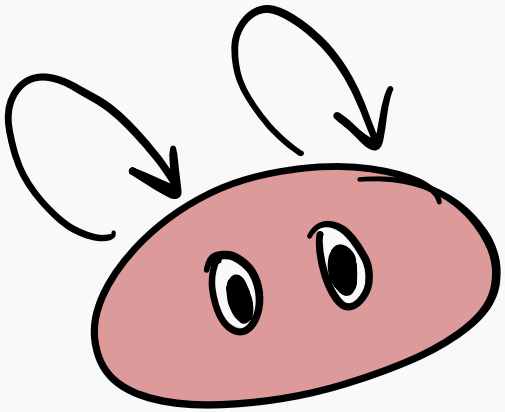
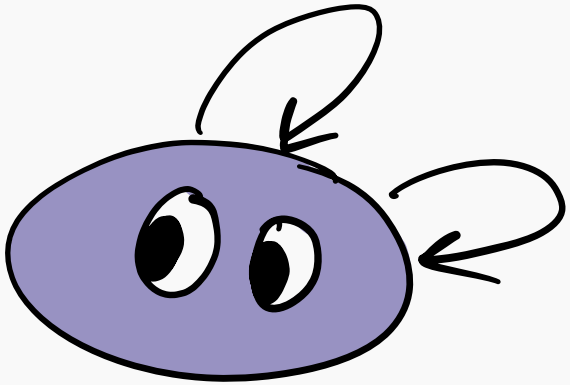
Intersecting Dense Automata

Constructions, Complexity and Certificates



Neha Rino

University of Warwick



Joint work with Dmitry Chistikov

Thanks to:
Centre for Discrete Mathematics and its Applications (DIMAP)
Feuer International Scholarship in AI
Engineering & Physical Sciences Research Council, UK

Plan

I NFA K-INTERSECTION (EMPTINESS)

II PRODUCT CONSTRUCTIONS

III REDUCTION FROM K-CLIQUE

IV CERTIFICATES AND SETH

Problems of Interest

Input: k NFAs $\mathcal{A}_1, \ldots \mathcal{A}_k$.

Problems of Interest

Input: k NFAs $\mathcal{A}_1, \ldots \mathcal{A}_k$.

Intersection

An NFA for $\bigcap_{i=1}^{k} \mathcal{L}(\mathcal{A}_i)$.

Problems of Interest

Input: k NFAs $\mathcal{A}_1, \ldots \mathcal{A}_k$.

Intersection

An NFA for $\bigcap_{i=1}^{k} \mathcal{L}(\mathcal{A}_i)$.

Intersection Emptiness

Is $\bigcap_{i=1}^{k} \mathcal{L}(\mathcal{A}_i) = \emptyset$?

Problems of Interest

Input: k NFAs $\mathcal{A}_1, \ldots \mathcal{A}_k$.

Intersection

An NFA for $\bigcap_{i=1}^{k} \mathcal{L}(\mathcal{A}_i)$.

NFA k-IE

Is $\bigcap_{i=1}^{k} \mathcal{L}(\mathcal{A}_i) = \emptyset$?

Complexity of Intersecting NFAs $\mathcal{A}_1, \ldots, \mathcal{A}_k$

Solution: Construct the direct product of $\mathcal{A}_1, \ldots, \mathcal{A}_k$.

[Rabin and Scott, 1959]

Intersecting NFAs $\mathcal{A}_1, \ldots, \mathcal{A}_k$ with n states and m transitions

Solution: Construct the direct product of $\mathcal{A}_1, \ldots, \mathcal{A}_k$.

Intersection

Size of the direct product is $O(m^k) = O(n^{2k})$.

[e.g. Esparza and Blondin, Automata Theory: An Algorithmic Approach, 2023]

NFA k-IE

DFS running time = $O(m^k) = O(n^{2k})$.

[e.g. Baier and Katoen, Principles of Model Checking, 2008]

Intersecting NFAs $\mathcal{A}_1, \ldots, \mathcal{A}_k$ with n states and m transitions

Solution: Construct the direct product of $\mathcal{A}_1, \ldots, \mathcal{A}_k$.

Intersection

Size of the direct product is $O(m^k) = O(n^{2k})$.

$\bigcap_{i=1}^{k} \mathcal{L}(\mathcal{A}_i)$ can require $\Omega(n^k)$ states

[Maslov, 1970].

NFA k-IE

DFS running time = $O(m^k) = O(n^{2k})$.

Faster than $\Omega(n^k)$-time $\implies$ breakthrough in solving SAT

[Wehar, 2014].

Today's question

Intersection and intersection emptiness (NFA k-IE)

Can we close this gap between $O(n^{2k})$ and $\Omega(n^k)$?

Today's question

Intersection and intersection emptiness (NFA k-IE)

Can we close this gap between $O(n^{2k})$ and $\Omega(n^k)$?

Yes!

Our contribution

- An $O(mn^{k-1}) = O(n^{k+1})$-sized NFA for the intersection.

Our contribution

- An $O(mn^{k-1}) = O(n^{k+1})$-sized NFA for the intersection.

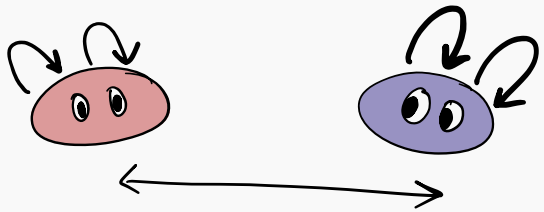
- A conditionally optimal algorithm for NFA k-IE.

Our contribution

- An $O(mn^{k-1}) = O(n^{k+1})$-sized NFA for the intersection.
- A conditionally optimal algorithm for NFA k-IE.
- A faster certificate scheme for NFA k-IE.

Plan

I. NFA k-INTERSECTION (EMPTINESS)

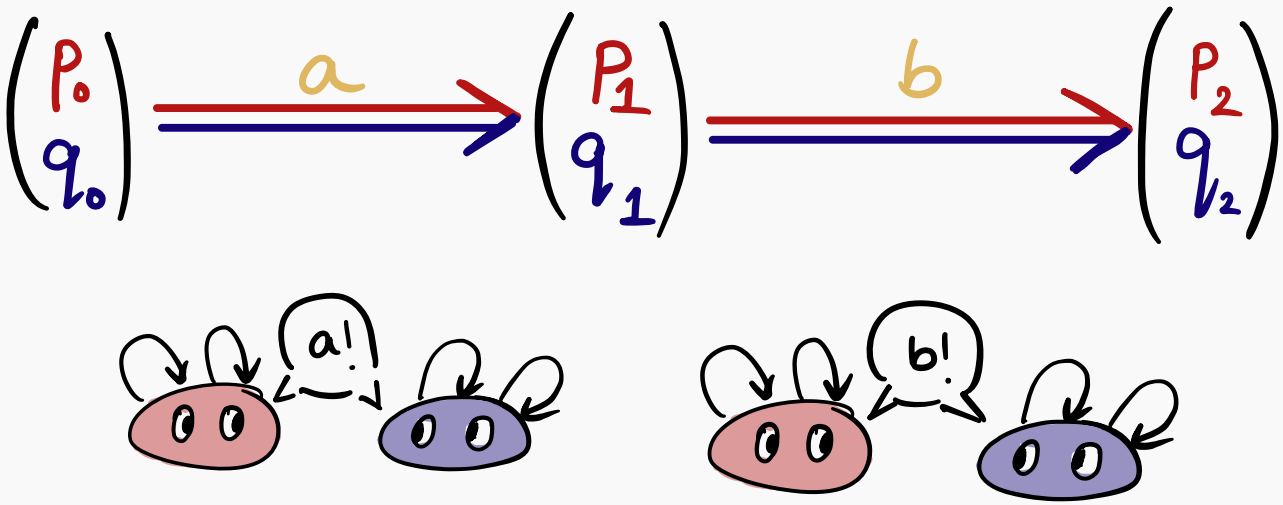


II PRODUCT CONSTRUCTIONS

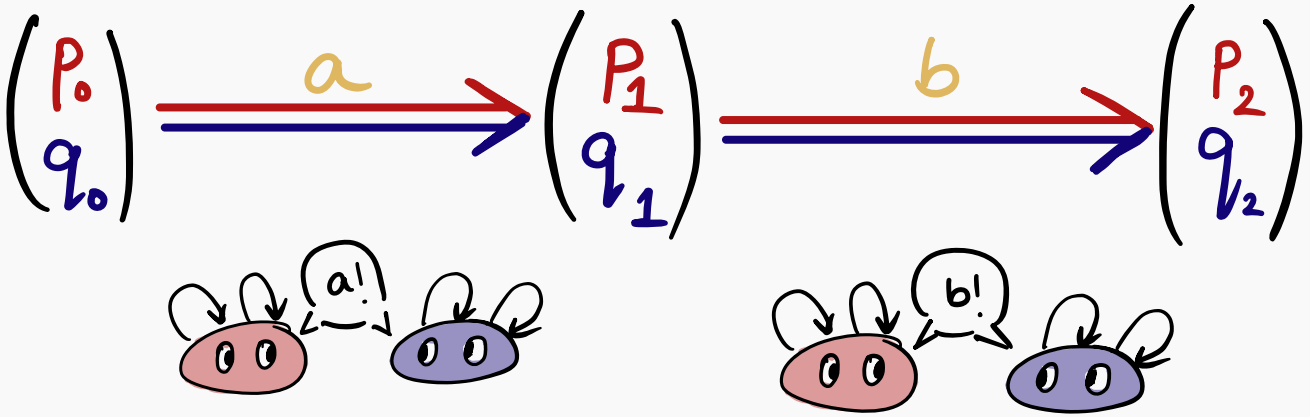
III REDUCTION FROM k-CLIQUE

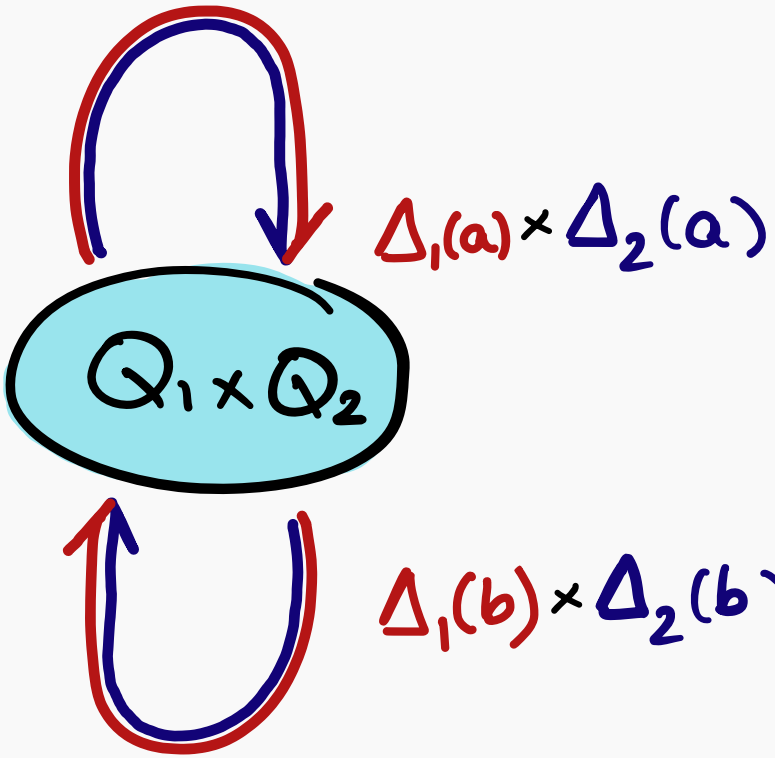
IV CERTIFICATES AND SETH

I. Direct product



I. Direct product



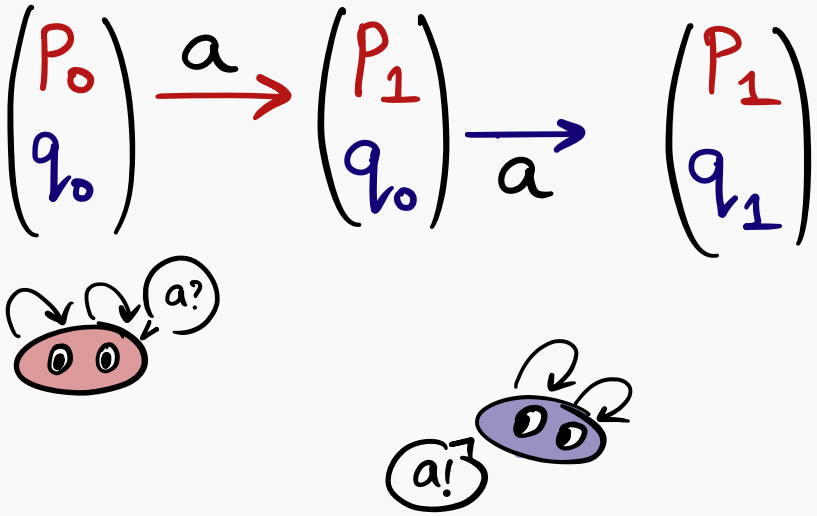


$$
\begin{aligned}
& |\Delta_1(a)| \times |\Delta_2(a)| \\
+\ & |\Delta_1(b)| \times |\Delta_2(b)| \\
=\ & O(m^2)
\end{aligned}
$$

II. Idea: Silently Nodding

Central Idea

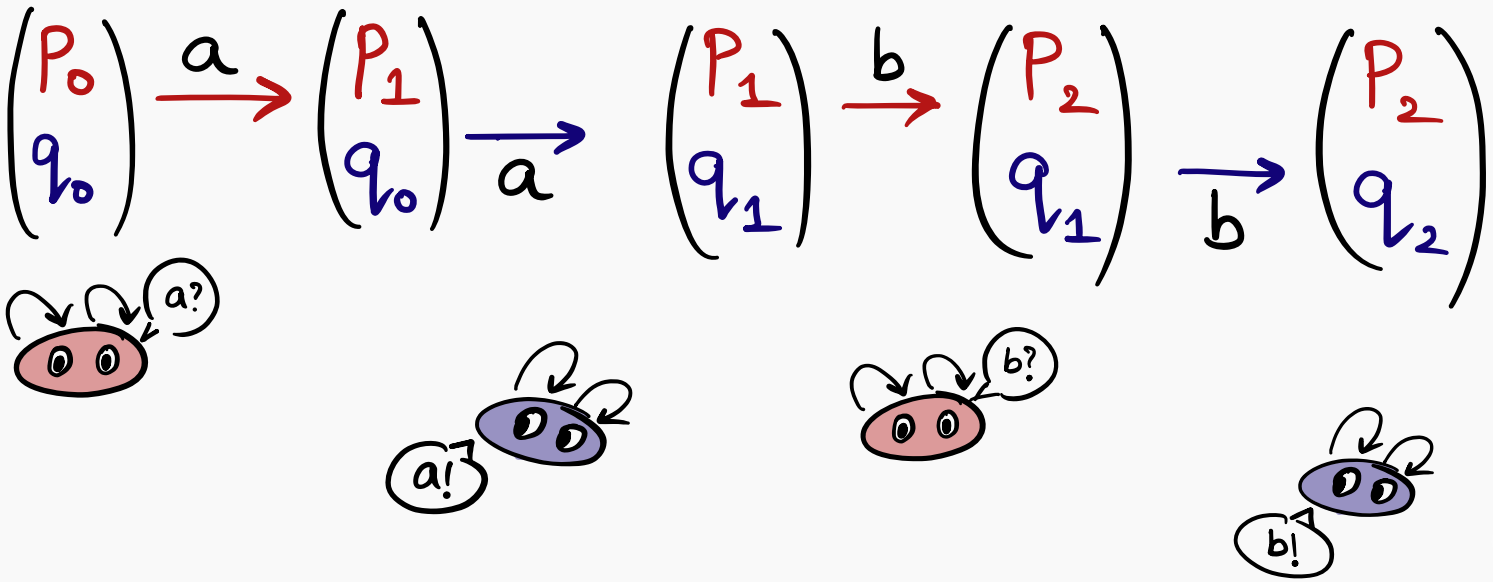
Interleave transitions?



II. Idea: Silently Nodding

Central Idea

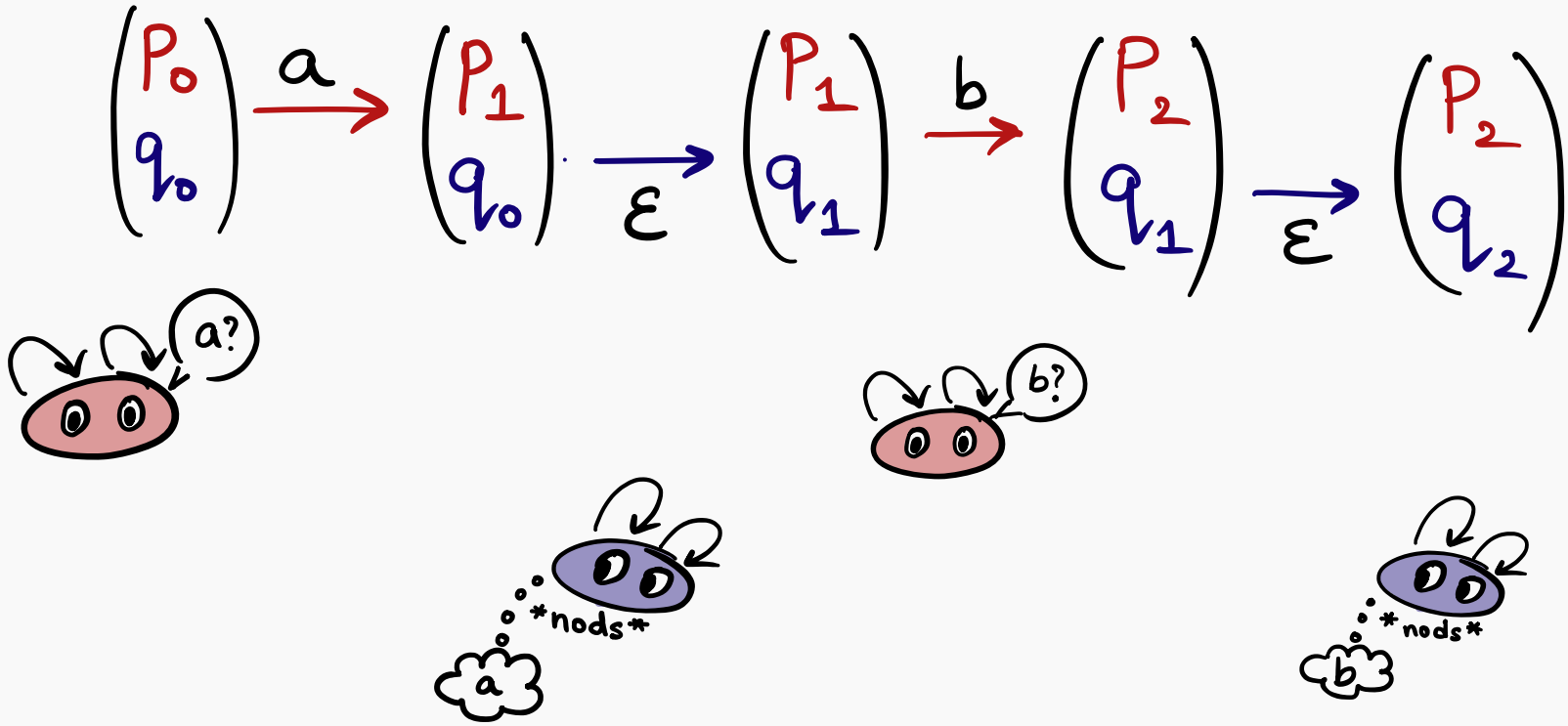
Interleave transitions?



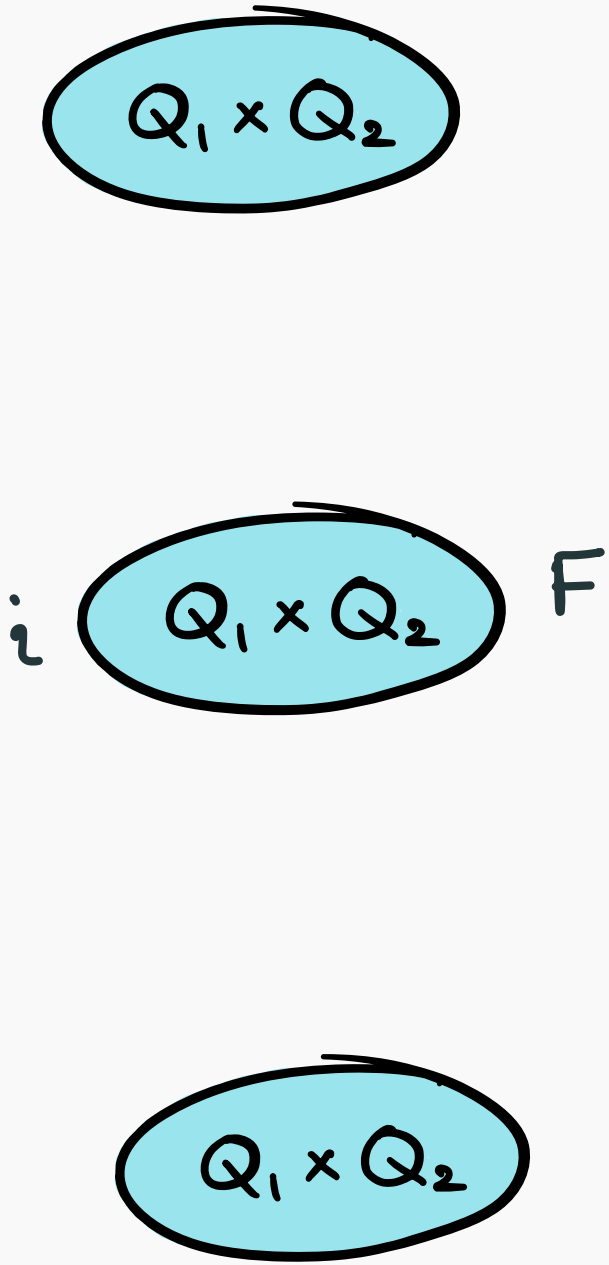
II. Idea: Silently Nodding

Central Idea

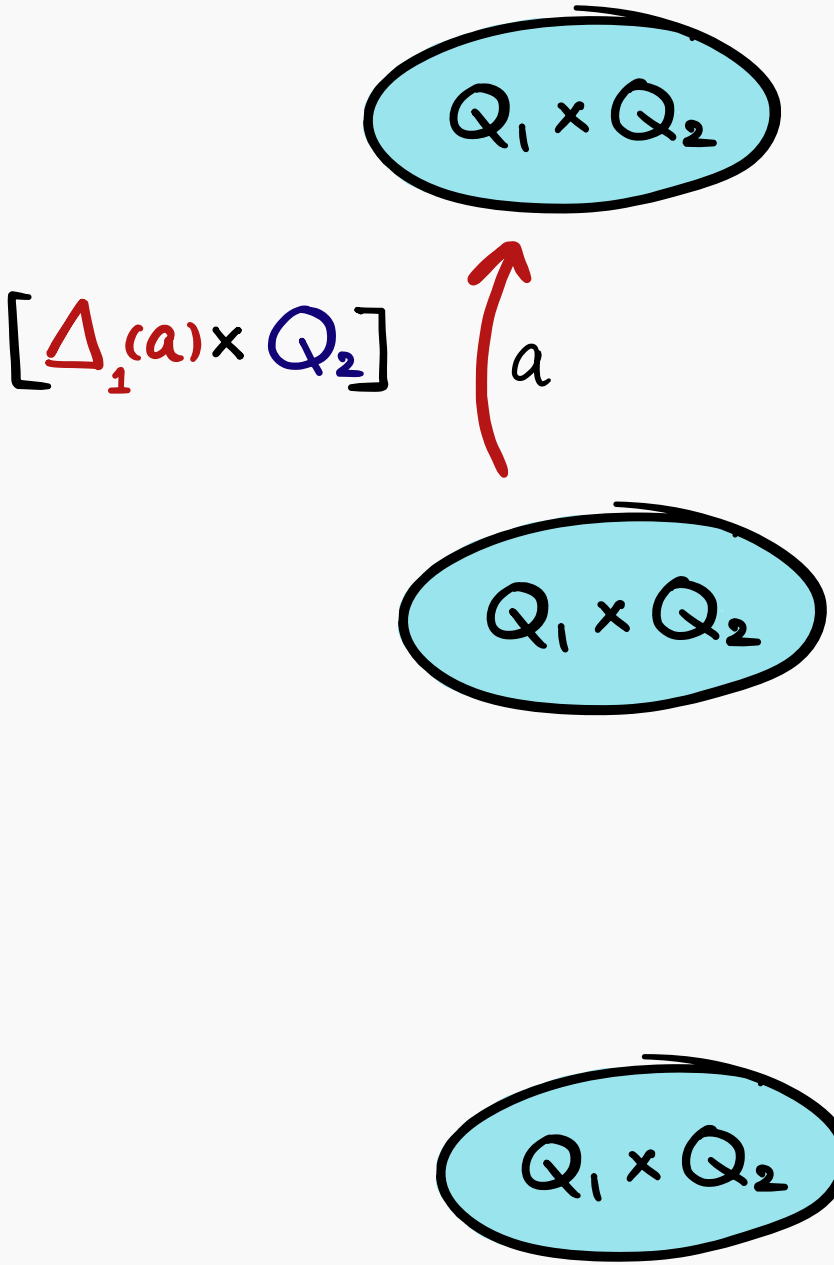
Interleave transitions while preserving the word read.



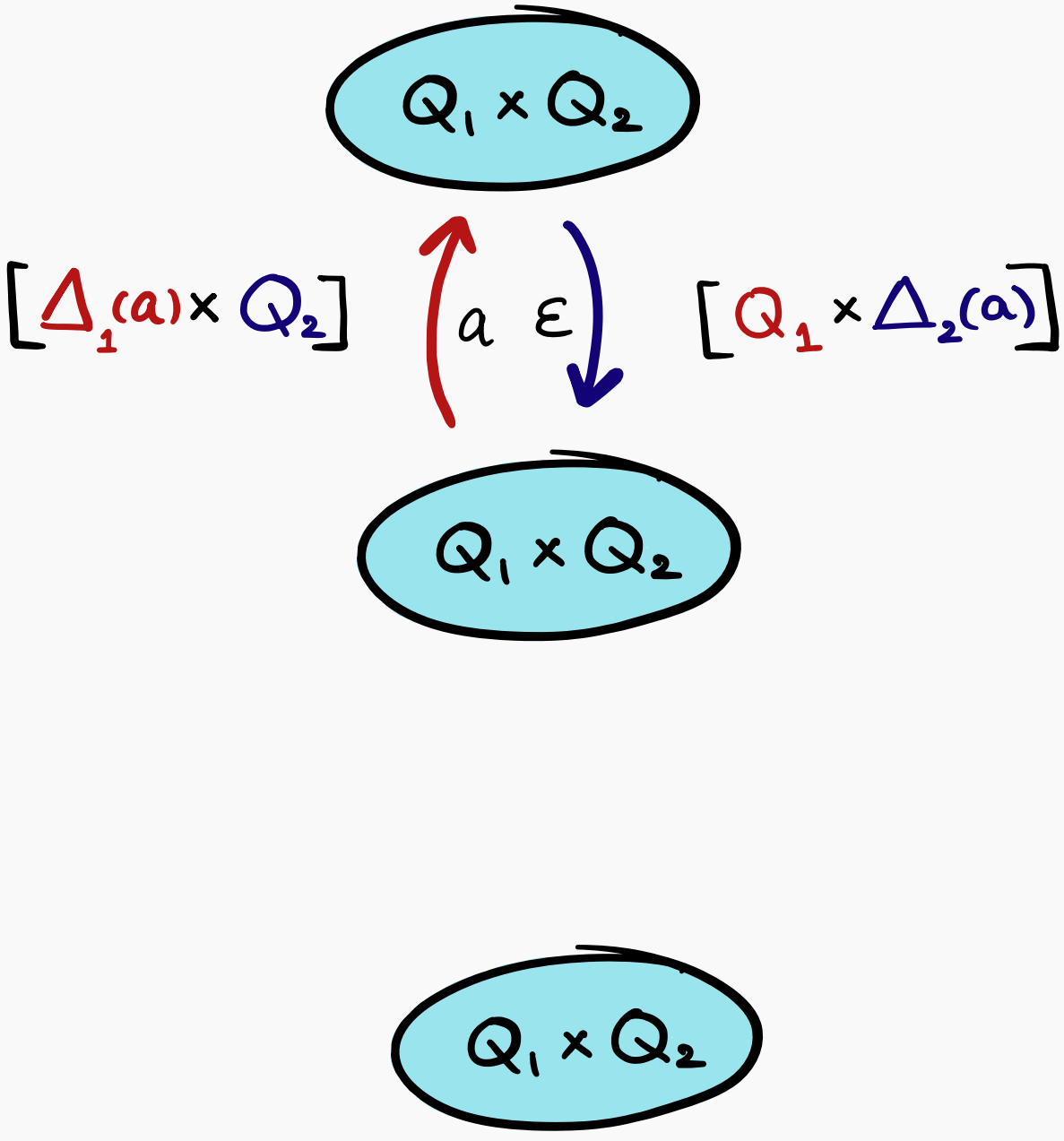
II. Nodding Product



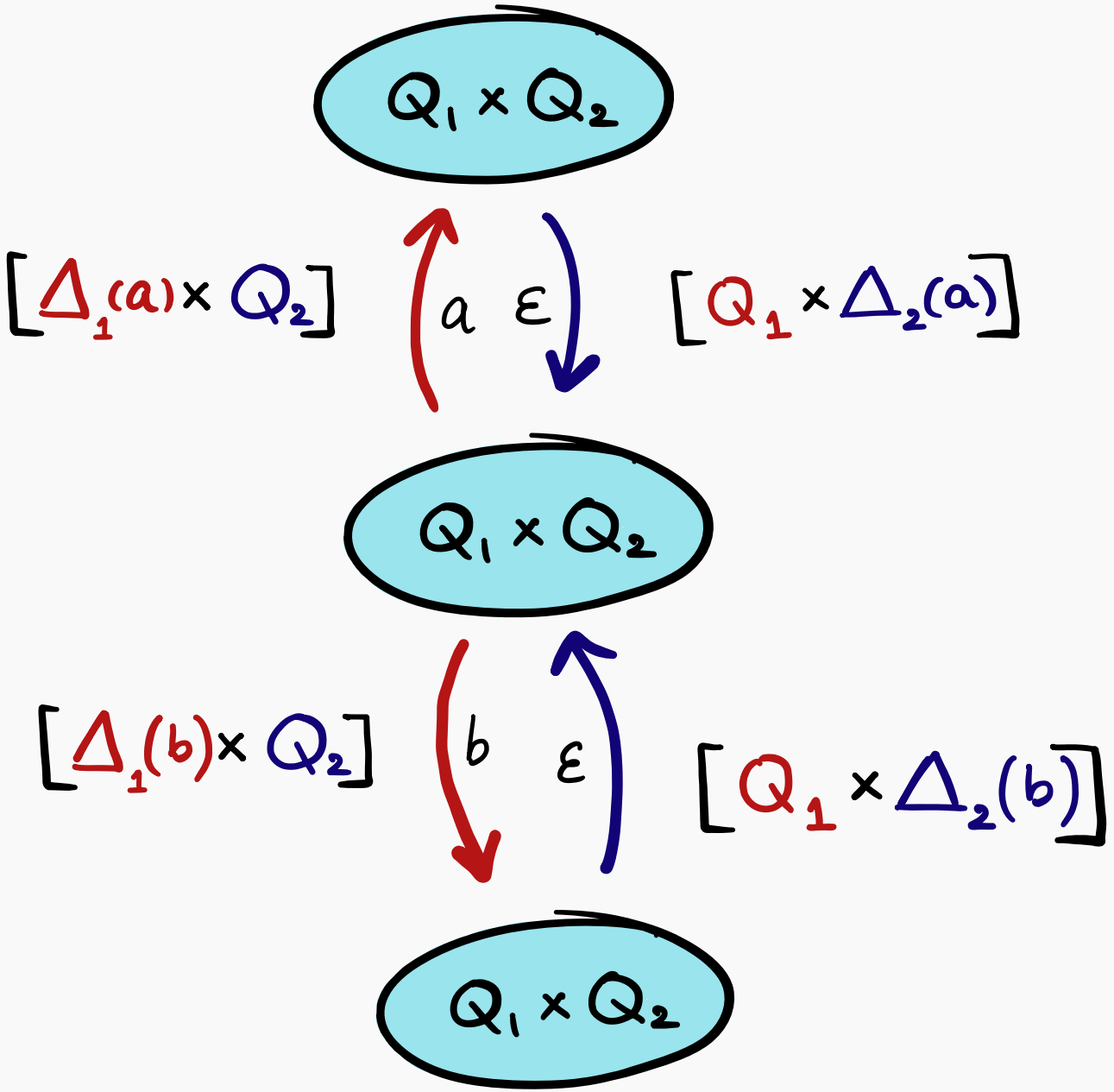
II. Nodding Product



II. Nodding Product

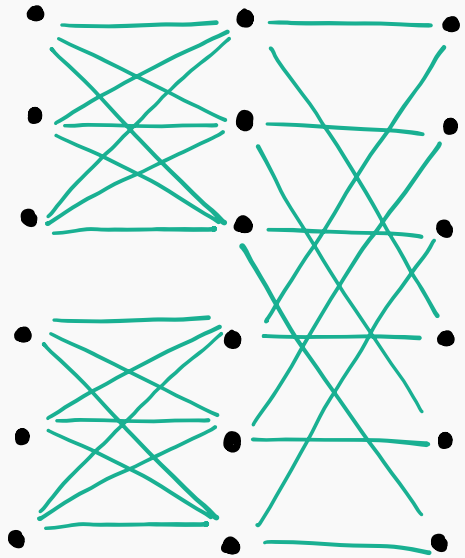


II. Nodding Product



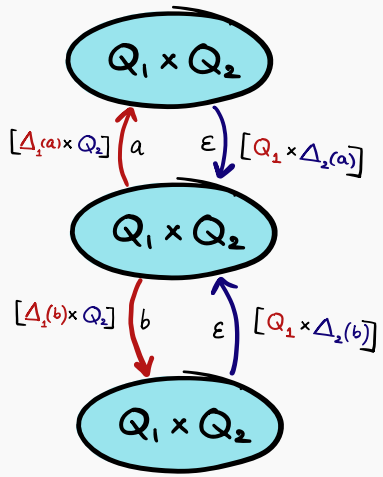
II. Nodding Product

Direct Product
(Dense)



Nodding product
(Sparse)

II. Size of the nodding product



The nodding product recognises the intersection, using

- $O(n^2)$ states
- $O(mn)$ transitions

II. Size of the nodding product

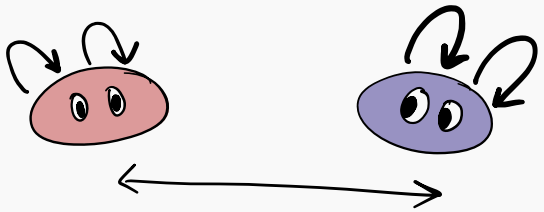
Theorem (1)

Given k NFAs over an alphabet of size ℓ, the nodding product recognises their intersection using

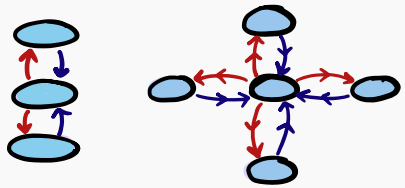
- $O(k\ell \cdot n^k)$ *states*
- $O(k \cdot mn^{k-1})$ *transitions.*

Plan

I NFA K-INTERSECTION (EMPTINESS)



II PRODUCT CONSTRUCTIONS



III REDUCTION FROM K-CLIQUE

IV CERTIFICATES AND SETH

I. Algorithm for NFA k-IE

Algorithm: Search in the nodding product, constructed on the fly. Check if some final state is reachable.

I. Algorithm for NFA k-IE

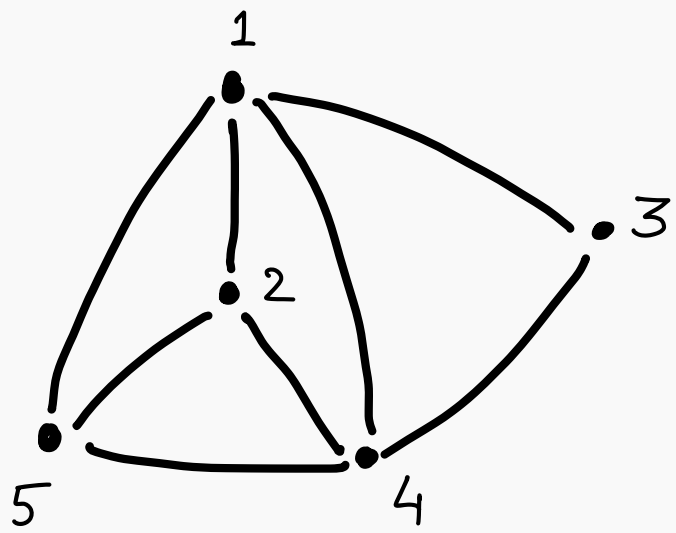
Algorithm: Search in the nodding product, constructed on the fly. Check if some final state is reachable.

Running time: The number of reachable states and transitions in the nodding product is $O(k \cdot mn^{k-1})$.

II. Spot the pattern

Pop quiz

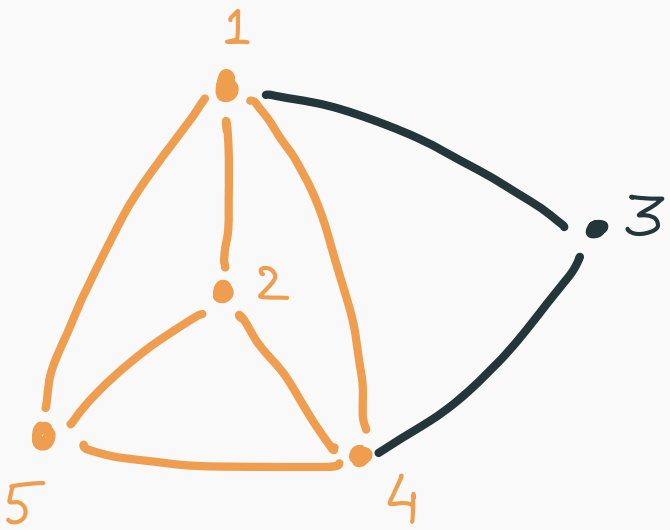
Does the following graph contain 4 vertices that are all pairwise adjacent?



II. Spot the pattern

k-Clique problem

Given a graph G, does it contain k vertices which are all pairwise adjacent?



Yes! {1, 2, 4, 5}

II. The k-Clique problem

k-Clique problem

Given a graph G, does it contain k vertices which are all pairwise adjacent?

The best known running times (upto polylog factors) for k-Clique algorithms are:

II. The k-Clique problem

k-Clique problem

Given a graph G, does it contain k vertices which are all pairwise adjacent?

The best known running times (upto polylog factors) for k-Clique algorithms are:

$O(n^{\omega\lceil\frac{k}{3}\rceil})$ time using fast matrix multiplication.

[Nešetřil and Poljak, 1985]

$O(n^k)$ time without fast matrix multiplication.

[Abboud et al, 2024]

II. The k-Clique problem

k-Clique problem

Given a graph G, does it contain k vertices which are all pairwise adjacent?

The best known running times (upto polylog factors) for k-Clique algorithms are:

$O(n^{\omega\lceil\frac{k}{3}\rceil})$ time using fast matrix multiplication.

[Nešetřil and Poljak, 1985]

$O(n^k)$ time without fast matrix multiplication.

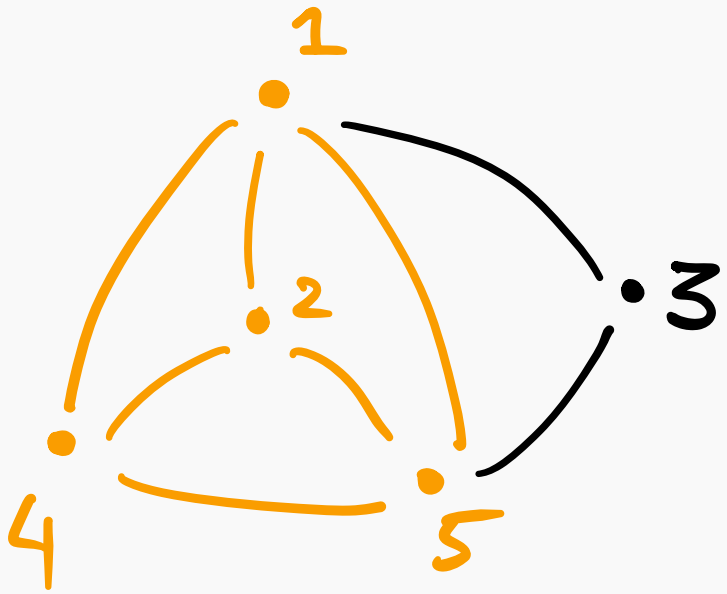
[Abboud et al, 2024]

$n \times n$ matrix multiplication is in $O(n^\omega)$ time, where $\omega < 2.372$.

II. Finding a 4-clique using 3 languages

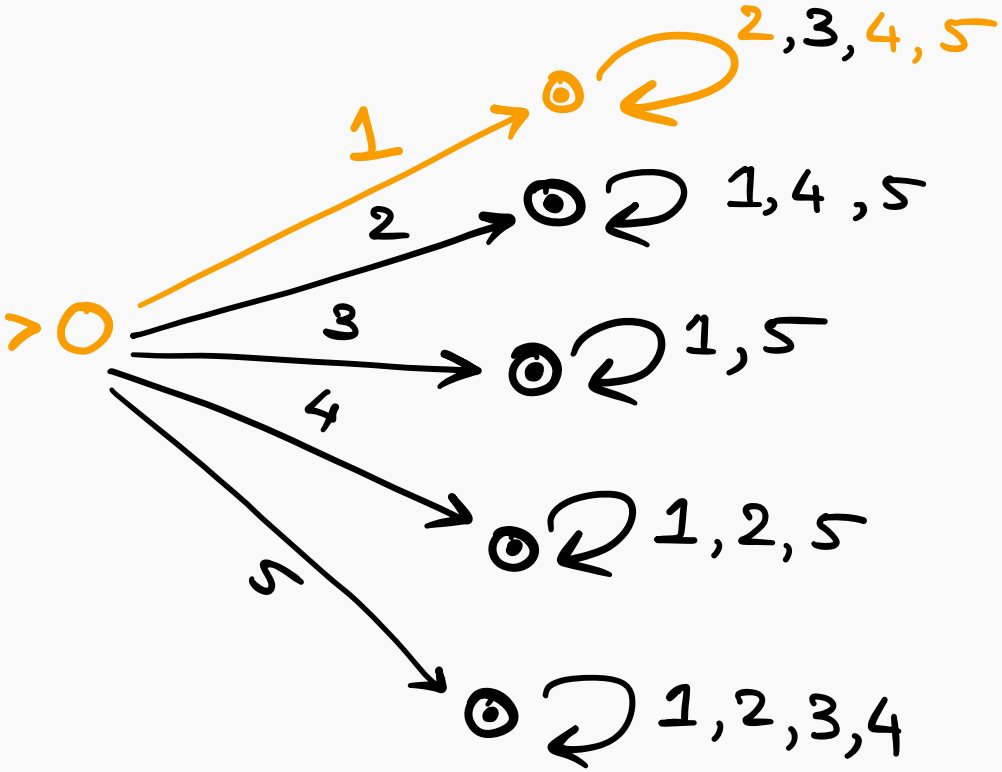
There is a 4 clique in G if:

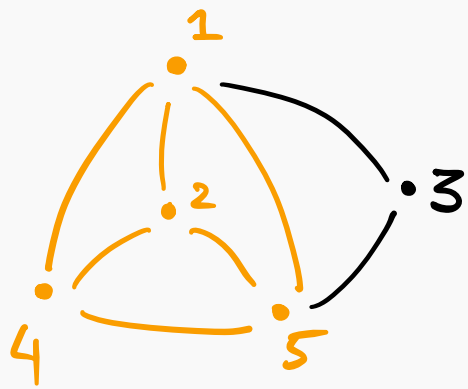
1. $\exists\ u_1, u_2, u_3, u_4$
2. u_1 is adjacent to u_2, u_3, u_4
3. u_2 is adjacent to u_3, u_4
4. u_3 is adjacent to u_4



II. The three NFAs

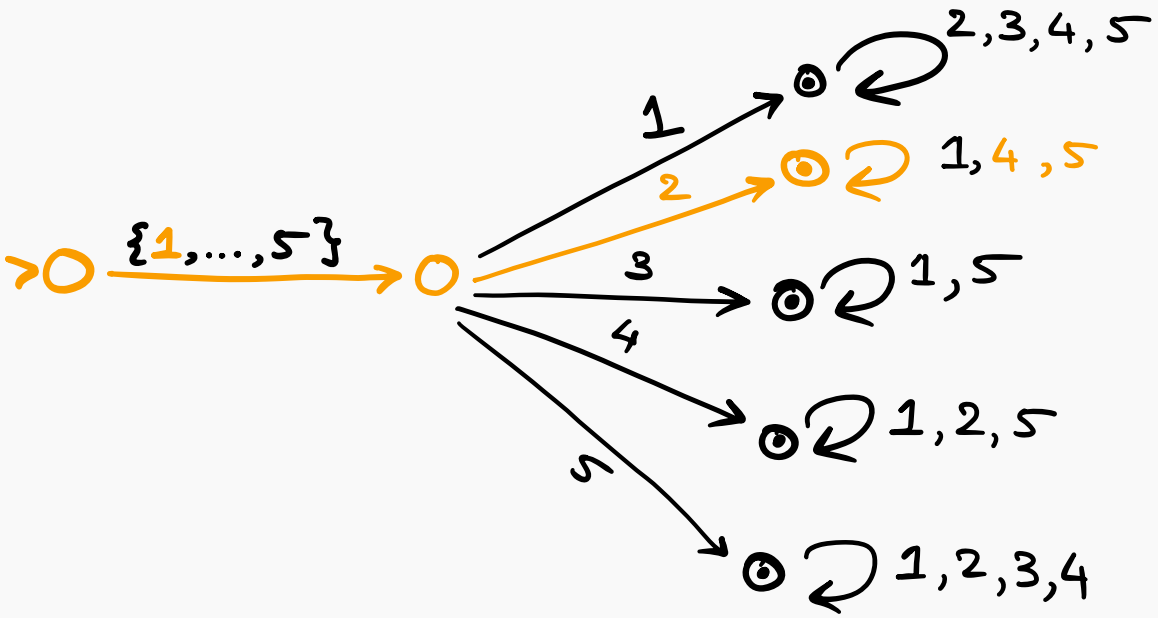
$$L(\mathcal{A}_1) = \bigcup_{u \in V} u \cdot (\mathrm{Neigh}(u))^*$$

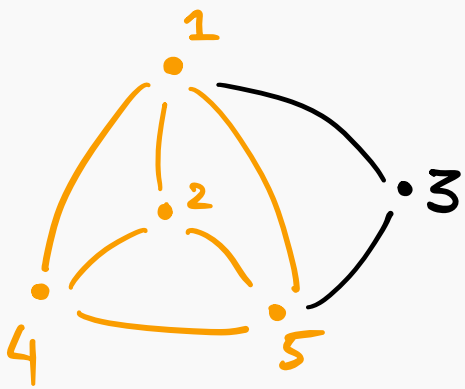




II. The three NFAs

$$L(\mathcal{A}_2) = V \cdot \left(\bigcup_{u \in V} u \cdot (\text{Neigh}(u))^* \right)$$





III. Conditional Optimality

Theorem (3)

If NFA k-IE has an $O((mn^{k-1})^{1-\epsilon})$ time algorithm, then $(k+1)$-Clique has an $O((n^{k+1})^{1-\epsilon})$ time algorithm.

III. Conditional Optimality

Theorem (3)

If NFA k-IE has an $O((mn^{k-1})^{1-\epsilon})$ time algorithm, then $(k+1)$-Clique has an $O((n^{k+1})^{1-\epsilon})$ time algorithm.

If NFA k-IE can be sped up without fast matrix multiplication, then so can k-Clique.

III. Conditional Optimality

Theorem (3)

If NFA k-IE has an $O((mn^{k-1})^{1-\epsilon})$ time algorithm, then $(k+1)$-Clique has an $O((n^{k+1})^{1-\epsilon})$ time algorithm.

If NFA k-IE can be sped up without fast matrix multiplication, then so can k-Clique.

Combinatorial k-Clique Hypothesis [Abboud, Backurs, Williams, 2018]

There is no $O((n^{k+1})^{1-\epsilon})$ time algorithm for $(k+1)$-Clique that avoids fast matrix multiplication, regardless of $\epsilon > 0$.

III. Conditional Optimality

Corollary (4)

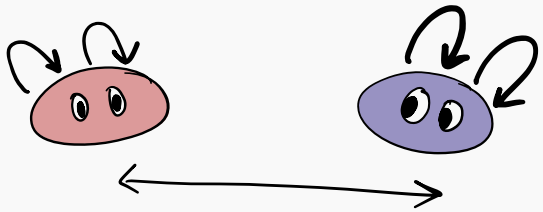
If NFA k-IE has an $O((mn^{k-1})^{1-\epsilon})$ time algorithm that avoids fast matrix multiplication, the Combinatorial k-Clique Hypothesis is false.

- It was known that triangle detection reduces to NFA 2-IE [Potechin and Shallit, 2020], even to the special case of NFA Acceptance.

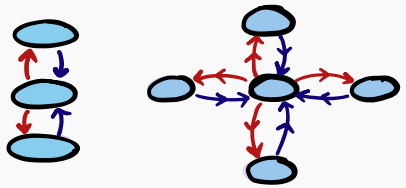
- It was known that triangle detection reduces to NFA 2-IE [Potechin and Shallit, 2020], even to the special case of NFA Acceptance.
- [Bringmann et al, 2024] conjecture that the NFA Acceptance problem, given an m transition NFA and a word w, already needs $m|w|$ time.

Plan

I NFA K-Intersection (Emptiness)



II Product Constructions



III Reduction From K-Clique

IV Certificates and Seth

0. What's better than an algorithm?

We now have a conditionally optimal algorithm for NFA k-IE!

0. What's better than an algorithm?

We now have a conditionally optimal algorithm for NFA k-IE!

Let's all implement it in (LASH, SPIN, Walnut, MATA, PathFinder) because it uses NFA intersection as a subprocedure.

0. What's better than an algorithm?

We now have a conditionally optimal algorithm for NFA k-IE!

Let's all implement it in (LASH, SPIN, Walnut, MATA, PathFinder) because it uses NFA intersection as a subprocedure.

But what if our implementation had a bug? How would we even notice?

I. A Certifying Algorithm

Certifying Algorithm (e.g. [McConnell et al 2010])

A certifying algorithm doesn't just output yes or no, it also hands you an independently verifiable certificate of correctness.

II. Certifying intersection non-emptiness (Warmup)

Input k NFAs $\mathcal{A}_1, \ldots, \mathcal{A}_k$

II. Certifying intersection non-emptiness (Warmup)

Input k NFAs $\mathcal{A}_1, \ldots, \mathcal{A}_k$

A certificate A word in $L(\mathcal{A}_1) \cap \cdots \cap L(\mathcal{A}_k)$

II. Certifying intersection non-emptiness (Warmup)

Input k NFAs $\mathcal{A}_1, \ldots, \mathcal{A}_k$

A certificate A word in $L(\mathcal{A}_1) \cap \cdots \cap L(\mathcal{A}_k)$

A very convenient certificate k accepting runs, one in each NFA, over the same short word.

II. Certifying intersection non-emptiness (Warmup)

Input k NFAs $\mathcal{A}_1, \ldots, \mathcal{A}_k$

A certificate A word in $L(\mathcal{A}_1) \cap \cdots \cap L(\mathcal{A}_k)$

A very convenient certificate k accepting runs, one in each NFA, over the same short word.

Verifier Verifies the runs are short, accepting, and all read the same word in $O(n^k)$ time.

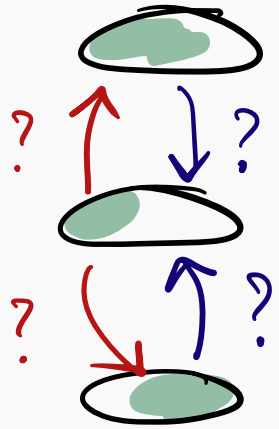
III. Certifying intersection emptiness

If $k = 1$, this is just non-reachability in a graph.

Input An NFA $\mathcal{A}$

Certificate A cut in the transition graph

Verifying a cut in the nodding product automaton involves enumerating all of its transitions.



III. Certifying intersection emptiness

If $k = 1$, this is just non-reachability in a graph.

Input An NFA $\mathcal{A}$

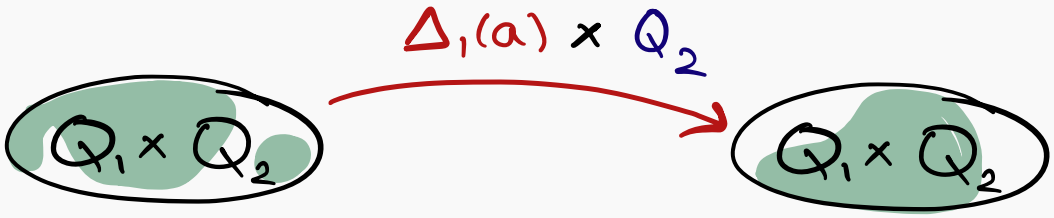
Certificate A cut in the transition graph

Verifying a cut in the nodding product automaton involves enumerating all of its transitions.

Hence, this takes $O(k \cdot mn^{k-1})$ time, same as solving NFA k-IE from scratch.

III. Idea: The shape of the nodding product

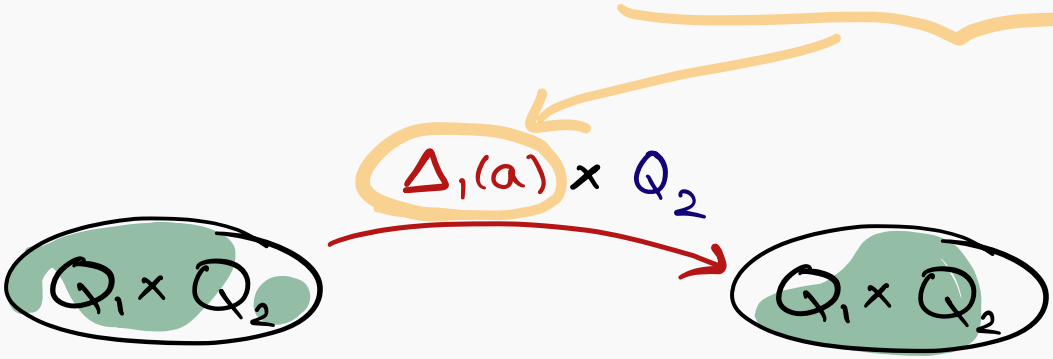
Transitions in the nodding product are grouped into volleys.



III. Idea: The shape of the nodding product

Transitions in the nodding product are grouped into volleys.

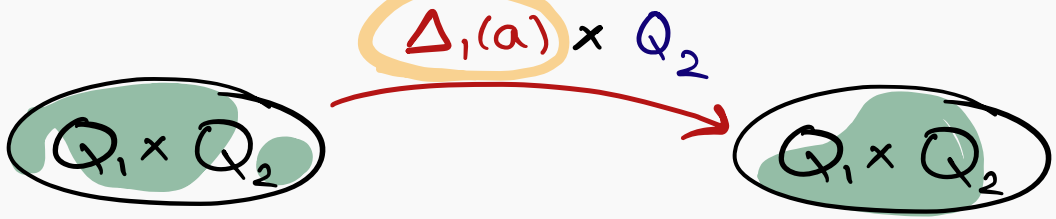
Each volley updates only one given component, by one letter's transitions.



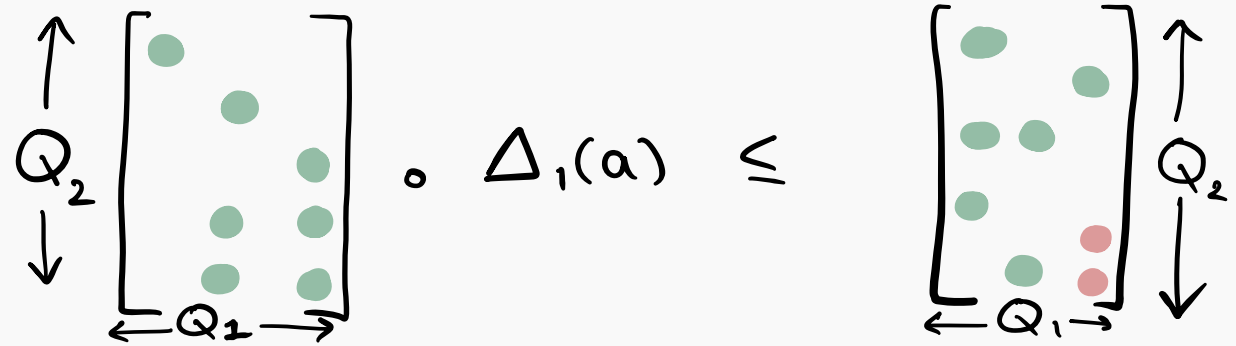
III. Idea: The shape of the nodding product

Transitions in the nodding product are grouped into volleys.

Each volley updates only one given component, by one letter's transitions.



Using matrices, verify that a volley does not leave the cut using one $n^{k-1} \times n$ by $n \times n$ matrix multiplication.



III. Certifying intersection emptiness

Theorem (5)

NFA k-IE has a certificate system that can be verified in time $O(k\ell \cdot n^{k+\omega-2})$.

III. Certifying intersection emptiness

Theorem (5)

NFA k-IE has a certificate system that can be verified in time $O(k\ell \cdot n^{k+\omega-2})$.

For sufficiently dense automata, this is faster than solving

NFA k-IE!

IV. SAT-based lower bounds

- P $\neq$ NP hypothesis: There is no $O(n^k)$-time algorithm for solving SAT

IV. SAT-based lower bounds

- P $\neq$ NP hypothesis: There is no $O(n^k)$-time algorithm for solving SAT
- Strong Exponential Time Hypothesis (SETH): There is no deterministic $1.999..9^n$-time algorithm for solving SAT

IV. SAT-based lower bounds

- P $\neq$ NP hypothesis: There is no $O(n^k)$-time algorithm for solving SAT
- Strong Exponential Time Hypothesis (SETH): There is no deterministic $1.999..9^n$-time algorithm for solving SAT
- Nondeterministic Strong Exponential Time Hypothesis (NSETH): There is no nondeterministic $1.999..9^n$-time algorithm for solving TAUT

IV. SETH for NFA k-IE

Theorem (Wehar, 2014)

An $O(n^{k-\epsilon})$ time algorithm for DFA k-IE would refute SETH.

IV. SETH for NFA k-IE

Theorem (Wehar, 2014)

An $O(n^{k-\epsilon})$ time algorithm for DFA k-IE would refute SETH.

Theorem (Carmosino et. al., 2016)

Whenever a problem has a certificate system that runs in time strictly below the best known upper bound, a tight SETH-based lower bound would contradict NSETH.

IV. A tight SETH lower bound is unlikely

Corollary (6)

An (mn^{k-1})-lower bound for NFA k-IE from SETH would contradict NSETH.

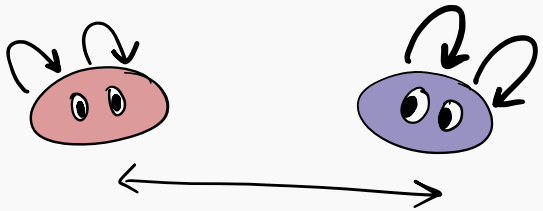
- Since there is an $\Omega(n^k)$ SETH lower bound [Wehar, 2014], a certificate system faster than $O(n^k)$ would also refute NSETH.

- Since there is an $\Omega(n^k)$ SETH lower bound [Wehar, 2014], a certificate system faster than $O(n^k)$ would also refute NSETH.

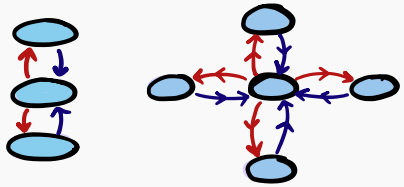
- Certifying language emptiness/non-acceptance quickly using fast matrix multiplication is also used in [Bringmann et al, 2024] and [Chistikov et al, 2022].

II. Summary

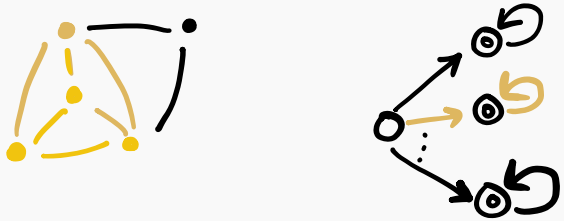
NFA K-Intersection (Emptiness)



Product Constructions



Reduction From k-Clique



Certificates and SETH