Matching algorithms (Cont…)

RAMESH K. JALLU

LEC-04

DT. 19/01//22

Recap

John Edward Hopcroft

Richard Manning Karp

- We have seen an algorithm (aka Kuhn's algorithm, 1965) based on augmenting trees which runs in $O(|V|(|V| + |E|))$ time
- If $|V| = n$ and $|E| = m$, then the running time is $O(mn)$
- If $m = O(n^2)$, then the worst case running time of the algorithm is $O(n^3)$
- Today, we study a faster bipartite matching algorithm originally proposed by John Hopcraft and Richard Karp (and independently by Alexander Karzanov) in 1973, which runs in $O(\sqrt{|V|}|E|)$
- In the case of dense graphs, the time bound becomes $O(|V|^{2.5})$

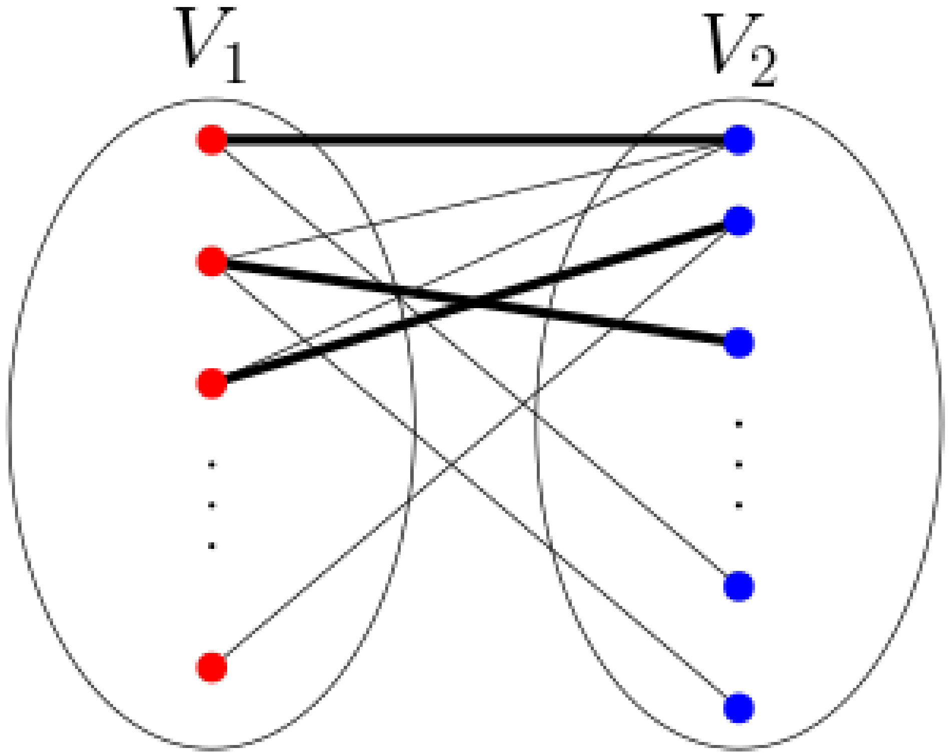
Idea of the Hopcroft-Karp Algorithm

- In Kuhn's algorithm, we choose one augmenting path in each iteration
- The number of augmentations decide the running time of the algorithm
  - In the worst, there could be $n/2$ augmentations
- In the Hopcroft-Karp algorithm, we attempt to find many disjoint augmenting paths, and use all of them to increase the size of the matching

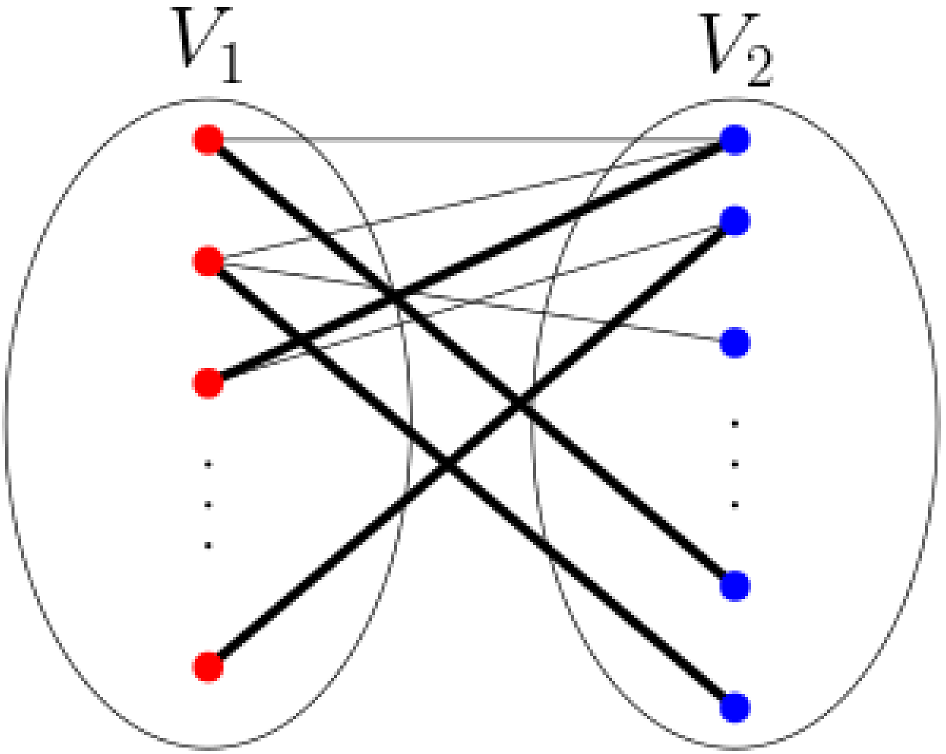
Blocking set of augmenting paths

- If G is a graph (bipartite or not) and M is a maximum matching, a ***blocking set of augmenting paths*** with respect to M is a set $\{P_1, P_2, \ldots, P_k\}$ of augmenting paths such that
  1. the paths $P_1, P_2, \ldots, P_k$ are vertex disjoint paths
  2. all the paths have the same length, say l
  3. l is the minimum length of an M-augmenting path
  4. every augmenting path of length l has at least one vertex in common with $P_1 \cup P_2 \cup \cdots \cup P_k$
- In other words, a blocking set of augmenting paths is a (set wise) maximal collection of vertex-disjoint minimum-length augmenting paths

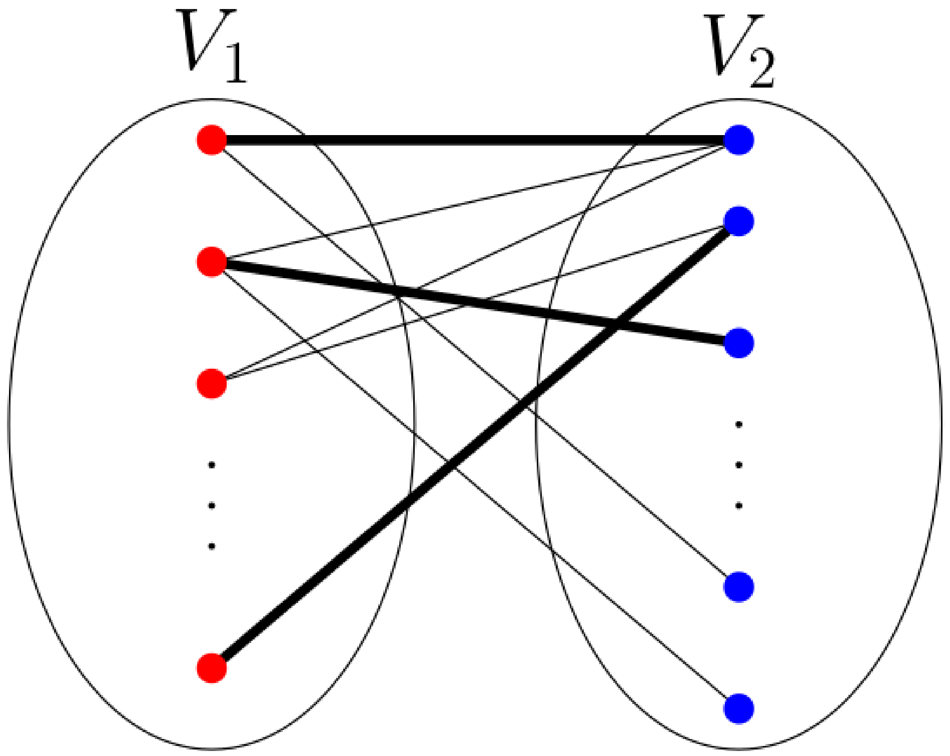
Example



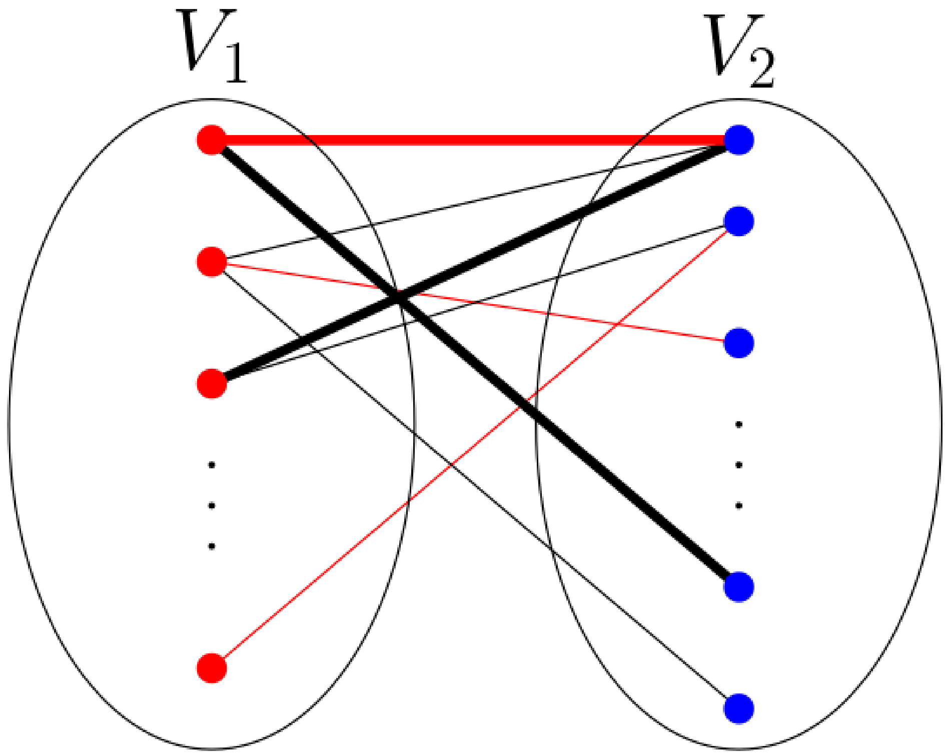
Example



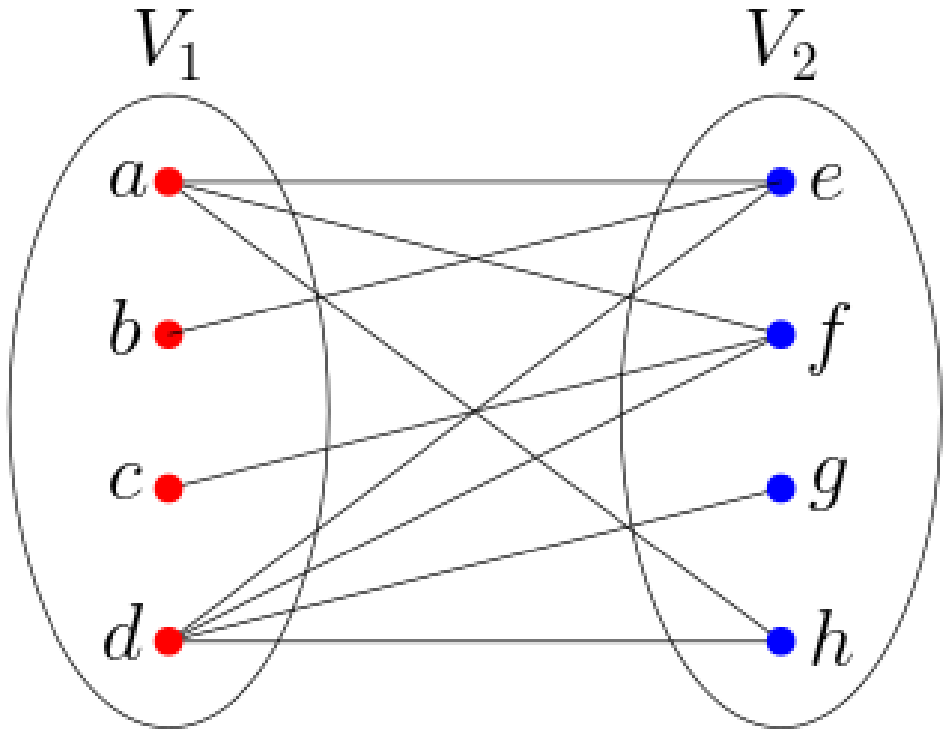
Example



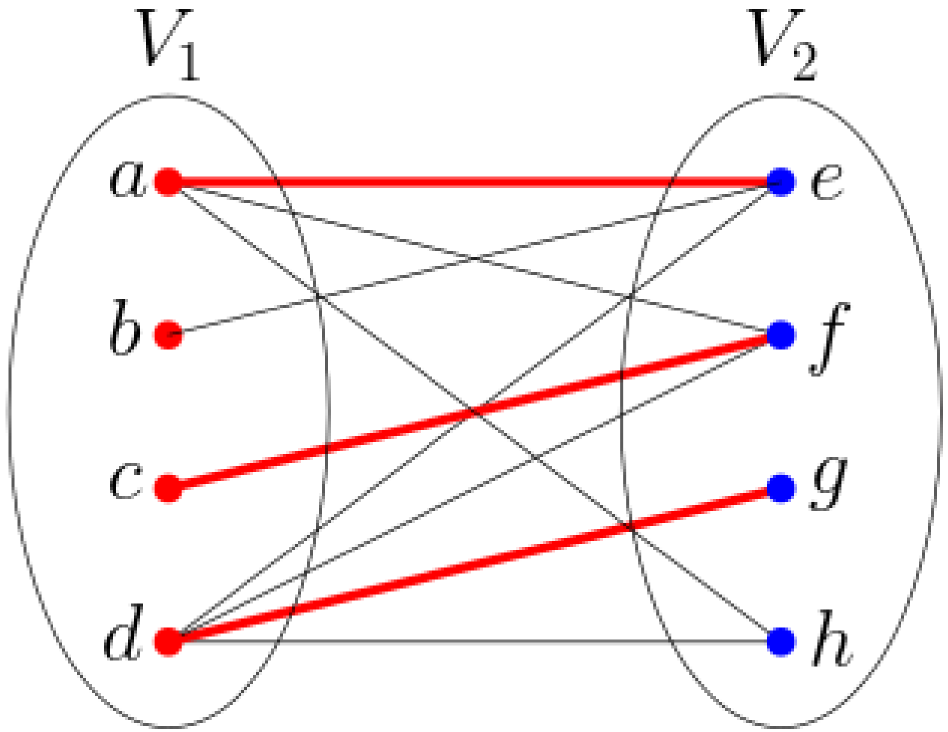
Example



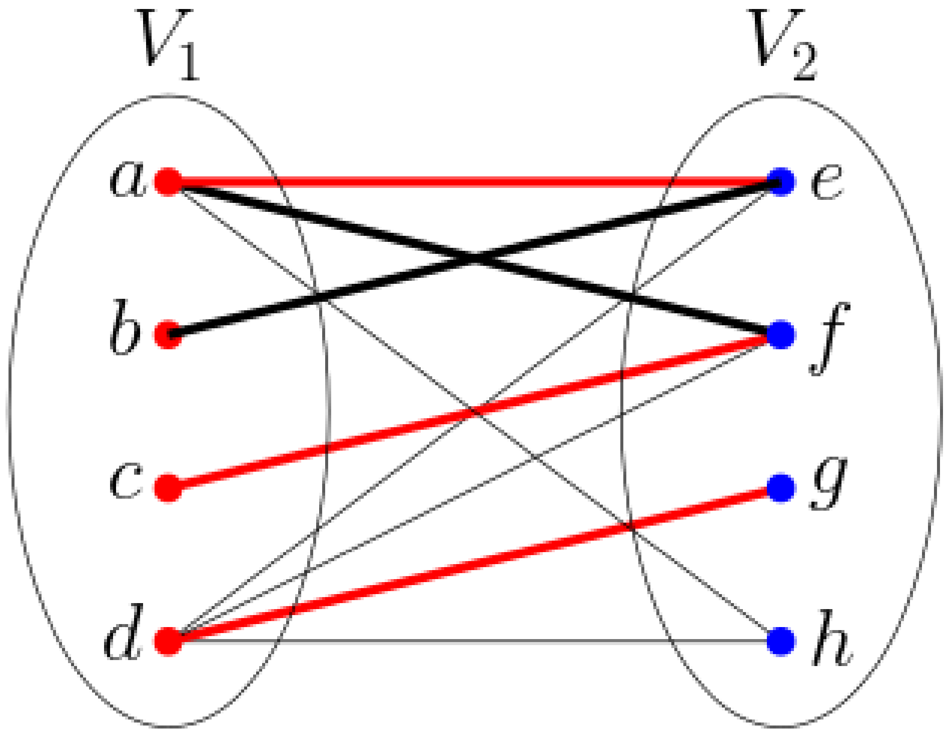
Example



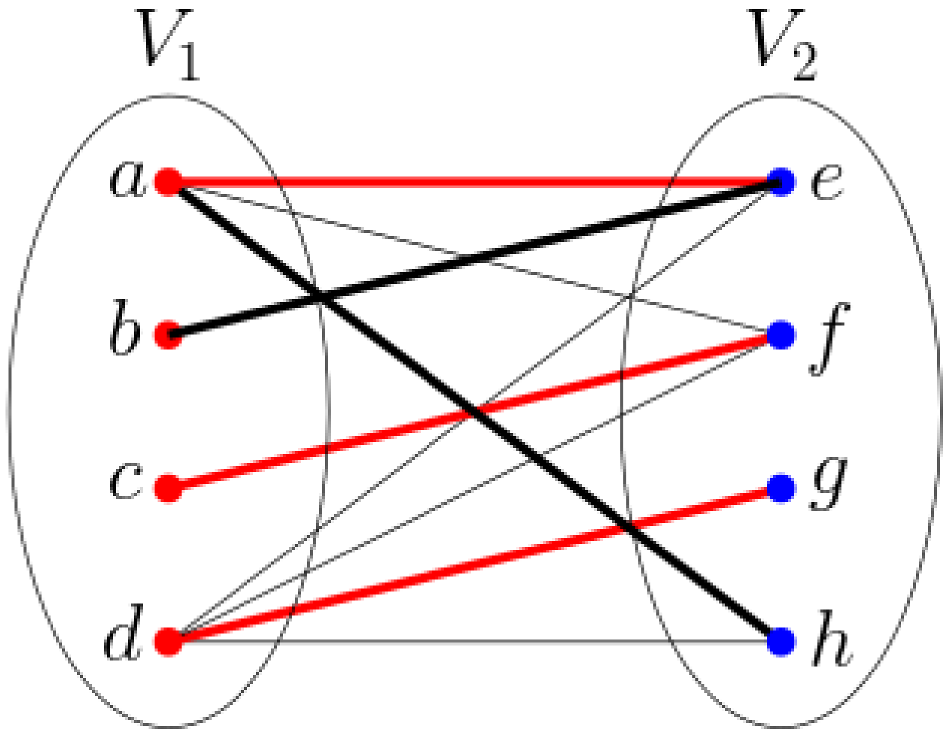
Example



Example



Example



Kuhn's algorithm (Recap)

Algorithm 1 Naïve iterative scheme for computing a maximum matching

1: Initialize $M = \emptyset$.
2: **repeat**
3:     Find an augmenting path P with respect to M.
4:     $M \leftarrow M \oplus P$
5: **until** there is no augmenting with respect to M.

Hopcroft-Karp Algorithm

Algorithm 2 Hopcroft-Karp algorithm

$M = \emptyset$
repeat
    Let $\{P_1, \ldots, P_k\}$ be a blocking set of augmenting paths with respect to M.
    $M \leftarrow M \oplus P_1 \oplus P_2 \oplus \cdots \oplus P_k$
until there is no augmenting path with respect to M

Correctness

1. In each iteration, the updated M is a matching
   - I.e., if M is a matching and $\{P_1, P_2, \ldots, P_k\}$ is any set of vertex-disjoint M-augmenting paths, then $M \oplus P_1 \oplus P_2 \oplus \ldots \oplus P_k$ is a matching of cardinality $|M| + k$
2. If M and M' are matching and maximum matchings in G; let $k = |M'| - |M|$. The edge set $M \oplus M'$ contains at least k vertex-disjoint M-augmenting paths
   - Each connected component in $M \oplus M'$ is an M-alternating component
   - Each M-alternating component which is not an M-augmenting path has at least as many edges in M as in M'
   - Each M-augmenting path has exactly one fewer edge in M as in M'
   - Therefore, at least k of the connected components of $M \oplus M'$ must be M-augmenting paths, and they are all vertex-disjoint

Correctness (Cont ⋯)

3. G has at least one M-augmenting path of length less than n/k, where n denotes the number of vertices of G

4. The minimum length of an M-augmenting path strictly increases after each iteration of the algorithm

- I.e., if $\{P_1, P_2, \ldots, P_k\}$ is any set of vertex-disjoint M-augmenting paths of length l_1, and if l_2 be the length of a shortest $M \oplus P_1 \oplus P_2 \oplus \ldots \oplus P_k$-augmenting path, then $l_2 \geq l_1 + 2$
- Let P' be an $M \oplus P_1 \oplus P_2 \oplus \ldots \oplus P_k$-augmenting path
- **Case 1**: If P' doesn't have any vertex common with $P_1, P_2, \ldots, P_k$, then P' is also an M-augmenting path. Which contradicts maximality of disjoint paths
- **Case 2**: If there is a common vertex, then $|P'| \geq |P_i| + |P' \cap P_i|$ for some i

Running time

- The Hopcroft-Karp algorithm terminates after fewer than $2\sqrt{n}$ iterations
  - After the first $\sqrt{n}$ iterations, the minimum length of an M-augmenting path is greater than $\sqrt{n}$
  - This implies, by observation 2, that $|M'| - |M| < \sqrt{n}$
  - Each remaining iteration strictly increases $|M|$, hence there are fewer than $\sqrt{n}$ iterations remaining
- By the time algorithm terminates, we have a maximum matching

Blocking set of augmenting paths

- If G is a graph (bipartite or not) and M is a maximum matching, a ***blocking set of augmenting paths*** with respect to M is a set $\{P_1, P_2, \ldots, P_k\}$ of augmenting paths such that
  1. the paths $P_1, P_2, \ldots, P_k$ are vertex disjoint paths
  2. all the paths have the same length, say l
  3. l is the minimum length of an M-augmenting path
  4. every augmenting path of length l has at least one vertex in common with $P_1 \cup P_2 \cup \cdots \cup P_k$
- In other words, a blocking set of augmenting paths is a (set wise) ***maximal*** **collection of vertex-disjoint minimum-length augmenting paths**

How to compute these paths?

- Thank you!