

# A Fast Matching Algorithm

## Algorithm 52 Bimatch-Hopcroft-Karp( $G$ )

```
1:  $M \leftarrow \emptyset$ 
2: repeat
3:   let  $\mathcal{P} = \{P_1, \dots, P_k\}$  be maximal set of
4:   vertex-disjoint, shortest augmenting path w.r.t.  $M$ .
5:    $M \leftarrow M \oplus (P_1 \cup \dots \cup P_k)$ 
6: until  $\mathcal{P} = \emptyset$ 
7: return  $M$ 
```

We call one iteration of the repeat-loop a **phase** of the algorithm.

# Analysis Hopcroft-Karp

## Lemma 90

Given a matching  $M$  and a matching  $M^*$  with  $|M^*| - |M| \geq 0$ .  
There exist  $|M^*| - |M|$  *vertex-disjoint augmenting paths* w.r.t.  $M$ .

### Proof:

- ▶ Similar to the proof that a matching is optimal iff it does not contain an augmenting path.
- ▶ Consider the graph  $G = (V, M \oplus M^*)$ , and mark edges in this graph blue if they are in  $M$  and red if they are in  $M^*$ .
- ▶ The connected components of  $G$  are cycles and paths.
- ▶ The graph contains  $k \stackrel{\text{def}}{=} |M^*| - |M|$  more red edges than blue edges.
- ▶ Hence, there are at least  $k$  components that form a path starting and ending with a red edge. These are augmenting paths w.r.t.  $M$ .

# Analysis Hopcroft-Karp

- ▶ Let  $P_1, \dots, P_k$  be a maximal collection of vertex-disjoint, shortest augmenting paths w.r.t.  $M$  (let  $\ell = |P_i|$ ).
- ▶  $M' \stackrel{\text{def}}{=} M \oplus (P_1 \cup \dots \cup P_k) = M \oplus P_1 \oplus \dots \oplus P_k$ .
- ▶ Let  $P$  be an augmenting path in  $M'$ .

## Lemma 91

The set  $A \stackrel{\text{def}}{=} M \oplus (M' \oplus P) = (P_1 \cup \dots \cup P_k) \oplus P$  contains at least  $(k + 1)\ell$  edges.

# Analysis Hopcroft-Karp

## Proof.

- ▶ The set describes exactly the symmetric difference between matchings  $M$  and  $M' \oplus P$ .
- ▶ Hence, the set contains at least  $k + 1$  vertex-disjoint augmenting paths w.r.t.  $M$  as  $|M'| = |M| + k + 1$ .
- ▶ Each of these paths is of length at least  $\ell$ .

# Analysis Hopcroft-Karp

## Lemma 92

$P$  is of length at least  $\ell + 1$ . This shows that the length of a shortest augmenting path increases between two phases of the Hopcroft-Karp algorithm.

### Proof.

- ▶ If  $P$  does not intersect any of the  $P_1, \dots, P_k$ , this follows from the maximality of the set  $\{P_1, \dots, P_k\}$ .
- ▶ Otherwise, at least one edge from  $P$  coincides with an edge from paths  $\{P_1, \dots, P_k\}$ .
- ▶ This edge is not contained in  $A$ .
- ▶ Hence,  $|A| \leq k\ell + |P| - 1$ .
- ▶ The lower bound on  $|A|$  gives  $(k + 1)\ell \leq |A| \leq k\ell + |P| - 1$ , and hence  $|P| \geq \ell + 1$ .

# Analysis Hopcroft-Karp

If the shortest augmenting path w.r.t. a matching  $M$  has  $\ell$  edges then the cardinality of the maximum matching is of size at most  $|M| + \frac{|V|}{\ell+1}$ .

## Proof.

The symmetric difference between  $M$  and  $M^*$  contains  $|M^*| - |M|$  vertex-disjoint augmenting paths. Each of these paths contains at least  $\ell + 1$  vertices. Hence, there can be at most  $\frac{|V|}{\ell+1}$  of them.

# Analysis Hopcroft-Karp

## Lemma 93

*The Hopcroft-Karp algorithm requires at most  $2\sqrt{|V|}$  phases.*

### Proof.

- ▶ After iteration  $\lfloor \sqrt{|V|} \rfloor$  the length of a shortest augmenting path must be at least  $\lfloor \sqrt{|V|} \rfloor + 1 \geq \sqrt{|V|}$ .
- ▶ Hence, there can be at most  $|V| / (\sqrt{|V|} + 1) \leq \sqrt{|V|}$  additional augmentations.

# Analysis Hopcroft-Karp

## Lemma 94

*One phase of the Hopcroft-Karp algorithm can be implemented in time  $\mathcal{O}(m)$ .*

construct a “level graph”  $G'$ :

- ▶ construct Level 0 that includes all free vertices on left side  $L$
- ▶ construct Level 1 containing all neighbors of Level 0
- ▶ construct Level 2 containing **matching** neighbors of Level 1
- ▶ construct Level 3 containing all neighbors of Level 2
- ▶ ...
- ▶ stop when a level (apart from Level 0) contains a free vertex

can be done in time  $\mathcal{O}(m)$  by a modified BFS



# Analysis Hopcroft-Karp

- ▶ a shortest augmenting path **must** go from Level 0 to the last layer constructed
- ▶ it can only use edges between layers
- ▶ construct a maximal set of vertex disjoint augmenting path connecting the layers
- ▶ for this, go forward until you either reach a free vertex or you reach a “dead end”  $v$
- ▶ if you reach a free vertex delete the augmenting path and all incident edges from the graph
- ▶ if you reach a dead end backtrack and delete  $v$  together with its incident edges

# Analysis Hopcroft-Karp

See lecture versions of the slides.

# Analysis: Shortest Augmenting Path for Flows

**cost for searches during a phase is  $\mathcal{O}(mn)$**

- ▶ a search (successful or unsuccessful) takes time  $\mathcal{O}(n)$
- ▶ a search deletes at least one edge from the level graph

**there are at most  $n$  phases**

Time:  $\mathcal{O}(mn^2)$ .

# Analysis for Unit-capacity Simple Networks

**cost for searches during a phase is  $\mathcal{O}(m)$**

- ▶ an edge/vertex is traversed at most twice

**need at most  $\mathcal{O}(\sqrt{n})$  phases**

- ▶ after  $\sqrt{n}$  phases there is a cut of size at most  $\sqrt{n}$  in the residual graph
- ▶ hence at most  $\sqrt{n}$  additional augmentations required

Time:  $\mathcal{O}(m\sqrt{n})$ .