Computing Large Matchings Fast

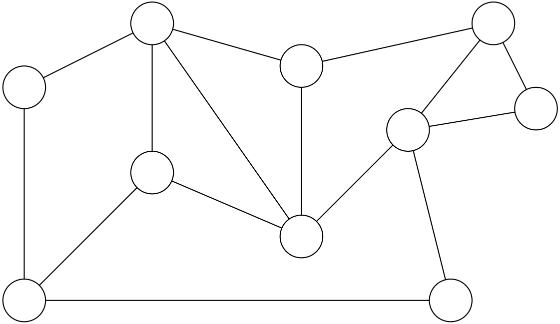
Ignaz Rutter Alexander Wolff

Karlsruhe University TU Eindhoven

Overview

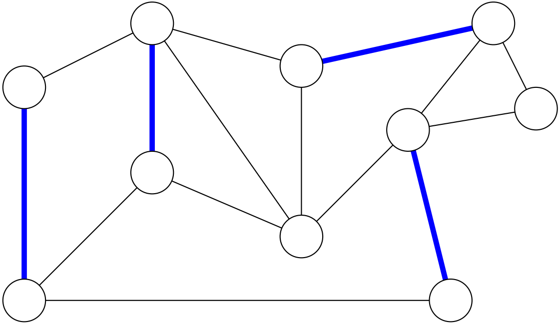
Overview

Matching



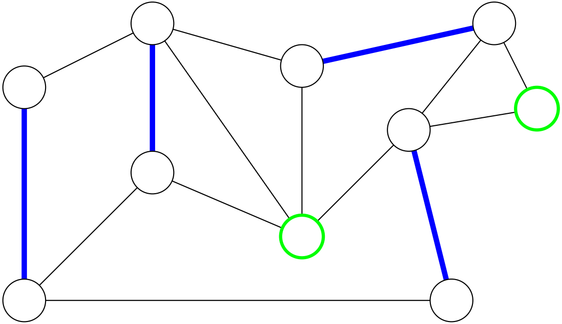
Given an undirected graph $G = (V, E)$...

Matching



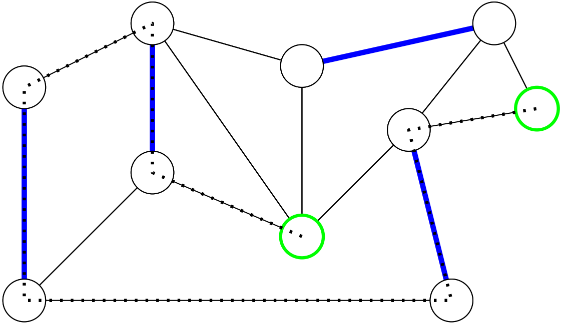
Given an undirected graph $G = (V, E)$...
...a *matching* is a set M of independent edges.

Matching



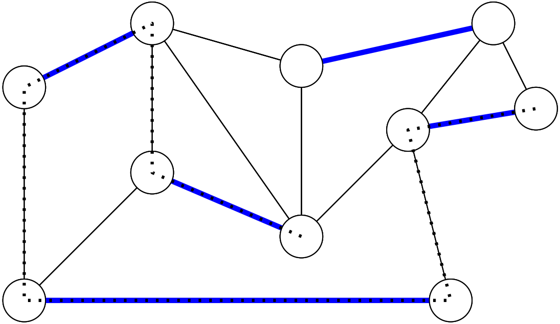
A *free vertex* is a vertex that is not incident to an edge of M.

Matching



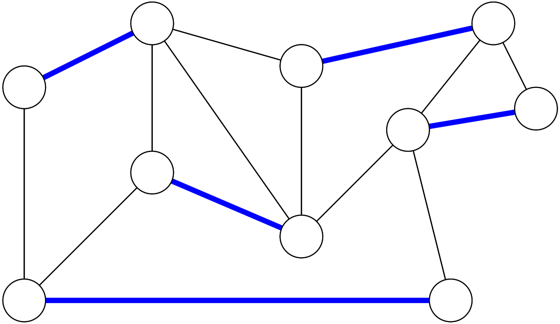
An *augmenting path* is a path that alternates between matching and non-matching edges, and starts and ends at different free vertices.

Matching



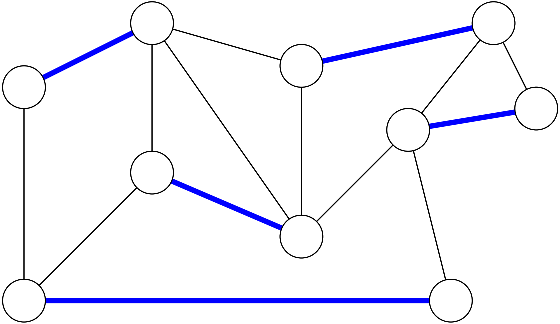
An *augmenting path* is a path that alternates between matching and non-matching edges, and starts and ends at different free vertices.

Matching



A *maximum matching* is a matching of maximum cardinality.

Matching



Theorem (Berge)

A matching is maximum ⇔ 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. [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	– " –

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: $O(nm\alpha(n, m))$ 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]

Graph	Bound 1	Bound 2
3-connected, planar	$\frac{n+4}{3}$	$\frac{2n+4-\ell_4}{4}$
maxdeg 3	$\frac{n-1}{3}$	$\frac{3n-n_2-2\ell_2}{6}$
3-regular	$\frac{4n-1}{9}$	$\frac{3n-2\ell_2}{6}$

Known results

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

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

Graph	Bound 1	Bound 2
3-connected, planar	$\frac{n+4}{3}$	$\frac{2n+4-\ell_4}{4}$
maxdeg 3	$\frac{n-1}{3}$	$\frac{3n-n_2-2\ell_2}{6}$
3-regular	$\frac{4n-1}{9}$	$\frac{3n-2\ell_2}{6}$

There are linear-time reductions: [Biedl SODA'01]

- 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

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 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.

This computes a maximum matching in a tree.

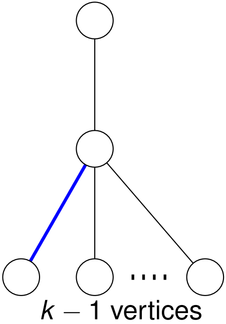
Maximum matchings in trees

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: $|matching| \geq (n-1)/3$ 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: $|matching| \geq (n-1)/3$ *in* $O(n)$ *time.*

Corollary

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

Corollary

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

Proof: A spanning tree with maxdeg 3 exists [Barnette '66]
and can be computed in $O(n)$ time. [Czumaj, Strothmann '97]

Overview

Graph	Bound 1	Bound 2
3-connected + planar	$\frac{n+4}{3}$	$\frac{2n+4-\ell_4}{4}$
max-deg 3	$\frac{n-1}{3}$	$\frac{3n-n_2-2\ell_2}{6}$
3-regular	$\frac{4n-1}{9}$	$\frac{3n-2\ell_2}{6}$

Overview

Graph	Bound 1	Bound 2
3-connected + planar	$\frac{n+4}{3}$	$\frac{2n+4-\ell_4}{4}$
max-deg 3	$\frac{n-1}{3}$ $\surd$	$\frac{3n-n_2-2\ell_2}{6}$
3-regular	$\frac{4n-1}{9}$	$\frac{3n-2\ell_2}{6}$

Overview

Graph	Bound 1	Bound 2
3-connected + planar	$\frac{n+4}{3}$	$\frac{2n+4-\ell_4}{4}$
max-deg 3	$\frac{n-1}{3}$ $\surd$	$\frac{3n-n_2-2\ell_2}{6}$
3-regular	$\frac{4n-1}{9}$	$\frac{3n-2\ell_2}{6}$

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

Overview

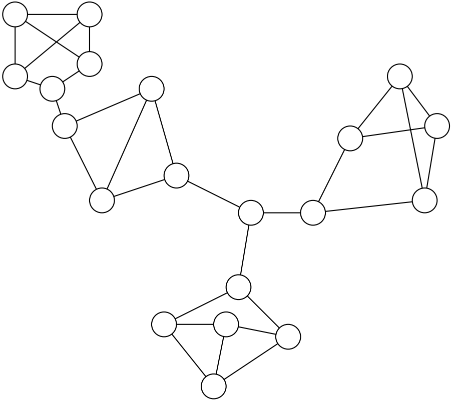
Graph	Bound 1	Bound 2
3-connected + planar	$\frac{n+4}{3}$ √	$\frac{2n+4-\ell_4}{4}$
max-deg 3	$\frac{n-1}{3}$ √	$\frac{3n-n_2-2\ell_2}{6}$
3-regular	$\frac{4n-1}{9}$	$\frac{3n-2\ell_2}{6}$

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

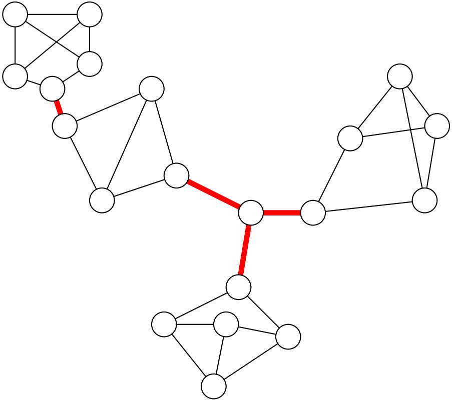
Overview

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

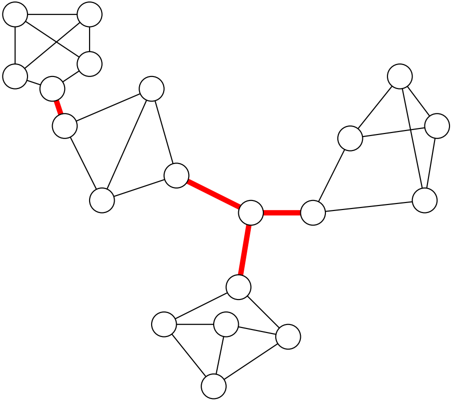
Bridge



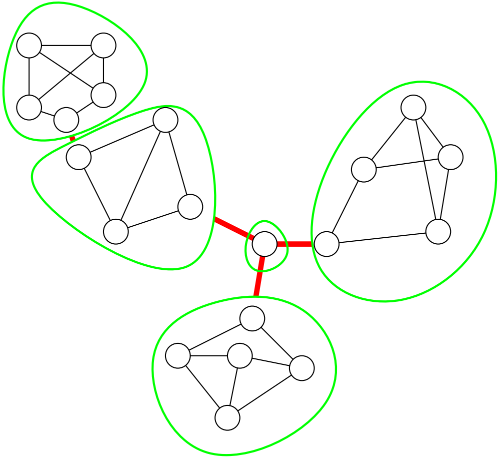
Bridge



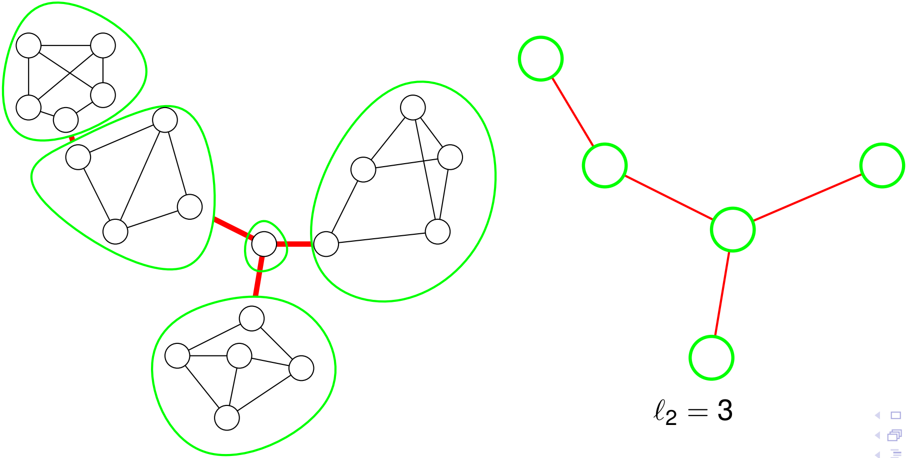
2-block tree



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)

Such 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 ℓ_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 ℓ_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 ℓ_2
- known: theorem holds for $\ell_2 = 1, 2$
- treat cases $\ell_2 = 3, 4$ separately

Arbitrary 3-regular graphs

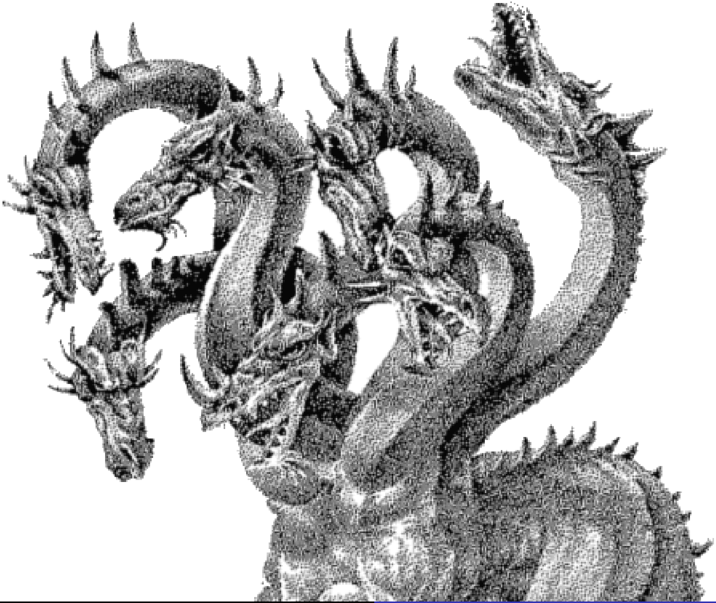
Biedl et al.: Every 3-regular graph whose 2-block tree has ℓ_2 leaves has a matching of size at least $(3n - 2\ell_2)/6$ ...

Theorem

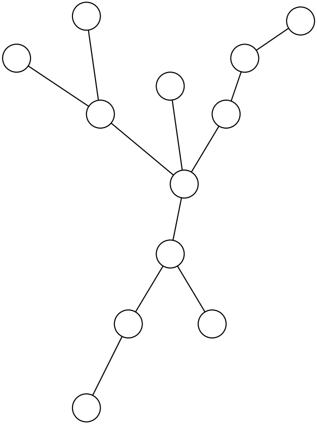
... such that every free vertex is incident to a bridge.

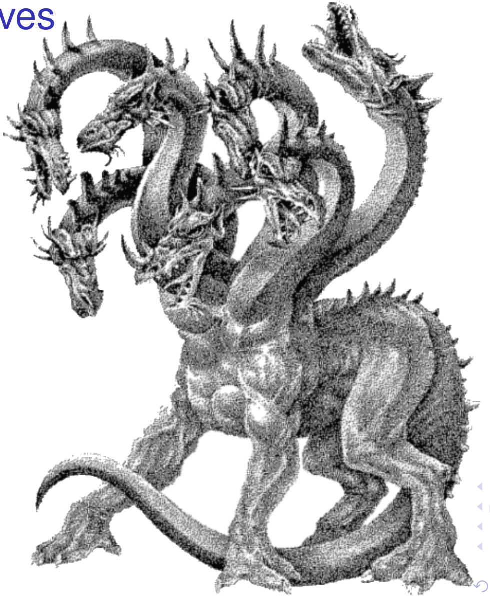
- constructive proof: induction on ℓ_2
- known: theorem holds for $\ell_2 = 1, 2$
- treat cases $\ell_2 = 3, 4$ separately

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

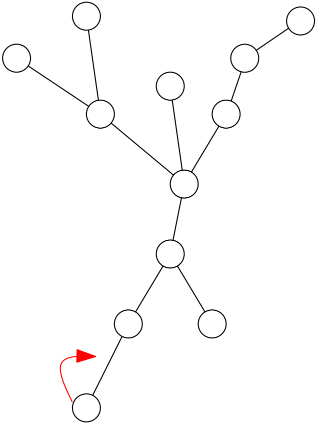


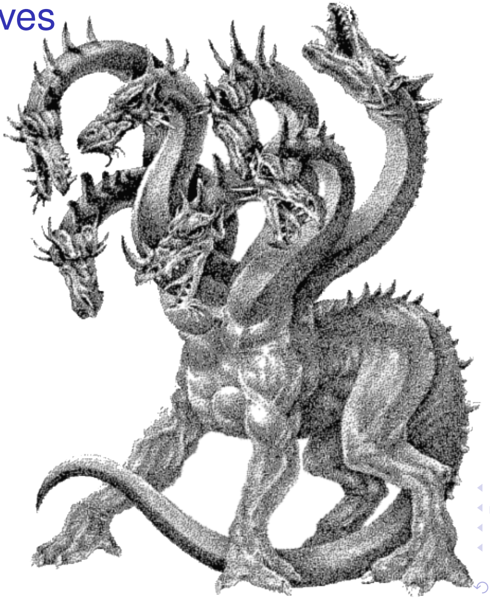
Case $\ell_2 \geq 5$: Cutting leaves



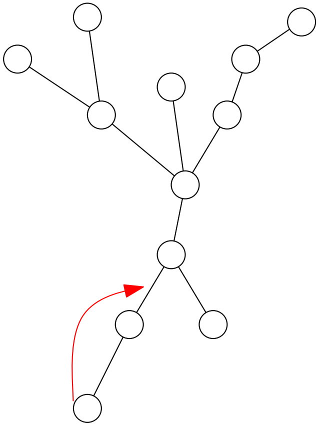


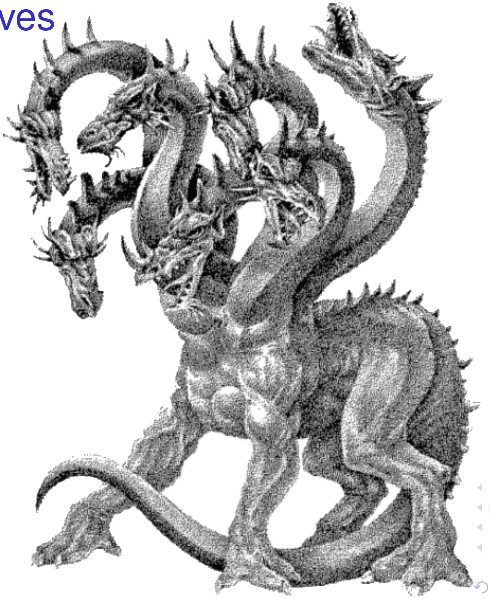
Case $\ell_2 \geq 5$: Cutting leaves



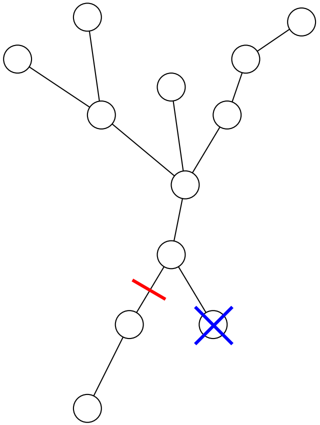


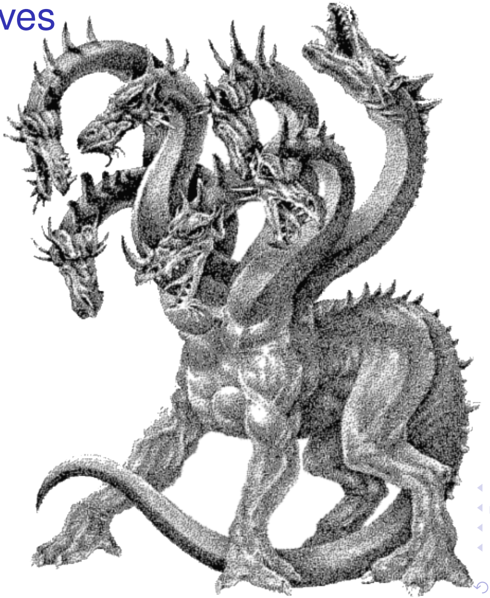
Case $\ell_2 \geq 5$: Cutting leaves



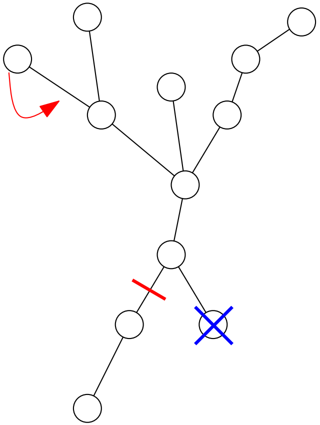


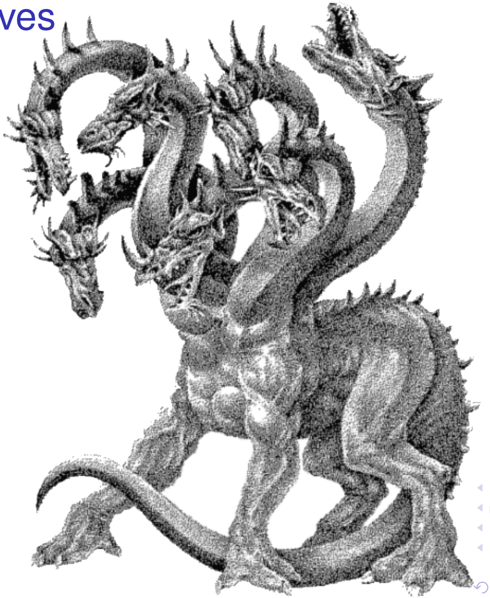
Case $\ell_2 \geq 5$: Cutting leaves



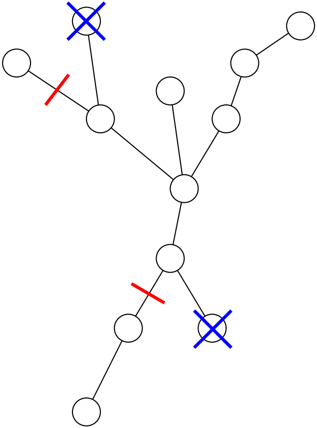


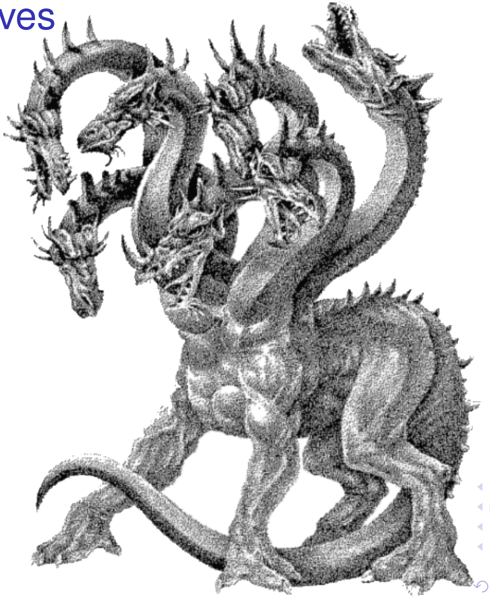
Case $\ell_2 \geq 5$: Cutting leaves



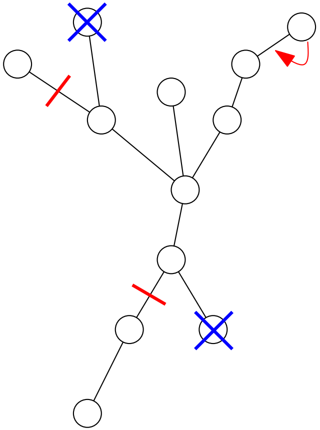


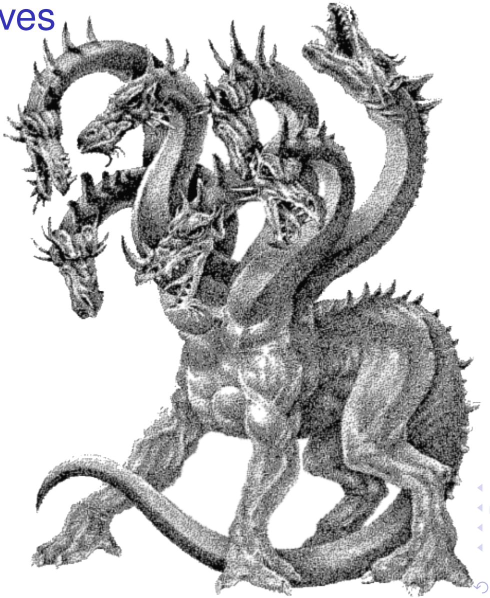
Case $\ell_2 \geq 5$: Cutting leaves



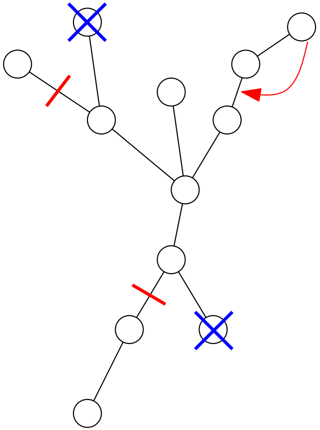


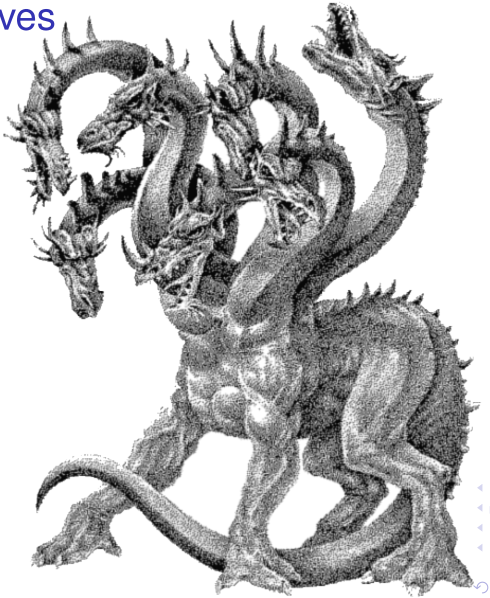
Case $\ell_2 \geq 5$: Cutting leaves



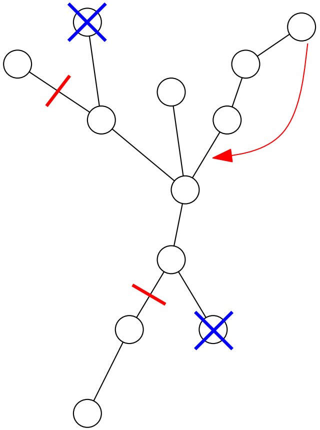


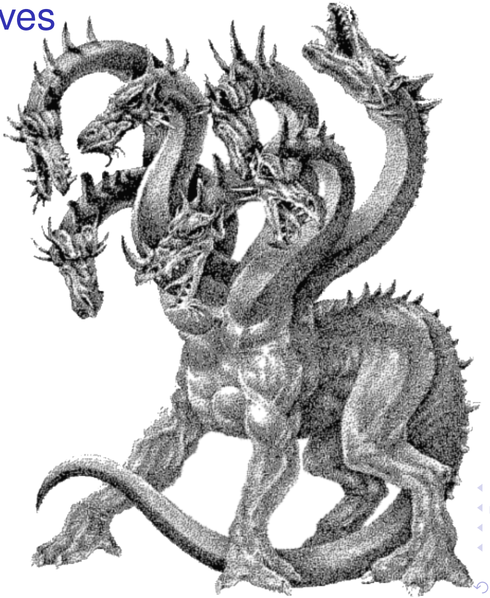
Case $\ell_2 \geq 5$: Cutting leaves



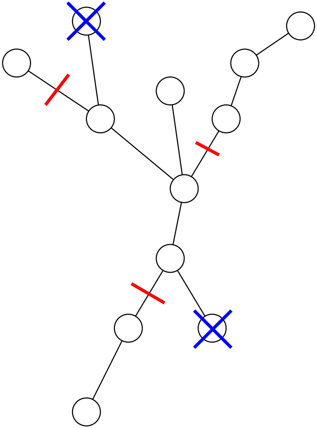


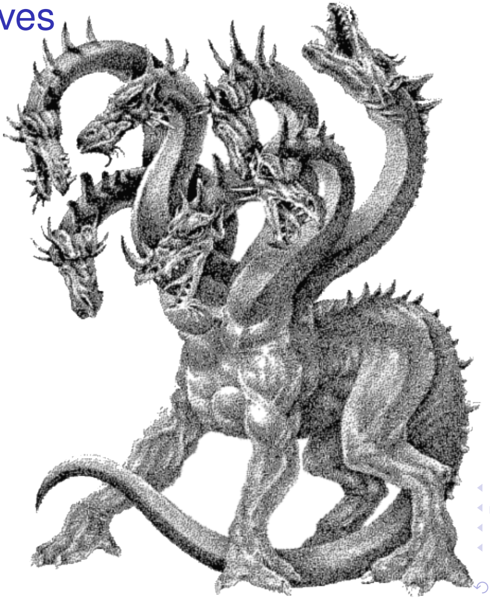
Case $\ell_2 \geq 5$: Cutting leaves



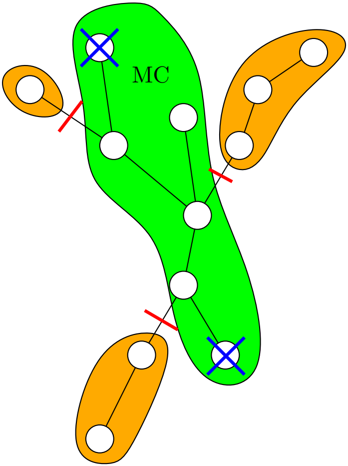


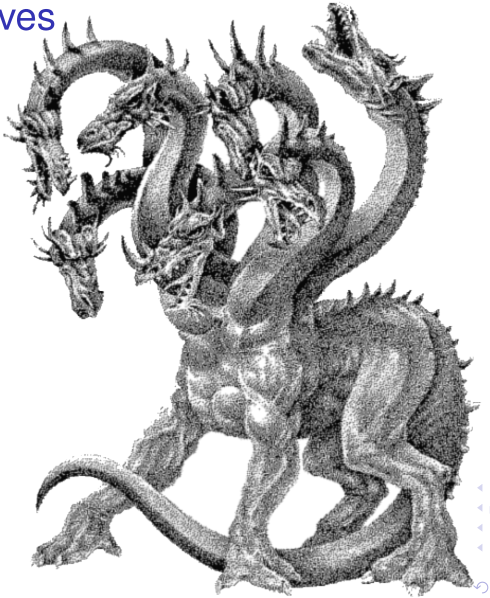
Case $\ell_2 \geq 5$: Cutting leaves



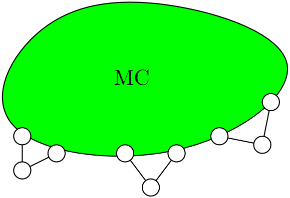


Case $\ell_2 \geq 5$: Cutting leaves



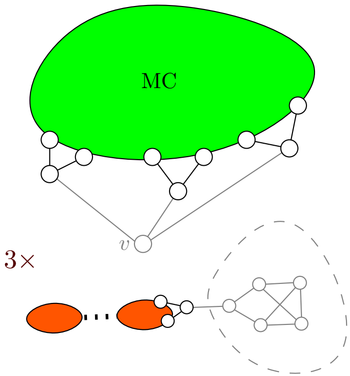


Repairing the cuts

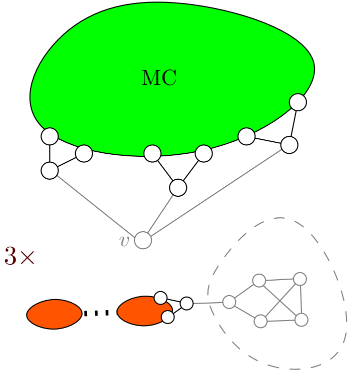


$3\times$

Repairing the cuts

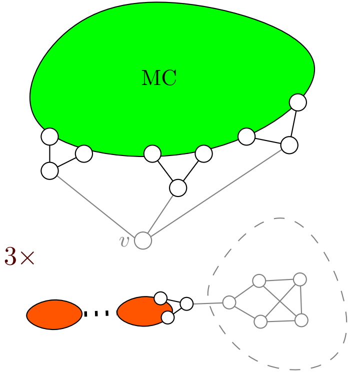


Repairing the cuts



- Compute matchings in all four components.

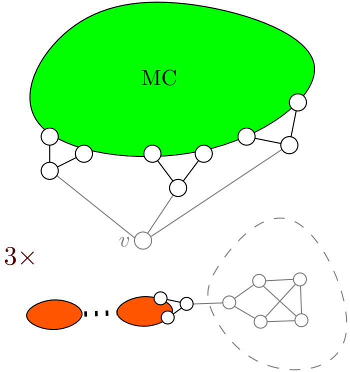
Repairing the cuts



- Compute matchings in all four components.

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

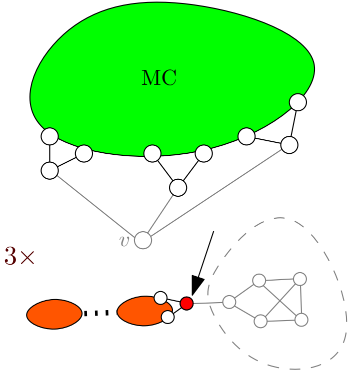
Repairing the cuts



$\ell_2(MC) = \ell_2(G) - 3.$
$\#freevertices_G = \#freevertices_{MC}$

- Compute matchings in all four components.

Repairing the cuts

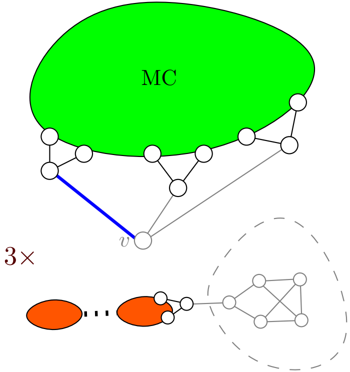


- Compute matchings in all four components.

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

$\#freevertices_G = \#freevertices_{MC} + 3$

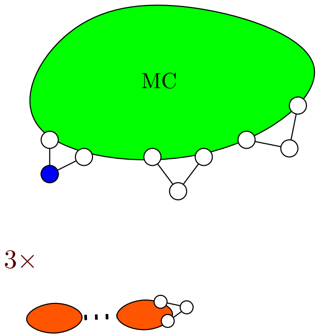
Repairing the cuts



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

$\ell_2(MC) = \ell_2(G) - 3.$
$\#freevertices_G = \#freevertices_{MC} + 3$

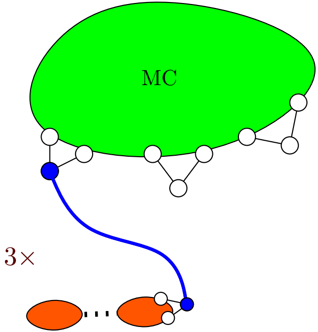
Repairing the cuts



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

$\ell_2(MC) = \ell_2(G) - 3.$
$\#freevertices_G = \#freevertices_{MC} + 3$

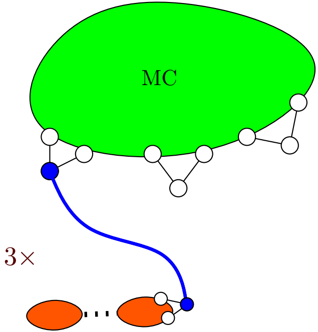
Repairing the cuts



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

$\ell_2(MC) = \ell_2(G) - 3.$
$\#freevertices_G = \#freevertices_{MC} + 3 - 1$

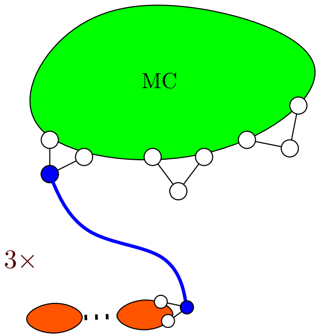
Repairing the cuts



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

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

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.

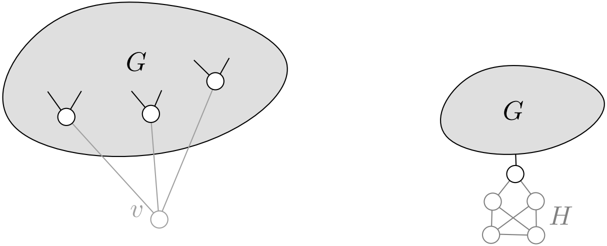
$\ell_2(MC) = \ell_2(G) - 3.$
$\#freevertices_G = \#freevertices_{MC} + 3 - 1 = \#freevertices_{MC} + 2.$

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

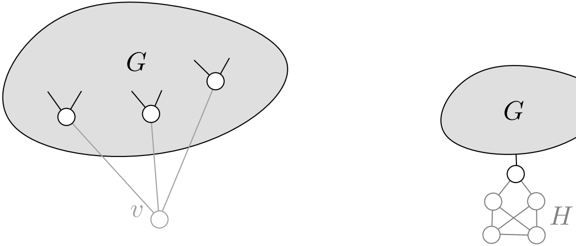
Graphs with maxdeg 3

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

Graph	Bound 1	Bound 2
3-connected + planar	$\frac{n+4}{3}$ √	$\frac{2n+4-\ell_4}{4}$
max-deg 3	$\frac{n-1}{3}$ √	$\frac{3n-n_2-2\ell_2}{6}$
3-regular	$\frac{4n-1}{9}$	$\frac{3n-2\ell_2}{6}$

Overview

Graph	Bound 1	Bound 2
3-connected + planar	$\frac{n+4}{3}$ √	$\frac{2n+4-\ell_4}{4}$
max-deg 3	$\frac{n-1}{3}$ √	$\frac{3n-n_2-2\ell_2}{6}$ √
3-regular	$\frac{4n-1}{9}$ √	$\frac{3n-2\ell_2}{6}$ √

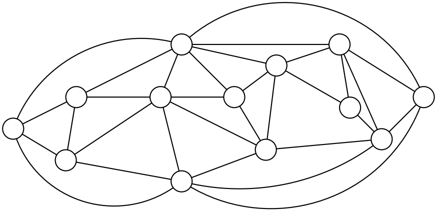
Overview

Graph	Bound 1	Bound 2
3-connected + planar	$\frac{n+4}{3}$ √	$\frac{2n+4-\ell_4}{4}$
max-deg 3	$\frac{n-1}{3}$ √	$\frac{3n-n_2-2\ell_2}{6}$ √
3-regular	$\frac{4n-1}{9}$ √	$\frac{3n-2\ell_2}{6}$ √

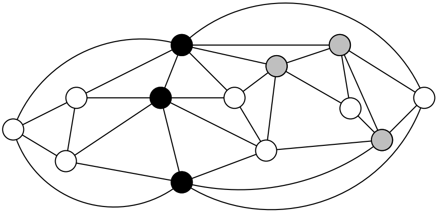
Overview

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

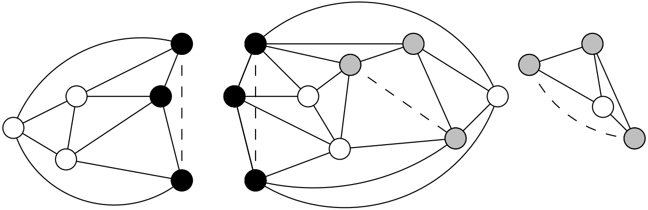
Separating triplets and the 4-block tree



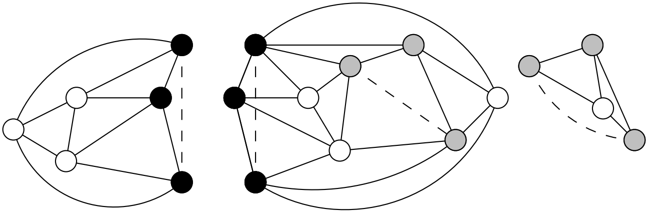
Separating triplets and the 4-block tree



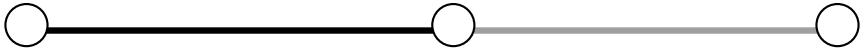
Separating triplets and the 4-block tree



Separating triplets and the 4-block tree



4-block tree:



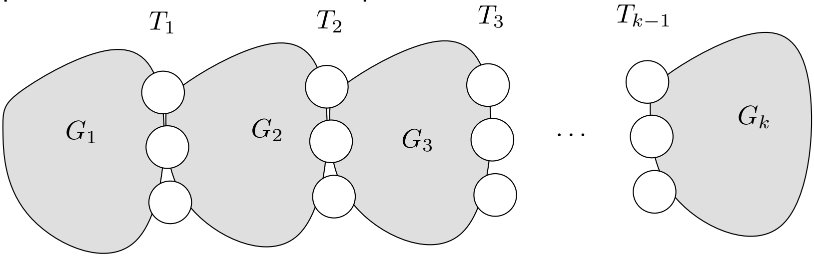
$\ell_4 = 2$

Algorithm: same story as before?

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

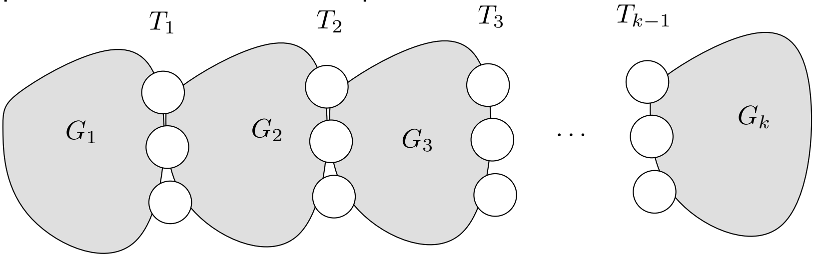
Algorithm: same story as before?

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 ℓ_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

Graph	Bound 1	Bound 2
3-connected planar	$\frac{n+4}{3}$ √	$\frac{2n+4-6\ell_4}{4}$ √
maxdeg 3	$\frac{n-1}{3}$ √	$\frac{3n-n_2-2\ell_2}{6}$ √
3-regular	$\frac{4n-1}{9}$ √	$\frac{3n-2\ell_2}{6}$ √

Overview

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: [Biedl SODA'01]

- max. matchings in planar graphs → in triangulated planar graphs
- max. matchings in general graphs → in 3-regular graphs

Conclusion and open questions

graph class	bound on matching size: type-1	bound on matching size: type-2	runtime $O(\cdot)$
3-regular	$(4n-1)/9$	$(3n-2\ell_2)/6$	$n \log^4 n$
maxdeg-3	$(n-1)/3$	$(3n-n_2-2\ell_2)/6$	$n \mid n \log^4 n$
3-connected, planar, $n \geq 10$	$(n+4)/3$	$(2n+4-6\ell_4)/4$	$n \mid n\,\alpha(n)$
3-regular planar	$(3n-6\ell_2)/6$		n
triangulated, planar	$(2n+4-2\ell_4)/4$		n
maxdeg-k	$(n-1)/k$		n
3-reg., bnd.-deg 2-bt	maximum		$n \log^4 n$
3-reg., planar, bnd.-deg 2-bt	maximum		n
3-conn., planar, bnd.-deg 4-bt	maximum		$n\,\alpha(n)$

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