Chapter 2, Cont.

OMv Conjecture implies **OuMv** Conjecture

Danupon Nanongkai

KTH, Sweden

Notice:
Technique similar to APSP
→ Negative Triangle

ADFOCS 2018
Last edited: Aug. 15, 2018

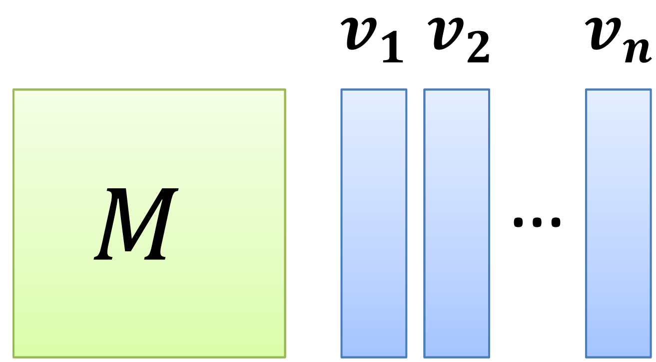
RECAP FROM LAST TIME

OMv Conjecture

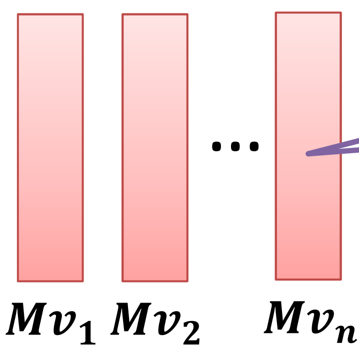
(**O**nline **M**atrix-**V**ector Multiplication) [Henzinger, Krinninger, N, Saranurak, STOC'15]

Input: $n \times n$ Boolean matrix **M**

Then: n Boolean vectors v_i



Output:



(OR,AND)-mult.
not (+,×)-mult.

Answer Mv_i before getting v_{i+1}

Conjecture: No algorithms with **total** time $O(n^{3-\epsilon})$

Current Best: $O(n^3/2^{\sqrt{\log n}})$ [Larsen-Williams SODA'17]

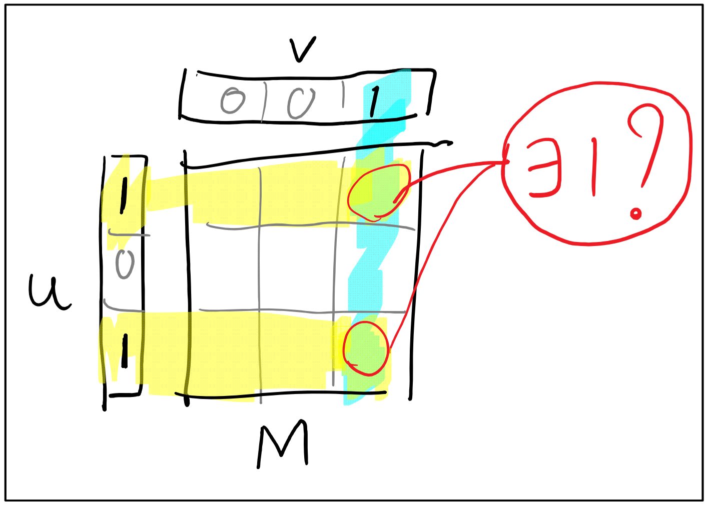
Example on board?

OuMv Conjecture (Matrix Form)

Input: $n \times n$ Boolean matrix **M**

Then: n **pairs** of Boolean vectors $(\boldsymbol{u_i}, \boldsymbol{v_i})$

Output: $u_i^T M v_i$



Answer $u_i^T M v_i$ before getting (u_{i+1}, v_{i+1})

Conjecture: No algorithms with **total** time $O(n^{3-\epsilon})$

even with polynomial time to process M!

Example on board?

Theorem 1: OMv conjecture holds with polynomial preprocessing time.

Details:

- OMv Conjecture (recall): No algorithms with **total** time $O(n^{3-\epsilon})$.
- OMv' Conjecture: No algorithms with **total** time $O\left(n^{3-\epsilon}\right)$, *even with polynomial time to process M!*

Claim: OMv Conjecture implies OMv' Conjecture

Idea: Partition M!

<u>Theorem 1</u>: OMv conjecture holds with polynomial preprocessing time.

<u>Main Idea:</u> Divide M into submatrices of small dimension n'

- Preprocessing time $\mathrm{poly}(\mathrm{n}') \ll n$^2 per element in M (when n' small enough)
- Time for each multiplication is still better than the trivial $(n')^3$ time if n' is not too small.

<u>Proof (sketched):</u>

1. Suppose algorithm $\boldsymbol{A}$ can solve OMv with $\mathbf{n^{10}}$ time for preprocessing M and $\mathrm{O}(\boldsymbol{n^{2.9}})$ time after that. We will construct an algorithm with $\boldsymbol{O}(\boldsymbol{n^{2.99}})$ time in total.
2. Divide M into submatrices each of dimension $n' \times n'$, where $\mathrm{n}' = n^{1/20}$.

<u>For each submatrix $\boldsymbol{M'}$:</u>

a. Preprocess in $(n')^{10} = n^{1/2}$ time.
   - There are at most n^2 submatrices → $\boldsymbol{O}(\boldsymbol{n^{2.5}})$ total preprocessing time

b. Handle every group of n' vectors v_i in $(n')^{2.9}$ time.
   - <u>Details</u>: “Roll-back” to the preprocessing stage after every n' pairs.
   - Time over n/n' groups and $\left(\frac{n}{n'}\right)^2$: $(n')^{2.9} \times \left(\frac{n}{n'}\right)^3 = n^{3-1/200}$

Claim 2: OMv’ conjecture implies OuMv conjecture

OMv' conjecture implies OuMv conjecture

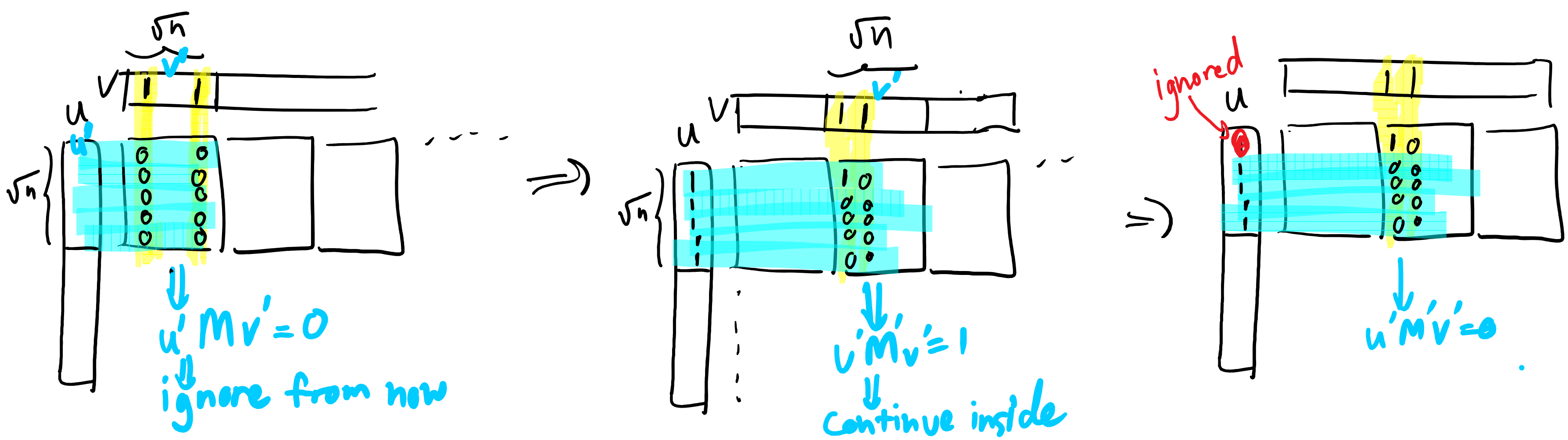
Idea: Suppose we can compute each uMv in $\boldsymbol{n^{1.9}}$ time. For each vector v, use u to **binary search** for i s.t. $(Mv)_i = 1$.

- Problem: To find all 1's in Mv needs to compute uMv up to n times → $n^{2.9}$ to compute one Mv.
- Fix: Partition M!

OMv' conjecture implies OuMv conjecture

<u>Algorithm Sketch</u>:

1. Divide M into submatrices each of dimension $n' \times n'$, where $\mathrm{n}' = n^{1/2}$.
2. To find $i \leq \sqrt{n}$ s.t. $(Mv)_i = 1$, compute $u'M'v'$ with $u' = (1, 1, \ldots)$ on each submatrix M' on top $\sqrt{n}$ rows.
3. $u'M'v' = 0$ → Ignore M' from now on
4. $u'M'v' = 1$ → Continue binary search within M'
5. Repeat with coordinate i ignored. Do the same for other rows.



Analysis

Algorithm Sketch:

1. Divide M into submatrices each of dimension $n' \times n'$, where $\mathrm{n}' = n^{1/2}$.
2. To find $i \leq \sqrt{n}$ s.t. $(Mv)_i = 1$, compute $u'M'v'$ with $u' = (1, 1, \ldots)$ on each submatrix M' on top $\sqrt{n}$ rows.
3. $u'M'v' = 0$ → Ignore M' from now on
4. $u'M'v' = 1$ → Continue binary search within M'
5. Repeat with coordinate i ignored. Do the same for other rows.

Time to compute each $\boldsymbol{Mv}$: Suppose $u'M'v'$ takes $(n')^{1.9}$.For each matrix M':

1. $u'M'v' = 0$ → Ignore M' from now on → Happens only once for each M', total time = $(\boldsymbol{n}')^{\mathbf{1.9}} \times \left(\frac{\boldsymbol{n}}{\boldsymbol{n}'}\right)^{\mathbf{2}} \ll \boldsymbol{n}^{\mathbf{2}}$
2. $u'M'v' = 1$ → Continue binary search within M' → Will find new coordinate, thus happens only once for each coordinate. Total time =$(\boldsymbol{n}')^{\mathbf{1.9}} \times \boldsymbol{n} \ll \boldsymbol{n}^{\mathbf{2}}$.

Questions?

Thanks to co-authors:

Sayan Bhattacharya, Jan van den Brand, Deeparnab Chakraborty, Sebastian Forster, Monika Henzinger, Christian Wulff-Nilsen, Thatchaphol Saranurak

This project has received funding from the *European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme* under grant agreement No 715672