locally testable** (T. Head, 1987)	**Recursively enumerable** (T. Head, Gh. Păun, D. Pixton, 1997)
Regular Initial Language	**Regular** (K. Culik II and T. Harju, 1991)	**Recurisvely enumerable** (Gh. Păun, 1996)

ACGTACGTATAC
CATACTTGCTTC

$$\begin{array}{c|c} T & \\ \hline T & \end{array}$$

ACGTACTTGCTTC

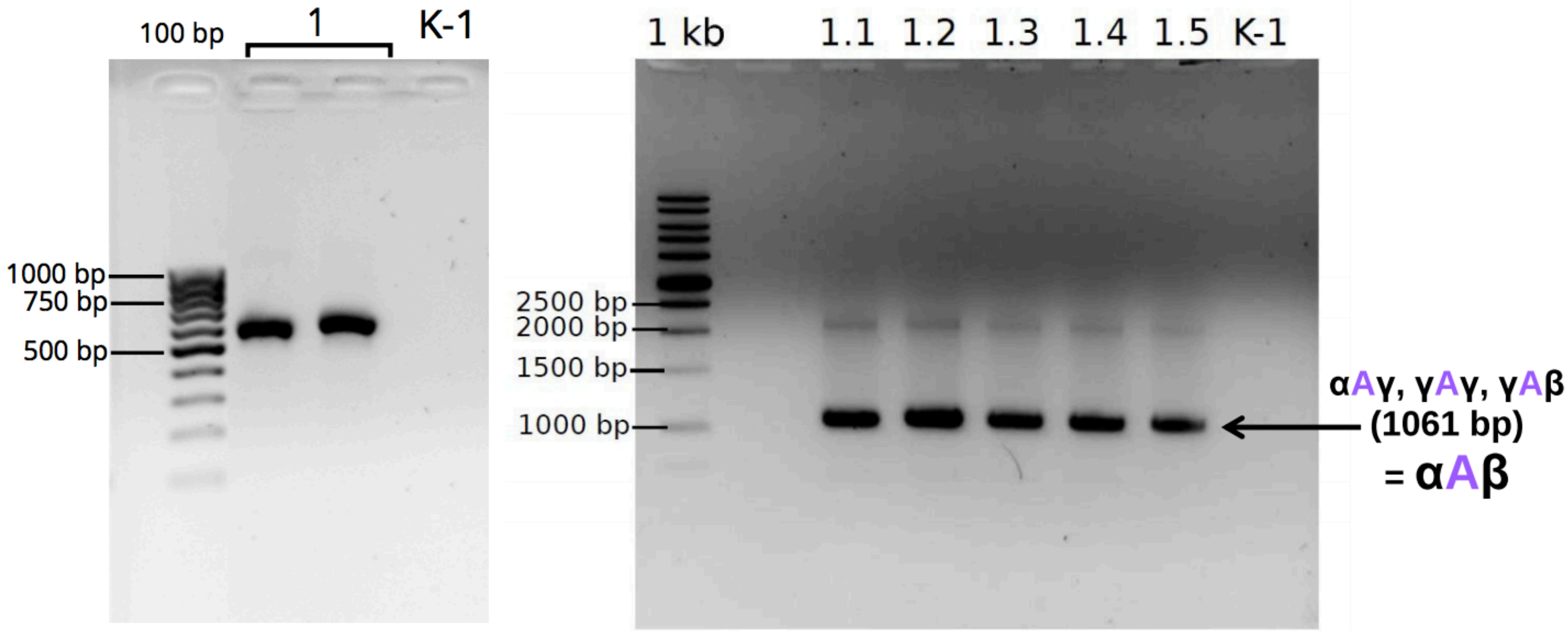
Simple splicing

A. Mateescu, Gh. Păun, G. Rozenberg, A. Salomaa (1998)

A splicing rule r is **simple** if $r = (u_1, ε; u_3, ε)$ where $u_1 = u_3$ and $|u_1| = 1$.

A splicing system with only simple rules is a **simple splicing system**.

A simple splicing system is denoted by $H = (\Sigma, M, L)$ where $M \subseteq \Sigma$. Then $a \in M$ means $(a, ε; a, ε)$ is a rule in H.



Franco, Bellamoli, Lampis (2017)

Word Blending

S.K. Enaganti, L. Kari, T. N., Z. Wang (2018)

ACGTACGTATAC
CATACTTGCTTC

ACGTACTTGCTTC

$$
\begin{aligned}
u \bowtie v = \{\alpha w \beta \mid & u = \alpha w \gamma_1, v = \gamma_2 w \beta, \\
& \alpha, \beta, w, \gamma_1, \gamma_2 \in \Sigma^*\}
\end{aligned}
$$

ACGTACGTATAC
CATACTTGCTTC

ACGTACTTGCTTC

$$
\begin{aligned}
u \bowtie v = \{\alpha a \beta \mid & u = \alpha a \gamma_1, v = \gamma_2 a \beta, \\
& \alpha, \beta, \gamma_1, \gamma_2 \in \Sigma^*, a \in \Sigma\}
\end{aligned}
$$

Descriptional complexity measures for splicing systems

- Radius (Gh. Păun, 1996)
- Size of initial language (A. Mateescu et al., 1998)
- Size of grammar (A. Mateescu et al., 1998)
- Number/length of rules (R. Loos et al., 2007)
- Size of nondeterministic finite automaton (R. Loos et al., 2007)

The **state complexity of a regular language** is the number of states in its minimal deterministic finite automaton.

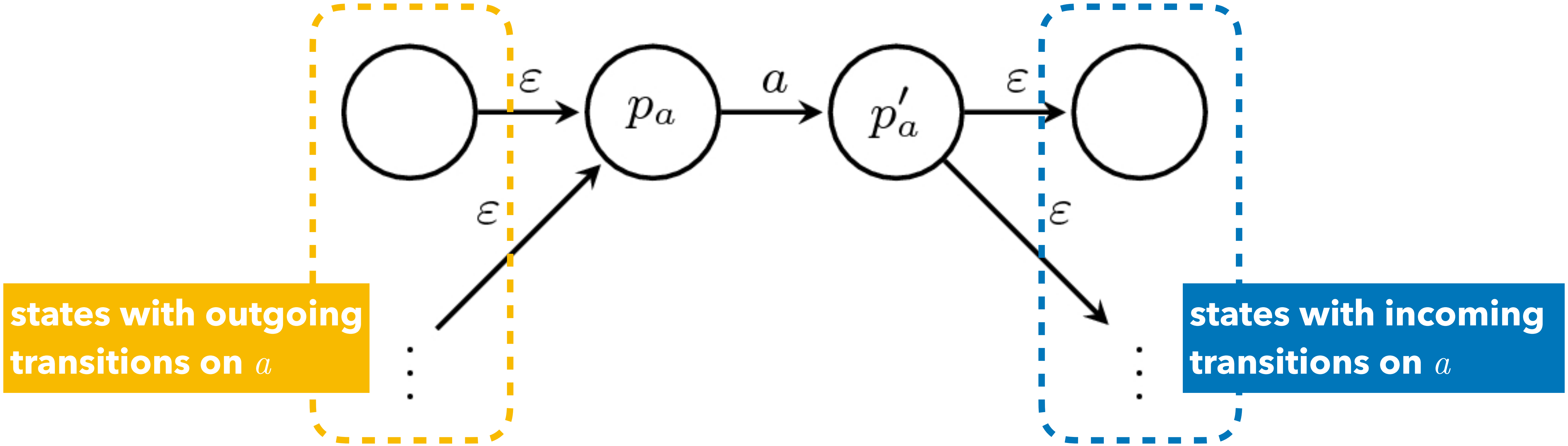
The **state complexity of an operation** is the worst-case state complexity of the language resulting from the operation, as a function of the state complexity of the operands.

For a **simple** splicing system with initial language L with state complexity n

$$\begin{cases} 2^n - 1 & \text{if } L \text{ is regular,} \\ 2^{n-2} + 1 & \text{if } L \text{ is finite,} \end{cases}$$

Let $H = (\Sigma, M, L)$ be a simple splicing system and let A **be the DFA for** L. From A, we will **construct an NFA** that recognizes the language of H.

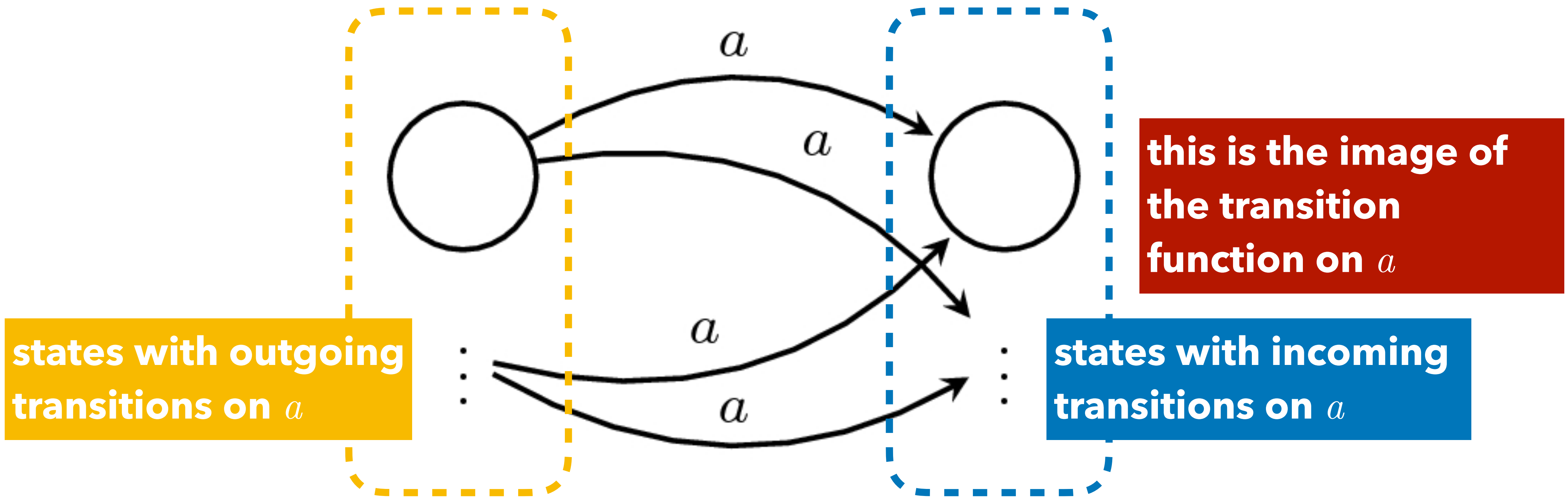
For each symbol $a \in M$, construct a **bridge**:



Add ε-transitions:

- from each state in A with outgoing transitions on a to p_a, and
- from p'_a to all states of A with incoming transitions on a

Then perform **ε-transition removal**:

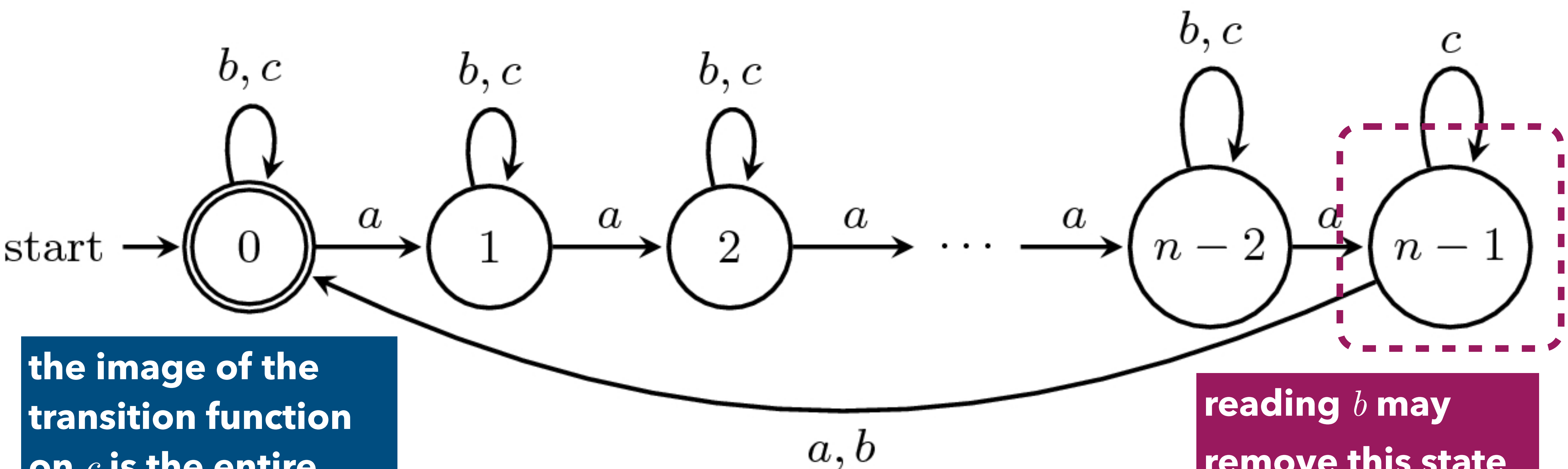


The new states p_a and p'_a disappear and collapse into transitions between states of A on a.

Since we begin with an n-state DFA, this process results in an **n-state NFA** after removing ε-transitions.

This gives an **upper bound of $2^n - 1$** reachable (non-empty subsets) states via the subset construction.

The upper bound is reachable via the simple splicing system $(\{a,b,c\},\{c\},L_n)$, where L_n is recognized by the following DFA



the image of the transition function on c is the entire state set

reading b may remove this state

The upper bound is lower with a finite initial language

- Consider a simple splicing system (Σ, M, I), where I **is a finite language with state complexity** n
- Since I is finite, its DFA A, is **acyclic**. Then the initial state q_0 of A has no incoming transitions so **the only reachable subset containing** q_0 **is** $\{q_0\}$.
- Since I is finite, A must contain a **sink state**.
- This gives a total of $2^{n-2} - 1 + 2$ **states**.

For a simple splicing system with initial **finite** language L with state complexity n **over** k **symbols**

$$\begin{cases} 2 & \text{if } k = 1, \\ 2n-3 & \text{if } k = 2, \\ 2^{\frac{n}{2}} + 1 & \text{if } k = 3 \text{ and } n \text{ is even}, \\ 3 \cdot 2^{\frac{n-3}{2}} + 1 & \text{if } k = 3 \text{ and } n \text{ is odd}, \\ 2^{n-2} + 1 & \text{if } k \geq 2^{n-3}. \end{cases}$$

Lemma. If $a \in M$, then $\mathrm{im}\ \delta_a{}'$ contains exactly the sink state and $\mathrm{im}\ \delta_a$.

In other words, if $a \in M$, then reading a will take the DFA to either exactly one subset or the sink state.

To reach the upper bound for finite initial languages

- Let q_1 be a state reachable **directly by the initial state** and no other state; this state must exist since the DFA is acyclic.

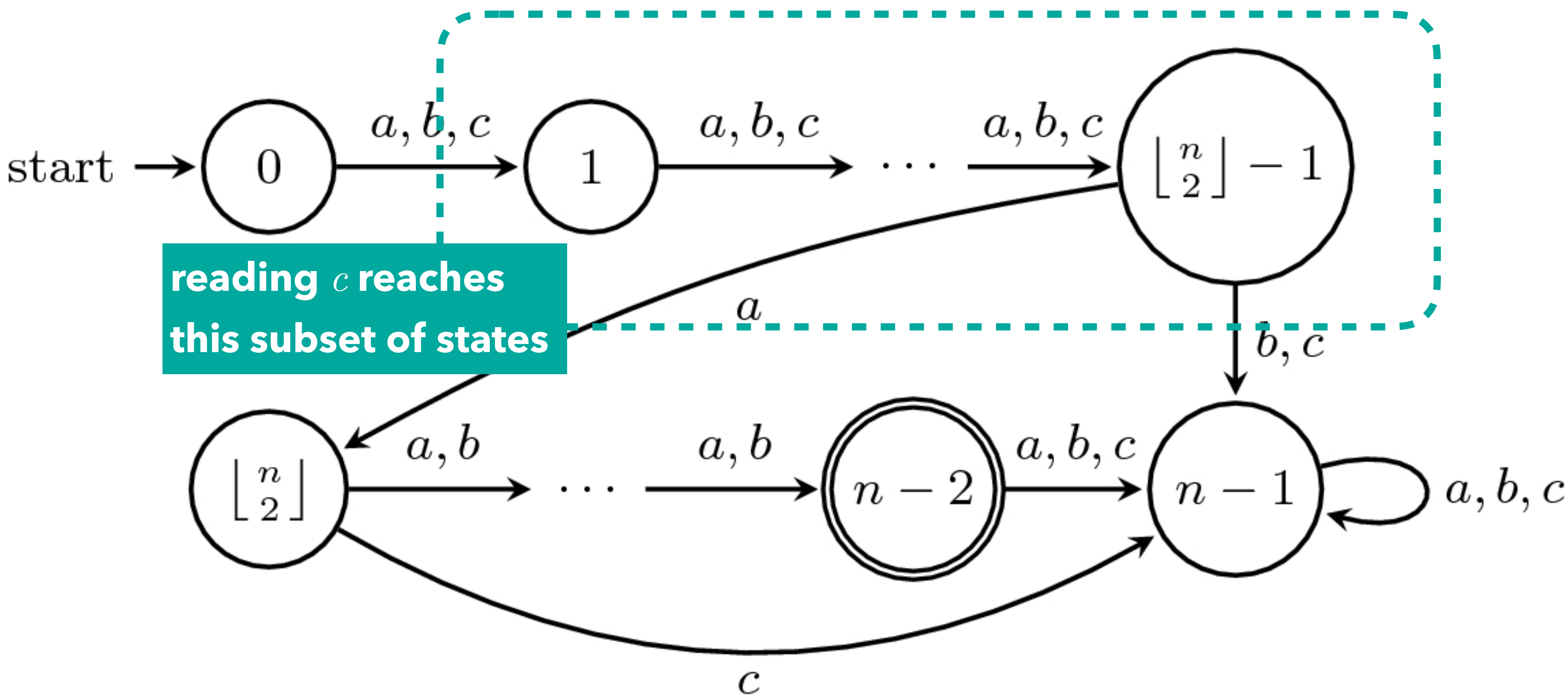
- If $q_1 \in S$, then S is reachable only if it is **the image of δ_a for some $a \in M$**.

- Since there are up to 2^{n-3} subsets that contain q_1, to reach each of these subsets, **there must be one $a \in M$ for each**.

If $k = 2$

- If a, b are not in M, then we just **have L**.
- If a, b are both in M, then we have **at most two** reachable subsets.
- If $a \in M$ and b is not, then to maximize the number of reachable states, we must have $\delta_b(i) = i+1$ and $|\mathrm{im}\ \delta_a| = 2$. This gives us **at most $2n-3$ states**.

For $k = 3$, the upper bound is reached by $(\{a,b,c\},\{c\},I_n)$ where I_n is recognized by the DFA below.



A splicing rule r is **semi-simple** if $r = (u_1, \varepsilon; u_3, \varepsilon)$ with $|u_1| = |u_3| = 1$.

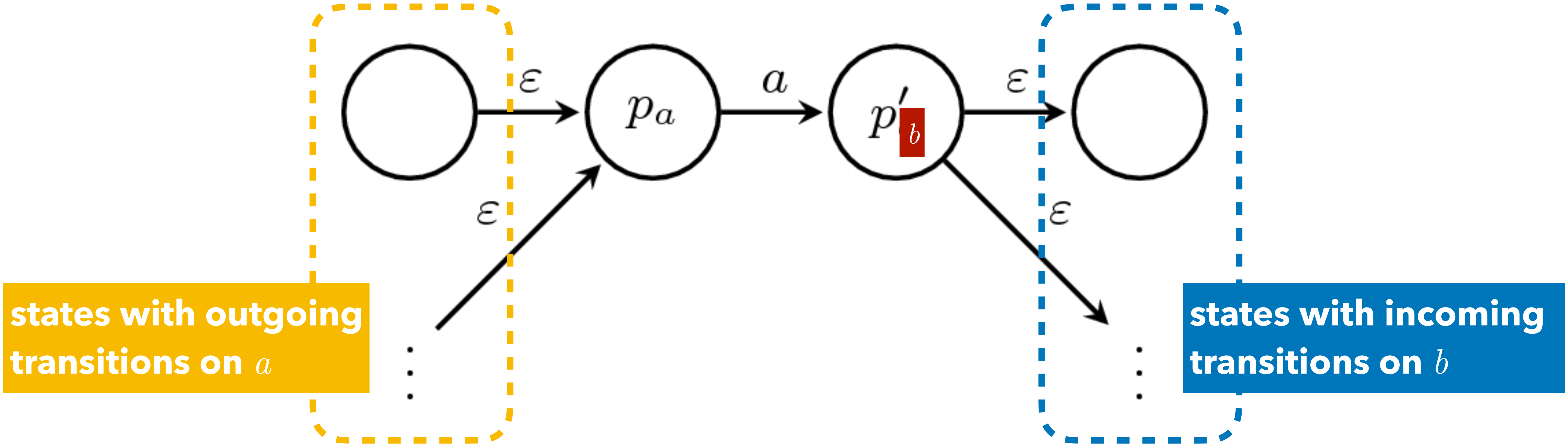
A splicing system with only simple rules is a **semi-simple splicing system**.

A semi-simple splicing system is denoted by $H = (\Sigma, M^{(2)}, L)$ where $M^{(2)} \subseteq \Sigma \times \Sigma$. Then $(a,b) \in M^{(2)}$ means $(a, \varepsilon; b, \varepsilon)$ is a rule in H.

Semi-simple splicing

E. Goode and D. Pixton (2001)

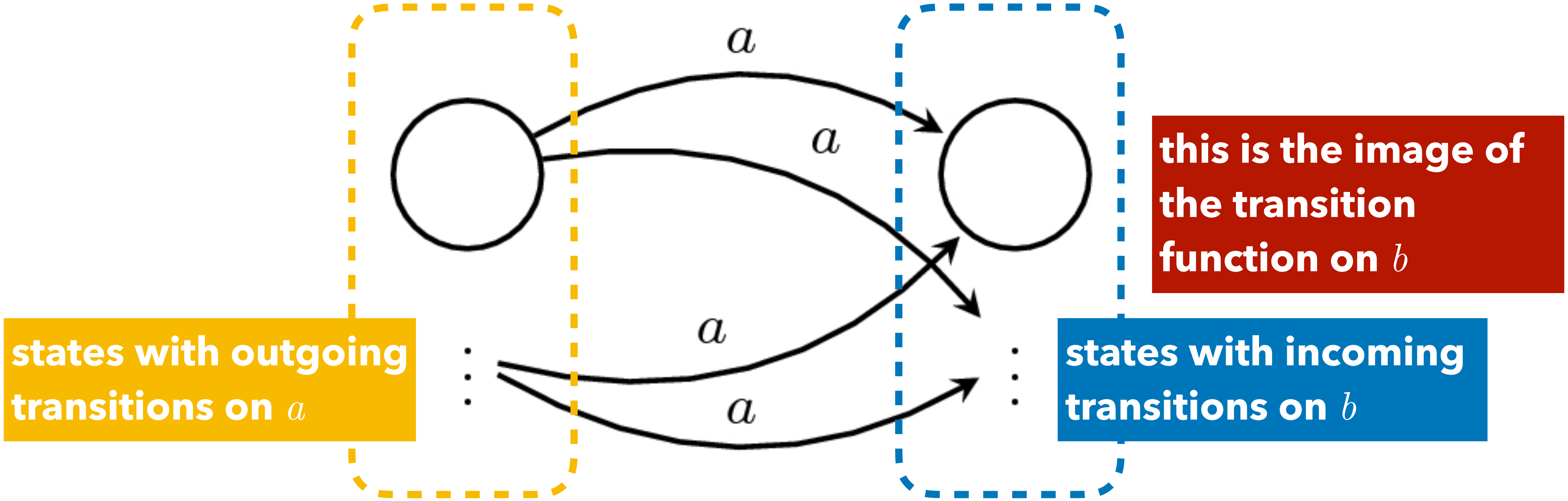
For each pair $(a, b) \in M^{(2)}$, construct a **bridge**:



Add ε-transitions:

- from each state in A with outgoing transitions on a to p_a, and
- from p'_b to all states of A with incoming transitions on b

Then perform **ε-transition removal**:

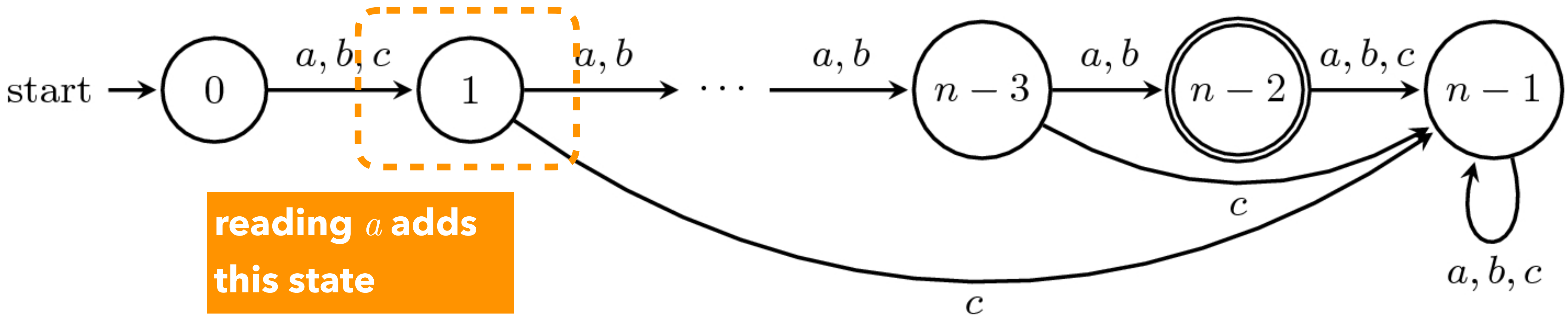


The new states p_a and p'_b disappear and collapse into transitions between states of A on a.

This construction shows that **semi-simple** splicing systems with regular and finite initial languages have the **same upper bound** for state complexity.

For semi-simple splicing systems with a regular initial language, this upper bound is reached by the **same lower bound witness** for simple splicing systems.

The upper bound for semi-simple splicing systems with a **finite initial language** can be reached via $(\{a,b,c\},\{(a,c)\},I_n)$, where I_n is recognized by the following DFA



For $M \subseteq \Sigma \times \Sigma$ we define the operation on two strings u,v by

$$u \diamond_M v = u'av'$$

if $u = u'a$ and $v = bv'$ for $(a,b) \in M$ and u',$v' \in \Sigma$; and is undefined otherwise.

The **crossover operation** can be defined in terms of this operation extended to languages by

$$L_1 \sharp_M L_2 = \mathrm{pref}(L_1) \diamond_M \mathrm{suff}(L_2)$$

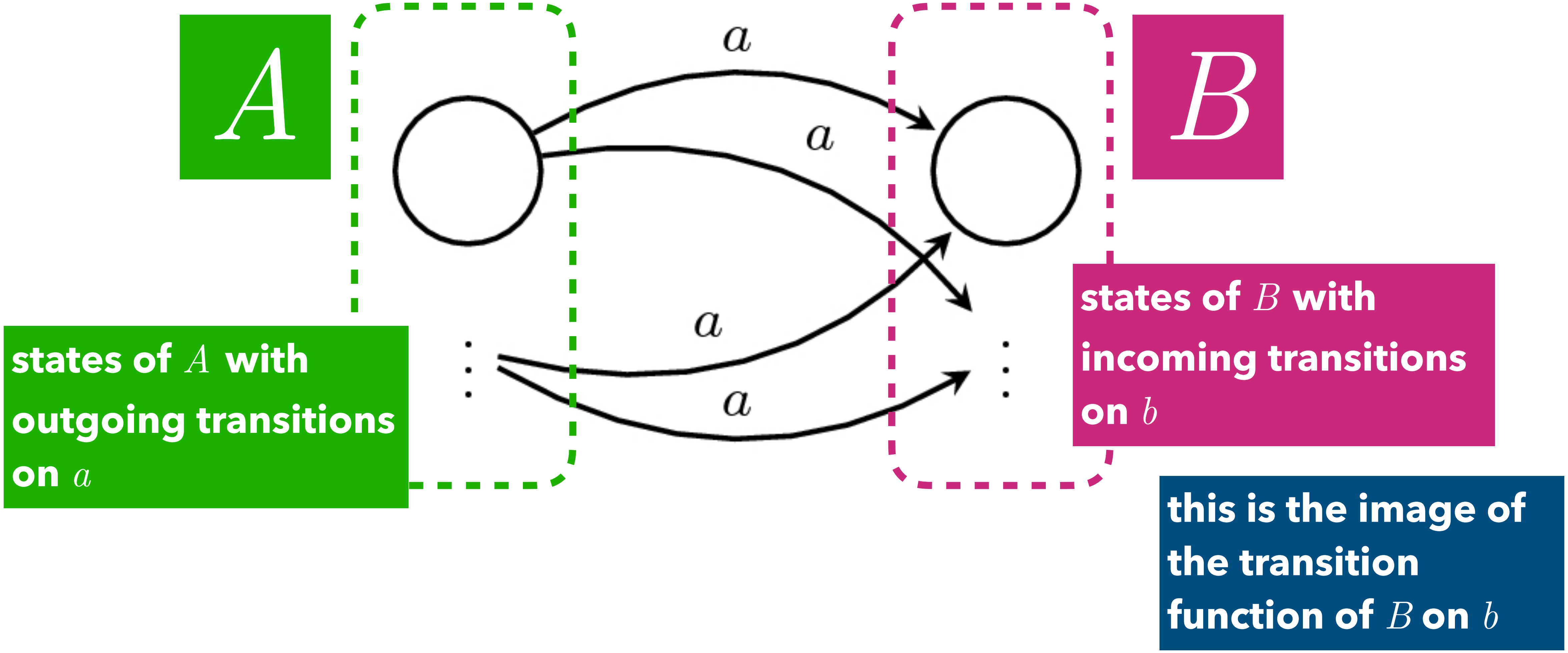
Crossover

A. Mateescu et al. (1998), R. Ceterchi (2006)

The crossover operation is used for the **algebraic characterization** of simple and semi-simple splicing.

The operation can be thought of as a **single step** of simple or semi-simple splicing.

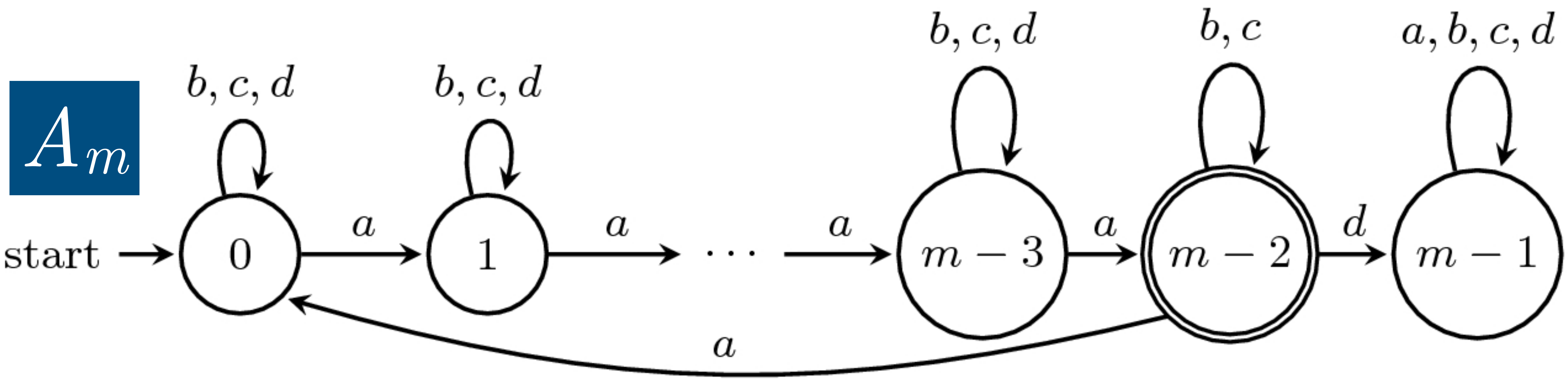
For two DFAs A and B, if $(a,b) \in M$, then for each state of A with outgoing transitions on a, add transitions on a to all states in B with incoming transitions on b.



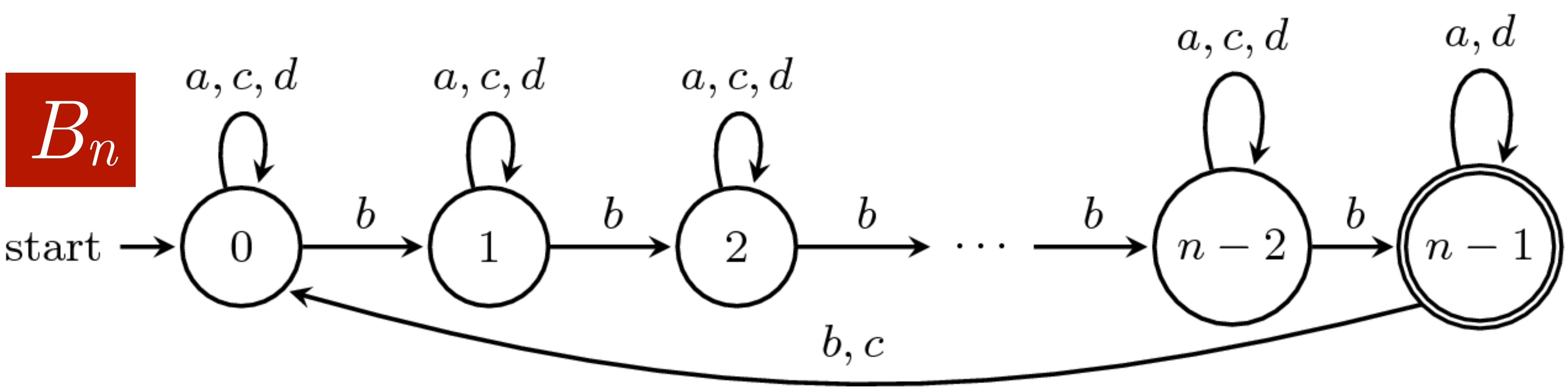
This gives **at most $m \times 2^n$ states**.

$$M = \{(d,d)\}$$

A_m


start
0
1
$m-3$
$m-2$
$m-1$
b, c, d
b, c
a, b, c, d
a
d
$\cdots$

B_n


start
0
1
2
$n-2$
$n-1$
a, c, d
a, d
b
b, c
$\cdots$

Conclusion

- State complexity for **simple** splicing systems with **regular** initial languages
- State complexity for **simple** splicing systems with **finite** initial languages defined over alphabets of size **1, 2, 3, and $\geq 2^{n-3}$**
- State complexity of **semi-simple** splicing systems with **regular** and **finite** initial languages
- State complexity of the **crossover** operation on regular languages

Open problems

- State complexity for **other simple and semi-simple splicing systems (2,4; 2,3; 1,4)** with finite and regular initial languages.

- State complexity of simple splicing systems with finite initial languages over **alphabets of size k for $3 < k < 2^{n-3}$**.

- State complexity of **k-limited splicing systems**, for $k = 1, 2, \ldots$.

Thank you