Descriptional Complexity of Formal Systems

Michal Hospodár

Mathematical Institute
Slovak Academy of Sciences
Košice, Slovakia

Supervisor: RNDr. Galina Jirásková, CSc.

Outline

- Basic Notions
- Results
  1. Concatenation on Alternating Finite Automata
  2. Descriptional Complexity of the Forever Operator
  3. NFA Complexity of L^k and L^+ on Convex Languages
  4. Magic Number Problem for the Cut Operation
  5. Ranges of Accepting State Complexities
- Summary and Open Problems

Basic Notions

Definition (Brzozowski, Leiss 1980)

A Boolean finite automaton is a quintuple $A = (Q, \Sigma, \delta, g_I, F)$ where

- $Q = \{q_1, \ldots, q_n\}$ is a set of states
- Σ is an input alphabet
- δ maps every pair $(q, a) \in Q \times \Sigma$ to a Boolean function over variables $q_1, \ldots, q_n$
- g_I is the initial Boolean function
- $F \subseteq Q$ is the set of final states

Example (Boolean automaton)

$A = (\{q_1, q_2\}, \{a, b\}, \delta, q_1 \vee \neg q_2, \{q_2\})$

δ	a	b
q_1	q_2	$q_1 \wedge \neg q_2$
q_2	q_1	$\neg q_1$

Example (Boolean automaton – cont.)

$A = (\{q_1, q_2\}, \{a, b\}, \delta, q_1 \vee \neg q_2, \{q_2\})$

δ	a	b
q_1	q_2	$q_1 \wedge \neg q_2$
q_2	q_1	$\neg q_1$

Computation on *ab*:

$$q_1 \vee \neg q_2 \xrightarrow{a} q_2 \vee \neg q_1 \xrightarrow{b} (\neg q_1) \vee \neg(q_1 \wedge \neg q_2) = \neg q_1 \vee q_2$$

- evaluate the resulting function at the finality vector $f = (0, 1)$
- gives $1 \Rightarrow ab$ is accepted by A
- the empty string is rejected by A

Basic Notions II

Definition

A Boolean finite automaton $A = (Q = \{q_1, \ldots, q_n\}, \Sigma, \delta, g_I, F)$ is

- alternating (AFA) if $g_I = q_1$
  - Chandra, Kozen, Stockmeyer 1981
  - Fellah, Jürgensen, Yu 1990
  - Birget 1996
  - Yu 1997
- nondeterministic (NFA) if $g_I = q_1$ and each $\delta(q, a)$ is a disjunction
- deterministic (DFA) if $g_I = q_1$ and each $\delta(q, a) = q_i$ for some i

1. Concatenation on Alternating Finite Automata

Known result from:

- A. Fellah, H. Jürgensen, and S. Yu:
  Constructions for alternating finite automata.
  Int. J. Comput. Math. 35 (1990) 117–132.

Theorem (Fellah, Jürgensen, Yu 1990)

If A is an m-state AFA and B is an n-state AFA, then $L(A)L(B)$ is accepted by a $(2^m + n + 1)$-state AFA.

> *"We conjecture that this number of states is actually necessary in the worst case, but have no proof."*

Here: to show that the conjecture holds

Concatenation on DFAs with Multiple Final States

To prove the conjecture from FJY 1990, we use:

Lemma (Brzozowski, Leiss 1980)

L is accepted by an n-state AFA
if and only if
L^R is accepted by a 2^n-state DFA with 2^{n-1} states final.

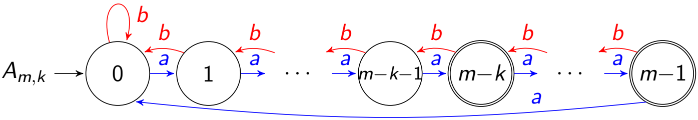
- this motivated us to study concatenation on DFAs with multiple final states
- known: if the first DFA has k final states,
  then the complexity of concatenation is $m2^n - k2^{n-1}$
  – the second witness DFA has one final state

Theorem

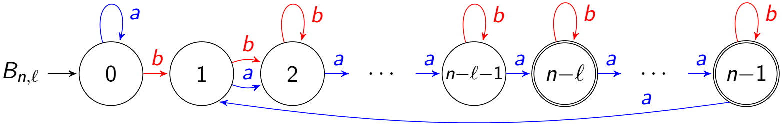
There exist binary languages K and L such that

- *K is accepted by an m-state DFA with k final states,*
- *L is accepted by an n-state DFA with ℓ final states,*
- *every DFA for KL has at least $m2^n - k2^{n-1}$ states.*

Binary Witnesses for Concatenation on DFAs



- $i \cdot a = (i+1) \bmod m$
- $i \cdot b = \max\{0, i-1\}$



- $0 \cdot a = 0, \quad (n-1) \cdot a = 1, \quad$ and $i \cdot a = i+1$ otherwise
- $0 \cdot b = 1, \quad 1 \cdot b = 2, \quad$ and $i \cdot b = i$ otherwise

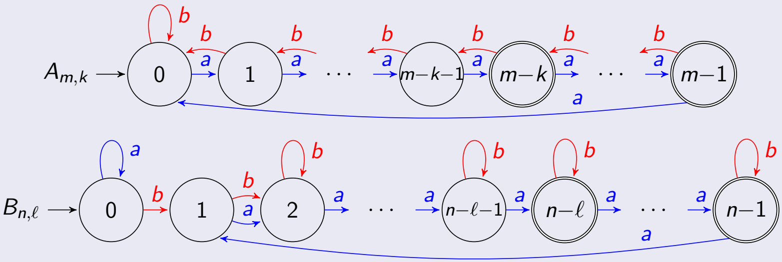
Theorem

Every DFA for $L(A_{m,k})L(B_{n,\ell})$ has at least $m2^n - k2^{n-1}$ states.

Witnesses for Concatenation on AFAs

We use:

Our binary witnesses for concatenation on DFAs:



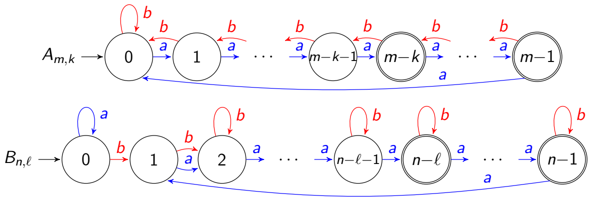
Lemma (Brzozowski, Leiss 1980)

L is accepted by an n-state AFA
if and only if
L^R is accepted by a 2^n-state DFA with 2^{n-1} states final.

...to describe witnesses for concatenation on AFAs

Witnesses for Concatenation on AFAs: Proof Idea

Our binary witnesses for concatenation on DFAs:



- every DFA for $L(A_{m,k})L(B_{n,\ell})$ has at least $m2^n - k2^{n-1}$ states

Binary witnesses for concatenation on AFAs (proof idea):

- $K = L(A_{2^n,2^{n-1}}) \overset{\text{BL-Lemma}}{\Longrightarrow} K^R$ is accepted by an n-state AFA
- $L = L(B_{2^m,2^{m-1}}) \overset{\text{BL-Lemma}}{\Longrightarrow} L^R$ is accepted by an m-state AFA
- every DFA for KL needs $2^n 2^{2^m} - 2^{n-1} 2^{2^m-1}$ states

$\overset{\text{BL-Lemma}}{\Longrightarrow}$ every AFA for $(KL)^R = L^R K^R$ needs $2^m + n + 1$ states

$\Longrightarrow$ L^R and K^R are witnesses for concatenation on AFAs

$\Longrightarrow$ conjecture from FJY 1990 holds

2. Descriptional Complexity of the Forever Operator

Forever operator $L \mapsto (\Sigma^* L^c)^c$

- from temporal logic
- forever := not eventually not

Jean-Éric Pin:

> *"Let $L \subseteq \Sigma^*$ be recognized by an NFA or a DFA with n states. How many states are sufficient and necessary in the worst case for an NFA (DFA) to recognize $(\Sigma^* L^c)^c$?"*

Partial answer by Jean-Camille Birget (1996):

	DFA	$\|\Sigma\|$	NFA	$\|\Sigma\|$	AFA	$\|\Sigma\|$
DFA	2^{n-1}	3	2^{n-1}	3		
NFA			$\geq 2^{n-1}$	3		
			$\leq 2^{n+1}+1$			
AFA					$\leq n+1$	

Here: exact trade-off in each box

Results for the Forever Operator $L \mapsto (\Sigma^* L^c)^c$

Results by Birget

	DFA	$\lvert\Sigma\rvert$	NFA	$\lvert\Sigma\rvert$	AFA	$\lvert\Sigma\rvert$
DFA	2^{n-1}	3	2^{n-1}	3		
NFA			$\geq 2^{n-1}$ $\leq 2^{n+1}+1$	3		
AFA					$\leq n+1$	

Our results

	DFA	$\lvert\Sigma\rvert$	NFA	$\lvert\Sigma\rvert$	AFA	$\lvert\Sigma\rvert$
DFA	2^{n-1}	2			n	3
NFA	$\mathrm{M}(n-1)$	2^{n+1}	2^{n-1}	3	n	2
AFA	$2^{2^{n-1}}$	2	$2^{n-1}+1$	2	n	1

- the most interesting: NFA-to-DFA trade-off
- $\mathrm{M}(n)$ = Dedekind number

Dedekind Number

Definition

Dedekind number $\mathrm{M}(n)$ = the number of antichains of subsets of an n-element set

Example

n	$\mathrm{M}(n)$
0	2
1	3
2	6
3	20
4	168
5	7 581
6	7 828 354
7	2 414 682 040 998
8	56 130 437 228 687 557 907 788

$$2^{2^{n-\log n}} \leq \mathrm{M}(n) \leq 2^{2^{n-\frac{\log n}{3}}}$$

NFA-to-DFA Trade-Off for the Forever Operator

Construction of a DFA for $(\Sigma^* L^c)^c$

n-state NFA for L with $Q = \{0, 1, \ldots, n-1\}$ and $q_I = 0$

$\Downarrow$ determinization + inverting finality

2^n-state DFA for L^c — states = subsets of Q

$\Downarrow$ adding a loop on each letter in the initial state $\{0\}$

2^n-state NFA for $\Sigma^* L^c$

$\Downarrow$ determinization + inverting finality

2^{2^n}-state DFA D for $(\Sigma^* L^c)^c$ — states = sets of subsets of Q

NFA-to-DFA: tight upper bound = $\mathrm{M}(n-1)$

- upper bound: show that each state of D is equivalent to an antichain on set $\{1, 2, \ldots, n-1\}$
- tightness for $|\Sigma| = 2^{n+1}$
- conjecture: 6-letter alphabet should work

3. Complexity of L^k and L^+ on Convex NFA Languages

3. Complexity of L^k and L^+ on Convex NFA Languages

- prefix, suffix of w
- factor of w := contiguous subsequence of w
- subword of w := scattered subsequence of w

Example

$w = BRATISLAVA$
$BRAT$ is a prefix of w
$LAVA$ is a suffix of w
RAT is a factor of w
BTS is a subword of w

A language is

- prefix-free if $w \in L \Rightarrow$ no proper prefix of w is in L
- prefix-closed if $w \in L \Rightarrow$ every prefix of w is in L
- prefix-convex if $u, v \in L$ and $u \leq_p v \Rightarrow$ every w with $u \leq_p w$ and $w \leq_p v$ is in L

analogously suffix-, factor-, subword-free, -closed, -convex

Subclasses of Convex Languages

A language L is

- right ideal if $L = L\Sigma^*$
- left ideal if $L = \Sigma^* L$
- two-sided ideal if $L = \Sigma^* L \Sigma^*$
- all-sided ideal if $L = L \sqcup\!\sqcup \Sigma^*$

- prefix-free, prefix-closed, and right ideal languages are also prefix-convex
- similarly suffix- (left), factor- (two-sided), subword- (all-sided)

History

- operations on DFAs for convex classes: Brzozowski et al. (2013-14)
- operations on NFAs for convex classes: Mlynárčik et al. (2014-17)

Here: L^k and L^+ on NFAs for subclasses of convex languages

Definition

$L^k = \{u_1 u_2 \cdots u_k \mid u_i \in L \text{ for } i = 1, 2, \ldots, k\}$ $\qquad L^+ = \bigcup_{i \geq 1} L^i$

Lower-bound methods:

- fooling set method
- lemma guaranteeing the existence of a fooling set

Lemma

Let N be an NFA with the state set $\{1, 2, \ldots, n\}$. If for each i,

- *$\{i\}$ is reachable,*
- *$\{1, 2, \ldots, i\}$ is co-reachable (i.e., reachable in N^R),*

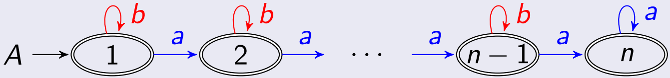
then N is minimal.

k-th Power on Subword-Closed Languages

If L is accepted by an n-state NFA N, then L^k is accepted by a kn-state NFA (take k copies of N and connect them properly).

Theorem (the most interesting result of this chapter)

Let L be accepted by an NFA A. Then L is subword-closed, and every NFA for L^k has at least kn states.



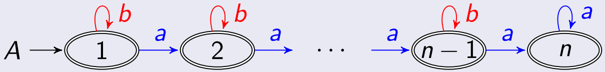
Proof idea.

- show that L^k is recognized by a partial DFA D
  – k copies of A connected by transitions $(jn, b, jn+1)$
- $\{i\}$ is reachable in D
- $\{1, 2, \ldots, i\}$ is co-reachable in D

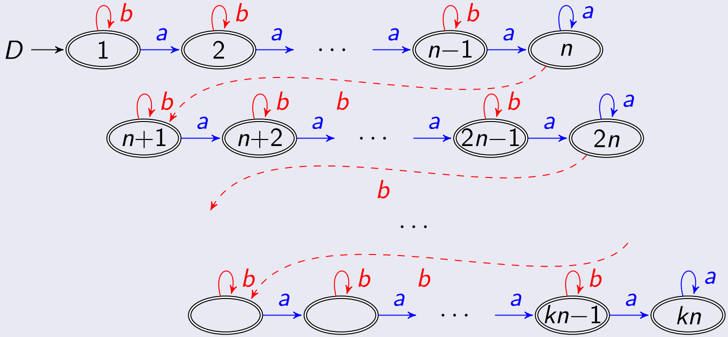
$\Rightarrow$ D is a minimal NFA □

The Most Interesting Result of This Chapter

NFA A for L



L^k is recognized by partial DFA D



- $\{i\}$ is reachable and $\{1, 2, \ldots, i\}$ is co-reachable

$\Rightarrow$ D is a minimal NFA $\Rightarrow$ every NFA for L^k has at least kn states

L^k and L^+ on Subclasses of Convex Languages

Our results

	L^k	$\|\Sigma\|$	L^+	$\|\Sigma\|$
free	$k(n-1)+1$	1	n	1
ideal	$k(n-1)+1$	1	n	1
prefix-, suffix-closed	kn	2	n	2
factor-, subword-closed	kn	2	1	1
convex	kn	2	n	1
regular	kn	2 [DO]	n	1 [HK]

- all alphabets are optimal

[DO] = Domaratzki, Okhotin: State complexity of power. Theoret. Comput. Sci, 2009

[HK] = Holzer, Kutrib: Nondeterministic descriptional complexity of regular languages. Internat. J. Found. Comput. Sci., 2003

4. The Magic Number Problem for the Cut Operation

4. The Magic Number Problem for the Cut Operation

The magic number problem for NFA-to-DFA conversion

- stated by Iwama, Kambayashi, Takaki 2000:
  "Given a minimal n-state NFA,
  how many states can the equivalent minimal DFA have?"

$\Rightarrow$ not only worst-case complexity, but rather
all possible complexities of the resulting languages

Cut operation (motivated by UNIX text processors)

The cut of languages K and L is the language

$K \,!\, L = \{uv \mid u \in K, v \in L, uv' \notin K$ for every non-empty prefix v' of $v\}$

The magic number problem for cut

Given two minimal DFAs of m and n states,
how many states can the minimal DFA
for the cut of their languages have?

The Cut Operation

Definition (Berglund, Björklund, Drewes, v.d.Merwe, Watson 2013)

The cut of languages K and L over Σ is the language

$K \,!\, L = \{uv \mid u \in K, v \in L, uv\surd \notin K$ for every non-empty prefix $\surd$ of $v\}$

The complexity of cut on DFAs (Drewes, Holzer, Jakobi, v.d.M. 2017)

$\lvert\Sigma\rvert \geq 2$	$(m-1)n+m$	with binary witnesses
$\lvert\Sigma\rvert = 1$	$\kappa = \max\{2m-1, m+n-2\}$	

Our results on the magic number problem for cut

	$1..2m{-}1$	$2m..n{-}1$	$n..m{+}n{-}2$	$\kappa{+}1..(m{-}1)n{+}m$
$\lvert\Sigma\rvert \geq 2$	✓	✓	✓	✓
$\lvert\Sigma\rvert = 1$	✓	✗	✓	–

- the magic number problem completely solved for every alphabet

5. The Ranges of Accepting State Complexities

5. The Ranges of Accepting State Complexities

Definition (Dassow 2016)

The accepting state complexity of L
$:=$ the smallest number of accepting states in any DFA for L.

Lemma (Dassow 2016)

$:=$ the number of accepting states in the minimal DFA for L.

Here: the ranges of accepting state complexities for

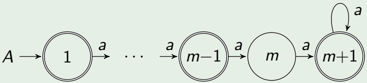
- intersection,
- symmetric difference,
- right and left quotient,
- reversal,
- permutation on binary finite languages,
- cut operation

Accepting State Complexities for Symmetric Difference

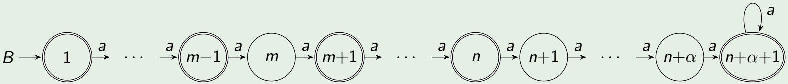
Example (Symmetric Difference)

W.l.o.g., $m \leq n$. Let $\alpha \geq 1$.

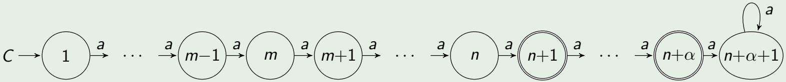
- Minimal unary DFA with m accepting states:



- Minimal unary DFA with n accepting states:



- Minimal unary DFA for $L(A) \oplus L(B)$ with α accepting states:



Results on Ranges of Accepting State Complexities

Known results [Dassow 2016]

operation	range of asc	$\|\Sigma\|$
difference	$[0, \infty)$	1
union	$[1, \infty)$	1
concatenation	$[1, \infty)$	1
star	$[1, \infty)$	1
complementation	$[1, \infty)$	1
intersection	$\subseteq [0, mn]$	

Our results

intersection	$[0, mn]$	2
right and left quotient	$[0, \infty)$	1
cut operation	$[0, \infty)$	1
symmetric difference	$[1, \infty)$	1
reversal	$[1, \infty)$	2
permutation on binary finite languages	$[2, \infty)$	2

Summary

1. concatenation on AFAs: $2^m + n + 1$
   (solves the conjecture from FJY 1990)
2. forever operator $L \mapsto (\Sigma^* L^c)^c$:
   - trade-offs between six different models of finite automata
   - exact in 32 out of 36 cases
3. the complexity of L^k and L^+ on NFAs
   in subclasses of convex languages
4. the magic number problem for cut on DFAs:
   completely solved
5. ranges of accepting state complexities:
   intersection, symmetric difference, right and left quotient,
   reversal, permutation, and cut

Open Problems

- concatenation on unary AFA languages:
  $m + n - 1 \leq \cdot \leq m + n + 1$
- MNFA-to-{DFA, partial DFA, NFA, MNFA} trade-offs
  for the forever operator $L \mapsto (\Sigma^* L^c)^c$
- DFA complexity of L^k and L^+
  in subclasses of convex languages
- cut operation on NFAs:
  complexity, magic number problem
- ranges of accepting state complexities
  for other operations (shuffle, power...)

Publications

1. M. Hospodár, G. Jirásková: The complexity of concatenation on deterministic and alternating finite automata. RAIRO – Theoretical Informatics and Applications 52, 153–168 (2018)
2. M. Hospodár, G. Jirásková, P. Mlynárčik: Descriptional complexity of the forever operator. International Journal of Foundations of Computer Science 30, 115–134 (2019)
3. M. Hospodár: Descriptonal complexity of power and positive closure on convex languages. Proc. CIAA 2019, Lecture Notes in Computer Science, vol. 11601, pp. 158–170. Springer (2019)
4. M. Holzer, M. Hospodár: The range of state complexities of languages resulting from the cut operation. Proc. LATA 2019, Lecture Notes in Computer Science, vol. 11417, pp. 190–202. Springer (2019)
5. M. Hospodár, M. Holzer: The ranges of accepting state complexities of languages resulting from some operations. Proc. CIAA 2018, Lecture Notes in Computer Science, vol. 10977, pp. 198–210. Springer (2018)

Questions from Reviewers

G. Andrejková

- Operations on languages in each class could be combined in many different ways. For example, the definition of "symmetric difference" operation is done by other investigated binary operations. How can you find interesting and important combinations of operations for the theory of formal languages?

Answer:

- 2007 Salomaa, Salomaa, Yu: State complexity of combined operations $\rightarrow$ start of systematic study
- many other combined operations without complementation...
- 2011 Brzozowski, Grant, Shallit: Closures in formal languages and Kuratowski's theorem $\rightarrow$ star-complement-star
- 1996 Birget: $(\Sigma^* L^c)^c$ $\rightarrow$ not intended as a combined operation

Questions from Reviewers

G. Andrejková

- Operation "forever" in the open problem 4. You have upper and lower bounds on trade-offs starting with MNFA. Could you explain why is it difficult to determine them?

Each state is equivalent to an antichain $\Rightarrow$ we count antichains

NFA-to-DFA trade-off:

- reachable states are of the form $\{\{s\}, S_2, S_3, \ldots, S_k\}$ where $\{S_2, S_3, \ldots, S_k\}$ is an antichain in $2^{Q \setminus \{s\}}$
- there is $\mathrm{M}(n-1)$ such states

MNFA-to-DFA trade-off:

- reachable states are of the form $\{I, S_2, S_3, \ldots, S_k\}$ where $\{I, S_2, S_3, \ldots, S_k\}$ is an antichain in 2^{Q}
- there is at most $\mathrm{M}(n)$ such states, but we do not know how many of them are reachable
- computations: $\mathrm{M}(n-1) < \cdot < \mathrm{M}(n)$

Questions from Reviewers

J. Dassow

- The notation asc is used for the state complexity of languages accepted by alternating finite automata as well as for the accepting state complexity of regular languages.

Answer: We could use $\mathrm{afa\text{-}sc}(L)$ for alternating state complexity.

J. Dassow

The notation is not very unified;

- in most cases the author uses $f(m, n)$ without a specification of the complexity measure, however, in Chapter 6 the measure is used as an upper index;
- sometimes the unary case is denoted by 1, sometimes by u.

Answer:
The notations $f(m, n)$ and $f_1(m, n)$ are just for two different functions. The index 1 does not mean "unary".

Questions from Reviewers

J. Dassow

- In Section 6.4 the operation can only be applied to finite languages such that the notations without f in the upper index (in Theorem 6.17) should be avoided.
- In Table 6.1, the result concerning permutations is not correctly given since the notation for the range is false.

Theorem 6.17. *We have*

$$g_{\mathrm{per}}^{\mathrm{asc}}(n) = g_{\mathrm{per}}^{\mathrm{asc},f}(n) = \begin{cases} \{0\}, & \text{if } n = 0; \\ \mathbb{N}, & \text{if } n = 1; \\ \mathbb{N} \setminus \{1\}, & \text{if } n \geq 2. \end{cases}$$

For unary regular languages, we have $g_{\mathrm{per}}^{\mathrm{asc},u}(n) = \{n\}$ *if* $n \geq 0$.

In the journal version of CIAA 2018 paper submitted to IJFCS, the notation $g_{per}^{\mathrm{asc}}(n)$ is avoided, only $g_{per}^{\mathrm{asc},f}(n)$ is used.

Questions from Reviewers

F. Mráz: remark for p.74, paragraph Next, the number...

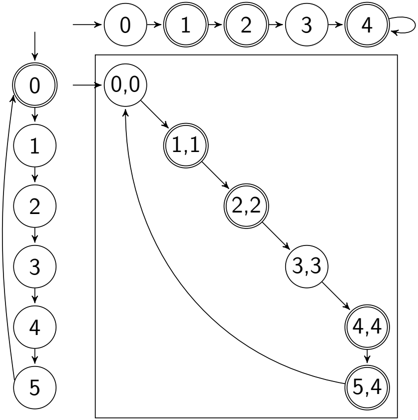
- The proposed modification of DFA B (when $m = 1$) consisting of adding one non-final state into DFA B results into $\alpha - 1$ accepting states instead of the desired α accepting states as the resulting automaton has only $m + \alpha + 1 = \alpha + 2$ states out of which 3 states will be non-final. I think that the automaton B should not be modified, and the original automaton B sould suffice.

Answer: You are right if $\alpha \neq 2n - 2$.
However, if $m = 1$ and $\alpha = 2n - 2$, then the resulting DFA would be non-minimal without a modification.

A detailed explanation is in the next slide.

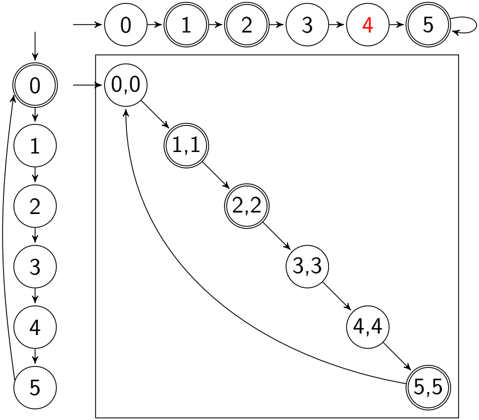
ASC of Cut, Case $m = 1$ and $\alpha = 2n - 2$

Without modification



Result has $\alpha/2$ final states after minimization

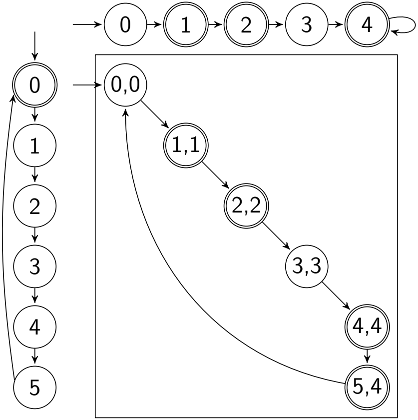
With incorrect modification



Result has $\alpha - 1$ final states

ASC of Cut, Case $m = 1$ and $\alpha = 2n - 2$

Without modification



With correct modification