Joint work with Alkida Balliu, Sebastian Brandt, Juho Hirvonen, Dennis Olivetti, Mikaël Rabie

Best paper award at FOCS 2019

Lower bounds for maximal matchings and maximal independent sets

Jukka Suomela · Aalto University · Finland
Two classical graph problems

Maximal matching

- **matching** = set of non-adjacent edges
- **maximal** = not a strict subset of another matching

Maximal independent set

- **independent set** = set of non-adjacent nodes
- **maximal** = not a strict subset of another independent set
Two classical graph problems

Maximal matching

Maximal independent set

Trivial linear-time centralized, sequential algorithm: add edges/nodes until stuck

*How easy is it to solve these problems in a distributed setting?*
Should I output 0 or 1?
Should I output 0 or 1?

Should I output 0 or 1?
Should I output 0 or 1?
Should I output 0 or 1?

How far do I need to see to safely choose my own output?
But what if everyone sees the same local neighborhood?

Should I output 0 or 1?
Deterministic distributed algorithms: we assume that nodes are labeled with unique identifiers.
Randomized distributed algorithms: we assume that nodes are labeled with a *stream of random bits*
How far do you need to see?

• More formally: time complexity in the **LOCAL model** of distributed computing

• Two equivalent perspectives:
  • *how far* does a node need to see to pick its own part of the solution?
  • *how many communication rounds* are needed in a message-passing system until all nodes can stop and announce their own outputs?

• Worst-case setting:
  • worst-case input graph
  • worst-case assignment of unique identifiers
\( n \) = number of nodes
\( \Delta \) = maximum degree
Old news: $O(\Delta + \log^* n)$

Our result: **this is tight!**
This project started around 2011...
### State of the art in the early 2010s

**Four key problems**
- that have been studied actively since the 1980s

**Trivial to solve with centralized sequential algorithms**

**All of these are “symmetry-breaking” problems**
- adjacent nodes/edges in the middle of a regular graph need to produce different outputs

<table>
<thead>
<tr>
<th>maximal independent set</th>
<th>maximal matching</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Δ+1)-vertex coloring</td>
<td>(2Δ−1)-edge coloring</td>
</tr>
</tbody>
</table>

- maximal independent set
- maximal matching
- (Δ+1)-vertex coloring
- (2Δ−1)-edge coloring
State of the art in the early 2010s

• **Lower bounds:**

• **Upper bounds:**

<table>
<thead>
<tr>
<th>maximal independent set</th>
<th>maximal matching</th>
</tr>
</thead>
<tbody>
<tr>
<td>(Δ+1)-vertex coloring</td>
<td>(2Δ−1)-edge coloring</td>
</tr>
</tbody>
</table>
State of the art in the early 2010s

- Algorithms for solving each of these problems in $O(\Delta + \log^* n)$ rounds

- Is this the best one can do, and why?

- Well-known that $O(\Delta) + o(\log^* n)$ is not possible
  - holds both for deterministic and randomized algorithms

- What about $o(\Delta) + O(\log^* n)$ ???
How to make sense of $O(\Delta + \log^* n)$?

- Why $O(\Delta + \log^* n)$?
  - $O(\log^* n)$: “symmetry-breaking”, adjacent nodes do different things
  - $O(\Delta)$: ???

- How to study dependency on $n$ in isolation?
  - just look at bounded-degree graphs, let $\Delta = O(1)$

- How to study dependency on $\Delta$ in isolation?
  - you can’t set $n = O(1)$ and see what happens…
  - but maybe we can eliminate “symmetry-breaking” concerns?
Eliminate “$O(\log^* n)$” part

• Could we find simple special cases of these problems that would be solvable in $O(\Delta)$ time, independently of $n$?

• Yes! Examples:
  • maximal matching: $O(\Delta + \log^* n)$
  • maximal fractional matching: $O(\Delta)$
  • maximal matching in bipartite graphs: $O(\Delta)$
  • maximal matching in edge-colored graphs: $O(\Delta)$

• Could we first prove a lower bound for one of these?
  • and if so, would it help to understand the general case?

Let’s look at this in more detail...
computer network with port numbering

bipartite, 2-colored graph

$\Delta$-regular (here $\Delta = 3$)

output: maximal matching
Very simple algorithm

unmatched white nodes:
send *proposal* to port 1
Very simple algorithm

unmatched white nodes: send *proposal* to port 1

black nodes: accept the first proposal you get, reject everything else (break ties with port numbers)
Very simple algorithm

unmatched white nodes: send *proposal* to port 1

black nodes: *accept* the first proposal you get, *reject* everything else
(break ties with port numbers)
Very simple algorithm

unmatched white nodes:
send *proposal* to port 2
Very simple algorithm

unmatched white nodes: send *proposal* to port 2

black nodes: *accept* the first proposal you get, *reject* everything else (break ties with port numbers)
**Very simple algorithm**

**unmatched white nodes:**
send *proposal* to port 2

**black nodes:**
*accept* the first proposal you get, *reject* everything else
(break ties with port numbers)
Very simple algorithm

unmatched white nodes:

send *proposal* to port 3
Very simple algorithm

unmatched white nodes:
send \textit{proposal} to port 3

black nodes:
\textit{accept} the first proposal you get, \textit{reject} everything else
(break ties with port numbers)
Very simple algorithm

unmatched white nodes:
send *proposal* to port 3

black nodes:
*accept* the first proposal you get, *reject* everything else
(break ties with port numbers)
Very simple algorithm

Finds a maximal matching in $O(\Delta)$ communication rounds

Note: running time does not depend on $n$
Bipartite maximal matching

• Maximal matching in 2-colored $\Delta$-regular graphs

• Simple algorithm: $O(\Delta)$ rounds, independently of $n$
  = if each node sees its radius-$O(\Delta)$ neighborhood, it can choose its own part of the solution (whether it is matched and with whom)

• Is this optimal?
  • $o(\Delta)$ rounds?
  • $O(\log \Delta)$ rounds??
  • 4 rounds??
Bipartite maximal matching

• Seemingly simple toy problem
  • no need for randomness, unique identifiers

• Promising starting point?
  • hypothesis: “$O(\Delta)$” in the proposal algorithm is there “for the same reason” as in much more complicated $O(\Delta + \log^* n)$-time algorithms
Progress since 2011

• **PODC 2012**: maximal matching not possible in $o(\Delta)$ time in the “edge-coloring model”
  • doesn’t tell anything about **bipartite maximal matching**

• **PODC 2014**: maximal *fractional* matching not possible in $o(\Delta)$ time in the usual LOCAL model
  • doesn’t tell anything about **bipartite maximal matching**

We had a lower-bound technique, but it couldn’t handle 2-colored graphs
Progress since 2011

• In the meantime, there were new upper bounds!

• **Barenboim, PODC 2015:**
  
  - *(Δ+1)*-vertex coloring and *(2Δ−1)*-edge coloring
  
  in \(O(Δ^{3/4} + \log* n)\) time

• Could it be the case that also maximal matching and maximal independent set are solvable in \(o(Δ)\) time using similar techniques??
Progress since 2011

• We kept working on the \textit{bipartite maximal matching} problem

• And it certainly wasn’t a secret!
  • e.g. in ADGA 2014 I gave a talk outlining the whole research program: solve the complexity of bipartite maximal matching, it would probably tell us something about all these problems
  • every time we had visitors, I annoyed them with questions about bipartite maximal matchings

• But zero new progress, until late 2018
  • or so we thought...

Summary

• Distributed time complexity, LOCAL model
• $O(\log^* n)$: “symmetry breaking”, OK
• $O(\Delta)$: “local coordination”, poorly understood
• Maximal \textit{fractional} matching solved, next step: \textit{bipartite} maximal matching

• Given:
  • algorithm $A_0$ solves $3$-coloring in $T = o(\log^* n)$ rounds

• We construct:
  • algorithm $A_1$ solves $2^3$-coloring in $T - 1$ rounds
  • algorithm $A_2$ solves $2^{2^3}$-coloring in $T - 2$ rounds
  • algorithm $A_3$ solves $2^{2^{2^3}}$-coloring in $T - 3$ rounds
  ...
  • algorithm $A_T$ solves $o(n)$-coloring in 0 rounds

• But $o(n)$-coloring is nontrivial, so $A_0$ cannot exist
Brandt et al. (2016): sinkless orientation

• Given:
  • algorithm $A_0$ solves sinkless orientation in $T = o(\log n)$ rounds

• We construct:
  • algorithm $A_1$ solves sinkless coloring in $T - 1$ rounds
  • algorithm $A_2$ solves sinkless orientation in $T - 2$ rounds
  • algorithm $A_3$ solves sinkless coloring in $T - 3$ rounds
    ...
  • algorithm $A_T$ solves sinkless orientation in 0 rounds

• But sinkless orientation is nontrivial, so $A_0$ cannot exist
Brandt (2019): this can be automated

• Always possible for any graph problem $P_0$ that is “locally verifiable”

• If problem $P_0$ has complexity $T$, we can always find in a mechanical manner problem $P_1$ that has complexity $T - 1$
  • holds for tree-like neighborhoods (e.g. high-girth graphs)

• This technique is nowadays known as “round elimination”
Late 2018 research meeting...

- Sebastian Brandt told us about his new lower bound technique that he had applied to weak 2-coloring
- We invited Sebastian for a 4-day visit so that he could present his proof
- He started by presenting the general round elimination technique, and before he could continue, we had already got sidetracked into discussing bipartite maximal matching...
- Could we use round elimination to prove a lower bound?
Late 2018 research meeting...

• Challenge when you try to apply round elimination:
  • you start with $P_0 =$ bipartite maximal matching
  • you apply round elimination for a few steps
  • $P_1 =$ something that still makes some sense
  ...
  • $P_3 =$ a complicated mess that fills two whiteboards, and nobody has any idea what the problem is about — how to continue?

• We need to discover a family of simpler problems $Q_0, Q_1, Q_2, ...$
  • $Q_i$ is a relaxation of $P_i$ (a lower bound for $Q_i$ gives a lower bound for $P_i$)
  • $Q_i$ isn’t too easy to solve (there exists a nontrivial lower bound for $Q_i$)
Late 2018 research meeting...

• Given a complicated graph problem with lots of possible output labels, one can try to \textit{simplify it in systematic ways} by e.g. merging labels

• But this is very slow (and very error-prone) to do by hand
  • round elimination is mechanical, but lots of work

• \textit{Dennis Olivetti} implemented \textit{round elimination as a computer program} in one evening
  • we could start to quickly explore what happens if we try out different simplification schemes — \textit{this led to the breakthrough}!
Main results

Maximal matching and maximal independent set cannot be solved in

- $o(\Delta + \log \log n / \log \log \log n)$ rounds with randomized algorithms
- $o(\Delta + \log n / \log \log n)$ rounds with deterministic algorithms
Latest news

• The complexity of both maximal matching and maximal independent set now well-understood, thanks to the network decomposition algorithm by Rozhoň & Ghaffari (STOC 2020)

• “Round Eliminator” program freely available online, with a web user interface
  • github.com/olidennis/round-eliminator

• Still wide open: complexity of graph coloring
  • can you find a $(\Delta+1)$-vertex coloring in $O(\log \Delta + \log^* n)$ rounds??
Representation for maximal matchings

white nodes “active”
output one of these:
· $1 \times M$ and $(\Delta - 1) \times O$
· $\Delta \times P$

black nodes “passive”
accept one of these:
· $1 \times M$ and $(\Delta - 1) \times \{P, O\}$
· $\Delta \times O$

$M$ = “matched”
$P$ = “pointer to matched”
$O$ = “other”
Representation for maximal matchings

white nodes “active”
output one of these:
\[ 1 \times M \text{ and } (\Delta - 1) \times O \]
\[ \Delta \times P \]

black nodes “passive”
accept one of these:
\[ 1 \times M \text{ and } (\Delta - 1) \times \{P, 0\} \]
\[ \Delta \times O \]

\[ W = M O^{\Delta - 1} \mid P^\Delta \]

\[ B = M [PO]^{\Delta - 1} \mid O^\Delta \]
Parameterized problem family

\[ W = M O^{\Delta - 1} \mid P^\Delta, \]
\[ B = M [P O]^{\Delta - 1} \mid O^\Delta \]

\[ W_\Delta(x, y) = \left( M O^{d-1} \mid P^d \right) O^y X^x, \]
\[ B_\Delta(x, y) = \left( [M X] [P O X]^{d-1} \mid [O X]^d \right) [P O X]^y [M P O X]^x, \]
\[ d = \Delta - x - y \]
Main lemma

• Given: $A$ solves $P(x, y)$ in $T$ rounds
• We can construct: $A'$ solves $P(x + 1, y + x)$ in $T - 1$ rounds

\[
W_\Delta(x, y) = \left( \begin{array}{c|c} MO^{d-1} & P^d \\ \end{array} \right) O^y X^x,
\]

\[
B_\Delta(x, y) = \left( \begin{array}{c|c} [MX][POX]^{d-1} & [OX]^d \\ \end{array} \right) [POX]^y [MPOX]^x,
\]

\[
d = \Delta - x - y
\]
Putting things together

Maximal matching in $o(\Delta)$ rounds

$\rightarrow$ “$\Delta^{1/2}$ matching” in $o(\Delta^{1/2})$ rounds

$\rightarrow$ $P(\Delta^{1/2}, 0)$ in $o(\Delta^{1/2})$ rounds

$\rightarrow$ $P(O(\Delta^{1/2}), o(\Delta))$ in 0 rounds

$\rightarrow$ contradiction

What we really care about

k-matching: select at most $k$ edges per node

Apply speedup simulation $o(\Delta^{1/2})$ times
Putting things together

- Basic version:
  - deterministic lower bound, *port-numbering model*

- Analyze what happens to local failure probability:
  - *randomized* lower bound, port-numbering model

- With randomness you can construct unique identifiers w.h.p.:
  - randomized lower bound, *LOCAL model*

- Fast deterministic $\rightarrow$ faster deterministic
  - stronger *deterministic* lower bound, LOCAL model

Proof technique does not work directly with unique IDs
Maximal matching, LOCAL model, $O(f(\Delta) + g(n))$

Algorithms:
- deterministic
- randomized

Lower bounds:
- deterministic
- randomized

Round elimination

Given: white algorithm $A$ that runs in $T = 2$ rounds

- $v_1$ in $A$ sees $U$ and $D_1$

Construct: black algorithm $A'$ that runs in $T - 1 = 1$ rounds

- $u$ in $A'$ only sees $U$

$A'$: what is the set of possible outputs of $A$ for edge ${u, v_1}$ over all possible inputs in $D_1$?