Computing Large Matchings Fast

Ignaz Rutter  Alexander Wolff

Karlsruhe University  TU Eindhoven
Overview

1. Introduction
   - Definitions and known results
   - Warm-up: simple algorithms for maxdeg-\( k \) graphs

2. Graphs with maxdeg 3
   - 3-regular graphs
   - Graphs with maxdeg 3

3. The missing algorithm and maximum matchings
   - 3-connected planar graphs
   - Graphs with bounded-degree block trees
Overview

1 Introduction
   - Definitions and known results
   - Warm-up: simple algorithms for maxdeg-$k$ graphs

2 Graphs with maxdeg 3
   - 3-regular graphs
   - Graphs with maxdeg 3

3 The missing algorithm and maximum matchings
   - 3-connected planar graphs
   - Graphs with bounded-degree block trees
Matching

Given an undirected graph $G = (V, E)$...
Matching

Given an undirected graph \( G = (V, E) \)...
...a matching is a set \( M \) of independent edges.
Matching

A free vertex is a vertex that is not incident to an edge of $M$. 
Matching

An *augmenting path* is a path that alternates between matching and non-matching edges, and starts and ends at different free vertices.
Matching

An *augmenting path* is a path that alternates between matching and non-matching edges, and starts and ends at different free vertices.
A *maximum matching* is a matching of maximum cardinality.
Matching

Theorem (Berge)

A matching is maximum $\iff$ there is no augmenting path.
Known results

Let $G = (V, E)$ and $n = |V|$, $m = |E|$.

- Maximum matchings take $O(\sqrt{n} \cdot m)$ time. [Micali, Vazirani ’80]
- If $m = \Theta(n)$: $O(n^{1.5})$ running time,
  e.g., graphs with constant maxdeg or planar graphs.
Known results

Let $G = (V, E)$ and $n = |V|$, $m = |E|$.

- Maximum matchings take $O(\sqrt{n} \cdot m)$ time. \[\text{[Micali, Vazirani '80]}\]
- If $m = \Theta(n)$: $O(n^{1.5})$ running time, e.g., graphs with constant maxdeg or planar graphs.
- Algorithms based on fast matrix multiplication:
  - dense graphs: $O(n^{2.38})$ time \[\text{[Mucha, Sankowski '04]}\]
  - graphs of bounded genus: $O(n^{1.19})$ time \[\text{[Yuster, Zwick SODA'07]}\]
  - $H$-minor free graphs: $O(n^{1.32})$ time

---

Ignaz Rutter and Alexander Wolff

Computing Large Matchings Fast
Known results

Let $G = (V, E)$ and $n = |V|, m = |E|$.

- Maximum matchings take $O(\sqrt{n} \cdot m)$ time. [Micali, Vazirani ’80]
- If $m = \Theta(n)$: $O(n^{1.5})$ running time, e.g., graphs with constant maxdeg or planar graphs.
- Algorithms based on fast matrix multiplication:
  - dense graphs: $O(n^{2.38})$ time [Mucha, Sankowski ’04]
  - graphs of bounded genus: $O(n^{1.19})$ time [Yuster, Zwick SODA’07]
  - $H$-minor free graphs: $O(n^{1.32})$ time
- LEDA and Boost: $O(nm^{\omega(n, m)})$ time, based on repeatedly finding augmenting paths. [Tarjan ’83]
Known results

Results on the *existence* of matchings in certain graph classes.
[Biedl, Demaine, Duncan, Fleischer, Kobourov, ’04]

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected, planar</td>
<td>$\frac{n+4}{3}$</td>
<td>$\frac{2n+4-l_4}{4}$</td>
</tr>
<tr>
<td>maxdeg 3</td>
<td>$\frac{n-1}{3}$</td>
<td>$\frac{3n-n_2-2l_2}{6}$</td>
</tr>
<tr>
<td>3-regular</td>
<td>$\frac{4n-1}{9}$</td>
<td>$\frac{3n-2l_2}{6}$</td>
</tr>
</tbody>
</table>

There are linear-time reductions:
-Biedl SODA’01
-max. matchings in planar graphs → in triangulated planar graphs
-max. matchings in general graphs → in 3-regular graphs
Known results

Results on the *existence* of matchings in certain graph classes.

[Biedl, Demaine, Duncan, Fleischer, Kobourov, ’04]

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected, planar</td>
<td>(\frac{n+4}{3})</td>
<td>(\frac{2n+4-l_4}{4})</td>
</tr>
<tr>
<td>maxdeg 3</td>
<td>(\frac{n-1}{3})</td>
<td>(\frac{3n-n_2-2l_2}{6})</td>
</tr>
<tr>
<td>3-regular</td>
<td>(\frac{4n-1}{9})</td>
<td>(\frac{3n-2l_2}{6})</td>
</tr>
</tbody>
</table>

There are linear-time reductions:

- max. matchings in planar graphs $\rightarrow$ in *triangulated planar graphs*
- max. matchings in general graphs $\rightarrow$ in *3-regular graphs*
Our results

We present algorithms that
- are relatively simple,
- run in $O(n \ polylog \ n)$ time,
- implement all (but one) of the bounds of Biedl et al. and thus
- give good guarantees on the size of the computed matchings.
Overview

1. Introduction
   - Definitions and known results
   - Warm-up: simple algorithms for maxdeg-$k$ graphs

2. Graphs with maxdeg 3
   - 3-regular graphs
   - Graphs with maxdeg 3

3. The missing algorithm and maximum matchings
   - 3-connected planar graphs
   - Graphs with bounded-degree block trees
Maximum matchings in trees

Strategy PICKLEAFEDGES:
As long as the graph has a leaf (i.e., a vertex of degree 1)
- Pick an arbitrary leaf $u$ and match it to its parent $v$.
- Remove $u$ and $v$ from the graph.
Maximum matchings in trees

Strategy PICKLEAF EDGES:
As long as the graph has a leaf (i.e., a vertex of degree 1)

- Pick an arbitrary leaf $u$ and match it to its parent $v$.
- Remove $u$ and $v$ from the graph.

This computes a maximum matching in a tree.
Maximum matchings in trees

What is known about $|M|$?
Maximum matchings in trees

What is known about $|M|$?

Bound maxdeg by $k$: 

$$|M| \geq \frac{m}{k} = \frac{n - 1}{k}$$
From trees to graphs

Theorem

A tree with maxdeg \( k \) has a matching of size at least \( (n - 1)/k \). Such a matching can be computed in linear time.
From trees to graphs

**Theorem**

A tree with maxdeg $k$ has a matching of size at least $(n - 1)/k$. Such a matching can be computed in linear time.

**Corollary**

**maxdeg-3-graphs:** $|\text{matching}| \geq (n - 1)/3$ in $O(n)$ time.
From trees to graphs

**Theorem**

A tree with maxdeg $k$ has a matching of size at least $(n - 1)/k$. Such a matching can be computed in linear time.

**Corollary**

maxdeg-$3$-graphs: $|\text{matching}| \geq (n - 1)/3$ in $O(n)$ time.

**Corollary**

3-connected planar graph: $|\text{matching}| \geq (n - 1)/3$ in $O(n)$ time.
From trees to graphs

**Theorem**

A tree with maxdeg $k$ has a matching of size at least $(n - 1)/k$. Such a matching can be computed in linear time.

**Corollary**

maxdeg-3-graphs: $|\text{matching}| \geq (n - 1)/3$ in $O(n)$ time.

**Corollary**

3-connected planar graph: $|\text{matching}| \geq (n - 1)/3$ in $O(n)$ time.

**Proof**: A spanning tree with maxdeg 3 exists and can be computed in $O(n)$ time. 

[Barnette ’66]  
[Czumaj, Strothmann ’97]
## Overview

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected + planar</td>
<td>( \frac{n+4}{3} )</td>
<td>( \frac{2n+4-l_4}{4} )</td>
</tr>
<tr>
<td>max-deg 3</td>
<td>( \frac{n-1}{3} )</td>
<td>( \frac{3n-n_2-2l_2}{6} )</td>
</tr>
<tr>
<td>3-regular</td>
<td>( \frac{4n-1}{9} )</td>
<td>( \frac{3n-2l_2}{6} )</td>
</tr>
</tbody>
</table>
## Overview

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected + planar</td>
<td>$\frac{n+4}{3}$</td>
<td>$\frac{2n+4-\ell_4}{4}$</td>
</tr>
<tr>
<td>max-deg 3</td>
<td>$\frac{n-1}{3}$</td>
<td>$\frac{3n-n_2-2\ell_2}{6}$</td>
</tr>
<tr>
<td>3-regular</td>
<td>$\frac{4n-1}{9}$</td>
<td>$\frac{3n-2\ell_2}{6}$</td>
</tr>
</tbody>
</table>

An augmenting path can be computed in $O(n)$ time. [Tarjan '83]
Overview

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected + planar</td>
<td>$\frac{n+4}{3}$</td>
<td>$\frac{2n+4-\ell_4}{4}$</td>
</tr>
<tr>
<td>max-deg 3</td>
<td>$\frac{n-1}{3}$</td>
<td>$\sqrt{\frac{3n-n_2-2\ell_2}{6}}$</td>
</tr>
<tr>
<td>3-regular</td>
<td>$\frac{4n-1}{9}$</td>
<td>$\frac{3n-2\ell_2}{6}$</td>
</tr>
</tbody>
</table>

An augmenting path can be computed in $O(n)$ time.  

[Tarjan ’83]
## Overview

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected + planar</td>
<td>$\frac{n+4}{3}$ ✓</td>
<td>$\frac{2n+4-\ell_4}{4}$</td>
</tr>
<tr>
<td>max-deg 3</td>
<td>$\frac{n-1}{3}$ ✓</td>
<td>$\frac{3n-n_2-2\ell_2}{6}$</td>
</tr>
<tr>
<td>3-regular</td>
<td>$\frac{4n-1}{9}$</td>
<td>$\frac{3n-2\ell_2}{6}$</td>
</tr>
</tbody>
</table>

An augmenting path can be computed in $O(n)$ time. [Tarjan '83]
Overview

1 Introduction
- Definitions and known results
- Warm-up: simple algorithms for maxdeg-\(k\) graphs

2 Graphs with maxdeg 3
- 3-regular graphs
- Graphs with maxdeg 3

3 The missing algorithm and maximum matchings
- 3-connected planar graphs
- Graphs with bounded-degree block trees
Bridge
Introduction
Graphs with maxdeg 3
The missing algorithm and maximum matchings

3-regular graphs
Graphs with maxdeg 3

Bridge
2-block tree
2-block tree
2-block tree

\[ \ell_2 = 3 \]
3-regular graphs whose 2-block tree is a path

**Theorem (Petersen, 1891)**

*Every 3-regular graph whose 2-block tree is a path has a perfect matching.*
3-regular graphs whose 2-block tree is a path

Theorem (Petersen, 1891)

*Every 3-regular graph whose 2-block tree is a path has a perfect matching.*

Theorem (Biedl, Bose, Demaine, Lubiw, ’01)

*S*uch a matching can be computed in $O(n \log^4 n)$ time.
Arbitrary 3-regular graphs

Biedl et al.: Every 3-regular graph whose 2-block tree has $\ell_2$ leaves has a matching of size at least $(3n - 2\ell_2)/6$ ...

**Theorem**

... such that every free vertex is incident to a bridge.
Arbitrary 3-regular graphs

Biedl et al.: Every 3-regular graph whose 2-block tree has $\ell_2$ leaves
has a matching of size at least $(3n - 2\ell_2)/6$ ...

**Theorem**

... such that every free vertex is incident to a bridge.

- constructive proof: induction on $\ell_2$
- known: theorem holds for $\ell_2 = 1, 2$
- treat cases $\ell_2 = 3, 4$ separately
Arbitrary 3-regular graphs

Biedl et al.: Every 3-regular graph whose 2-block tree has $\ell_2$ leaves has a matching of size at least $(3n - 2\ell_2)/6$ ...

Theorem

... such that every free vertex is incident to a bridge.

- constructive proof: induction on $\ell_2$
- known: theorem holds for $\ell_2 = 1, 2$
- treat cases $\ell_2 = 3, 4$ separately

matching size $(3n - 2\ell_2)/6$:

$\Rightarrow (3n - 2\ell_2)/3 = n - 2\ell_2/3$ matched vertices

$\Rightarrow 2$ free vertices for every 3 leaves of the 2-block tree
Graphs with maxdeg 3
The missing algorithm and maximum matchings

Ignaz Rutter and Alexander Wolff
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Case $\ell_2 \geq 5$: Cutting leaves
Repairing the cuts

\[ 3 \times \]

[Diagram of a graph with a region labeled MC and three bridges highlighted]

**3-regular graphs**

Graphs with maxdeg 3

The missing algorithm and maximum matchings

Ignaz Rutter and Alexander Wolff

Computing Large Matchings Fast
Repairing the cuts

Compute matchings in all four components. $v$ is not incident to a bridge and hence is not free. Add one of the bridges. All free vertices are incident to a bridge.

$$\ell_2(MC) = \ell_2(G) - 3.$$  

$\#_{\text{free vertices}} G = \#_{\text{free vertices}} MC + 3 - 1 = \#_{\text{free vertices}} MC + 2.$
Repairing the cuts

- Compute matchings in all four components.
Repairing the cuts

- Compute matchings in all four components.

\[ \ell_2(MC) = \ell_2(G) - 3. \]
\[ \# \text{freevertices}_G = \]
Repairing the cuts

- Compute matchings in all four components.

\[ \ell_2(MC) = \ell_2(G) - 3. \]

\[ \#\text{freevertices}_G = \#\text{freevertices}_{MC} \]
Repairing the cuts

- Compute matchings in all four components.

\[ \ell_2(MC) = \ell_2(G) - 3. \]
\[ \# freevertices_G = \# freevertices_{MC} + 3 \]
Repairing the cuts

- Compute matchings in all four components.
- \( v \) is not incident to a bridge and hence is *not* free.

\[
\ell_2(MC) = \ell_2(G) - 3.
\]

\[
\# \text{freevertices}_G = \# \text{freevertices}_{MC} + 3
\]
Repairing the cuts

- Compute matchings in all four components.
- \( v \) is not incident to a bridge and hence is \textit{not} free.

\[ \ell_2(MC) = \ell_2(G) - 3. \]
\[ \#\text{freevertices}_G = \#\text{freevertices}_{MC} + 3 \]
Repairing the cuts

- Compute matchings in all four components.
- $v$ is not incident to a bridge and hence is \textit{not} free.
- Add one of the bridges.

\[ \ell_2(MC) = \ell_2(G) - 3. \]
\[ \#\text{freevertices}_G = \#\text{freevertices}_{MC} + 3 - 1 \]
Repairing the cuts

- Compute matchings in all four components.
- \( v \) is not incident to a bridge and hence is \emph{not} free.
- Add one of the bridges.

\[ \ell_2(MC) = \ell_2(G) - 3. \]

\[ \#\text{freevertices}_G = \#\text{freevertices}_{MC} + 3 - 1 = \#\text{freevertices}_{MC} + 2. \]
Repairing the cuts

- Compute matchings in all four components.
- \( v \) is not incident to a bridge and hence is *not* free.
- Add one of the bridges.
- All free vertices are incident to a bridge.

\[
\ell_2(MC) = \ell_2(G) - 3.
#freevertices_G = #freevertices_{MC} + 3 - 1 = #freevertices_{MC} + 2.
\]
Overview

1. Introduction
   - Definitions and known results
   - Warm-up: simple algorithms for maxdeg-$k$ graphs

2. Graphs with maxdeg 3
   - 3-regular graphs
   - Graphs with maxdeg 3

3. The missing algorithm and maximum matchings
   - 3-connected planar graphs
   - Graphs with bounded-degree block trees
Graphs with maxdeg 3

Add dummy edges and vertices to make graph 3-regular...
Graphs with maxdeg 3

Add dummy edges and vertices to make graph 3-regular...

... apply previous algorithm, and remove dummies.
### Overview

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected + planar</td>
<td>$\frac{n+4}{3}$ ✓</td>
<td>$\frac{2n+4-\ell_4}{4}$</td>
</tr>
<tr>
<td>max-deg 3</td>
<td>$\frac{n-1}{3}$ ✓</td>
<td>$\frac{3n-n_2-2\ell_2}{6}$</td>
</tr>
<tr>
<td>3-regular</td>
<td>$\frac{4n-1}{9}$</td>
<td>$\frac{3n-2\ell_2}{6}$</td>
</tr>
</tbody>
</table>
## Overview

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected + planar</td>
<td>$\frac{n+4}{3}$ ✓</td>
<td>$\frac{2n+4-\ell_4}{4}$ ✓</td>
</tr>
<tr>
<td>max-deg 3</td>
<td>$\frac{n-1}{3}$ ✓</td>
<td>$\frac{3n-n_2-2\ell_2}{6}$ ✓</td>
</tr>
<tr>
<td>3-regular</td>
<td>$\frac{4n-1}{9}$ ✓</td>
<td>$\frac{3n-2\ell_2}{6}$ ✓</td>
</tr>
</tbody>
</table>
### Overview

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected + planar</td>
<td>( \frac{n+4}{3} ) ✓</td>
<td>( \frac{2n+4-\ell_4}{4} )</td>
</tr>
<tr>
<td>max-deg 3</td>
<td>( \frac{n-1}{3} ) ✓</td>
<td>( \frac{3n-n_2-2\ell_2}{6} ) ✓</td>
</tr>
<tr>
<td>3-regular</td>
<td>( \frac{4n-1}{9} ) ✓</td>
<td>( \frac{3n-2\ell_2}{6} ) ✓</td>
</tr>
</tbody>
</table>
Overview

1 Introduction
   - Definitions and known results
   - Warm-up: simple algorithms for maxdeg-\(k\) graphs

2 Graphs with maxdeg 3
   - 3-regular graphs
   - Graphs with maxdeg 3

3 The missing algorithm and maximum matchings
   - 3-connected planar graphs
   - Graphs with bounded-degree block trees
Separating triplets and the 4-block tree
Separating triplets and the 4-block tree
Separating triplets and the 4-block tree
Separating triplets and the 4-block tree

4-block tree:

$$\ell_4 = 2$$

Ignaz Rutter and Alexander Wolff
Algorithm: same story as before?

Cut off leaves and compute perfect matchings in 3-connected planar graphs whose 4-block tree is a path.
Algorithm: same story as before?

Cut off leaves and compute perfect matchings in 3-connected planar graphs whose 4-block tree is a path.
Algorithm: same story as before?

Cut off leaves and compute perfect matchings in 3-connected planar graphs whose 4-block tree is a path.

- Hamiltonian cycles take $O(n)$ time in 4-connected planar graphs. [Chiba, Nishizeki '89]
- Compute matchings in 4-blocks and combine by DP.
From 4-block paths to 4-block trees

Lemma

Let $G$ be a 3-connected planar graph whose 4-block tree is a path. A (nearly) perfect matching in $G$ can be computed in $O(n)$ time.
From 4-block paths to 4-block trees

Lemma

Let $G$ be a 3-connected planar graph whose 4-block tree is a path. A (nearly) perfect matching in $G$ can be computed in $O(n)$ time.

Sizes of matchings
in 3-connected planar graph whose 4-block tree has $\ell_4$ leaves:

Biedl et al.: $\frac{2n + 4 - \ell_4}{4}$ existence

Our algorithm: $\frac{2n + 4 - 6\ell_4}{4}$ in $O(n\alpha(n))$ time.

Triangulation: $\frac{2n + 4 - 2\ell_4}{4}$ in $O(n)$ time.
## Overview

<table>
<thead>
<tr>
<th>Graph</th>
<th>Bound 1</th>
<th>Bound 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>3-connected planar</td>
<td>$\frac{n+4}{3}$ ✓</td>
<td>$\frac{2n+4-6\ell_4}{4}$ ✓</td>
</tr>
<tr>
<td>maxdeg 3</td>
<td>$\frac{n-1}{3}$ ✓</td>
<td>$\frac{3n-n_2-2\ell_2}{6}$ ✓</td>
</tr>
<tr>
<td>3-regular</td>
<td>$\frac{4n-1}{9}$ ✓</td>
<td>$\frac{3n-2\ell_2}{6}$ ✓</td>
</tr>
</tbody>
</table>
Overview

1. Introduction
   - Definitions and known results
   - Warm-up: simple algorithms for maxdeg-$k$ graphs

2. Graphs with maxdeg 3
   - 3-regular graphs
   - Graphs with maxdeg 3

3. The missing algorithm and maximum matchings
   - 3-connected planar graphs
   - Graphs with bounded-degree block trees
Bounded-degree block trees

Theorem

Let $G$ be a 3-connected planar graph with bounded-deg. 4-block tree. Maximum matching takes $O(n^{\alpha(n)})$ time.

Proof:

- Compute local matchings in 4-blocks.
- Count number of free vertices for every configuration.
- Use DP to find a maximum matching.
Bounded-degree block trees

Theorem

Let $G$ be a 3-connected planar graph with bounded-deg. 4-block tree. Maximum matching takes $O(n^{\alpha(n)})$ time.

Theorem

Let $G$ be a 3-regular graph with bounded-deg. 2-block tree. Maximum matching takes $O(n \log^4 n)$ time; planar case: $O(n)$ time.
Bounded-degree block trees

Theorem

Let $G$ be a 3-connected planar graph with bounded-deg. 4-block tree. Maximum matching takes $O(n^{\alpha(n)})$ time.

Theorem

Let $G$ be a 3-regular graph with bounded-deg. 2-block tree. Maximum matching takes $O(n \log^4 n)$ time; planar case: $O(n)$ time.

Can we do better??
Bounded-degree block trees

**Theorem**

Let $G$ be a 3-connected planar graph with bounded-deg. 4-block tree. Maximum matching takes $O(n \alpha(n))$ time.

**Theorem**

Let $G$ be a 3-regular graph with bounded-deg. 2-block tree. Maximum matching takes $O(n \log^4 n)$ time; planar case: $O(n)$ time.

Can we do better??

There are linear-time reductions:

- max. matchings in planar graphs $\rightarrow$ in triangulated planar graphs
- max. matchings in general graphs $\rightarrow$ in 3-regular graphs
## Conclusion and open questions

<table>
<thead>
<tr>
<th>graph class</th>
<th>bound on matching size</th>
<th>runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>type-1</td>
<td>type-2</td>
</tr>
<tr>
<td>3-regular maxdeg-3</td>
<td>$(4n - 1)/9$</td>
<td>$n \log^4 n$</td>
</tr>
<tr>
<td></td>
<td>$(n - 1)/3$</td>
<td>$n</td>
</tr>
<tr>
<td>3-connected, planar, $n \geq 10$</td>
<td>$(n + 4)/3$</td>
<td>$n</td>
</tr>
<tr>
<td>3-regular planar triangulated,</td>
<td>$(3n - 2\ell_2)/6$</td>
<td>$n$</td>
</tr>
<tr>
<td>planar</td>
<td>$(2n + 4 - 6\ell_4)/4$</td>
<td>$n$</td>
</tr>
<tr>
<td>maxdeg-3</td>
<td></td>
<td>$n$</td>
</tr>
<tr>
<td>maxdeg-k</td>
<td></td>
<td>$n \log^4 n$</td>
</tr>
<tr>
<td>3-reg., bnd.-deg 2-bt maxdeg-k</td>
<td>maximum</td>
<td>$n$</td>
</tr>
<tr>
<td>3-reg., planar, bnd.-deg 2-bt</td>
<td>maximum</td>
<td>$n \alpha(n)$</td>
</tr>
<tr>
<td>3-conn., planar, bnd.-deg 4-bt</td>
<td>maximum</td>
<td>$n \alpha(n)$</td>
</tr>
</tbody>
</table>

- Improve **running time** in the planar case!
- Remove 6!