طراحی الگوریتم

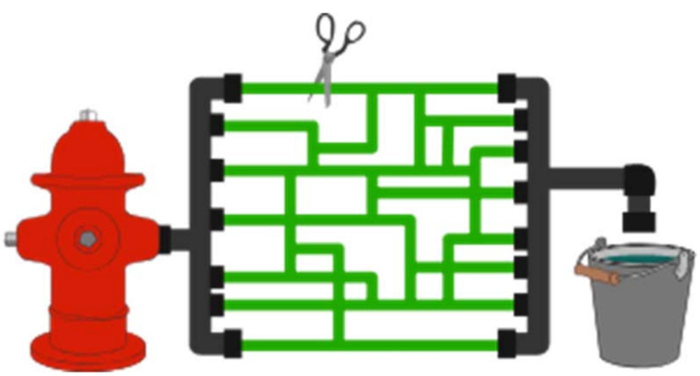
۱۷ آذر ۹۸

ملکی مجد

Topic	Reference
Recursion and Backtracking	Ch.1 and Ch.2 JeffE
Dynamic Programming	Ch.3 JeffE and Ch.15 CLRS
Greedy Algorithms	Ch.4 JeffE and Ch.16 CLRS
Amortized Analysis	Ch.17 CLRS
Elementary Graph algorithms	Ch.6 JeffE and Ch.22 CLRS
Minimum Spanning Trees	Ch.7 JeffE and Ch.23 CLRS
Single-Source Shortest Paths	Ch.8 JeffE and Ch.24 CLRS
All-Pairs Shortest Paths	Ch.9 JeffE and Ch.25 CLRS
Maximum Flow	Ch.10 JeffE and Ch.26 CLRS
String Matching	Ch.32 CLRS
NP-Completeness	Ch.12 JeffE and Ch.34 CLRS

Maximum Flow

+a graph-theoretic definition of flow networks



flow

A flow in G is a **real-valued function f** $: V \times V \rightarrow R$ that satisfies the following three properties:

- **Capacity constraint:** For all $u, v \in V$, we require $f(u, v) \leq c(u, v)$.
- **Skew symmetry:** For all $u, v \in V$, we require $f(u, v) = -f(v, u)$.
- **Flow conservation:** For all $u \in V - \{s, t\}$, we require $\sum_{v \in V} f(u, v) = 0$

flow in equals **flow out** for vertex other than source and sink

The ***value*** of a flow ***f :***

total flow out of the source ($|f| = \sum_{v \in V} f(s, v)$)

maximum-flow problem

In the ***maximum-flow problem***, we are given a flow network G with source s and sink t, and we wish to find a flow of maximum value.

General method
How increase the value of flow

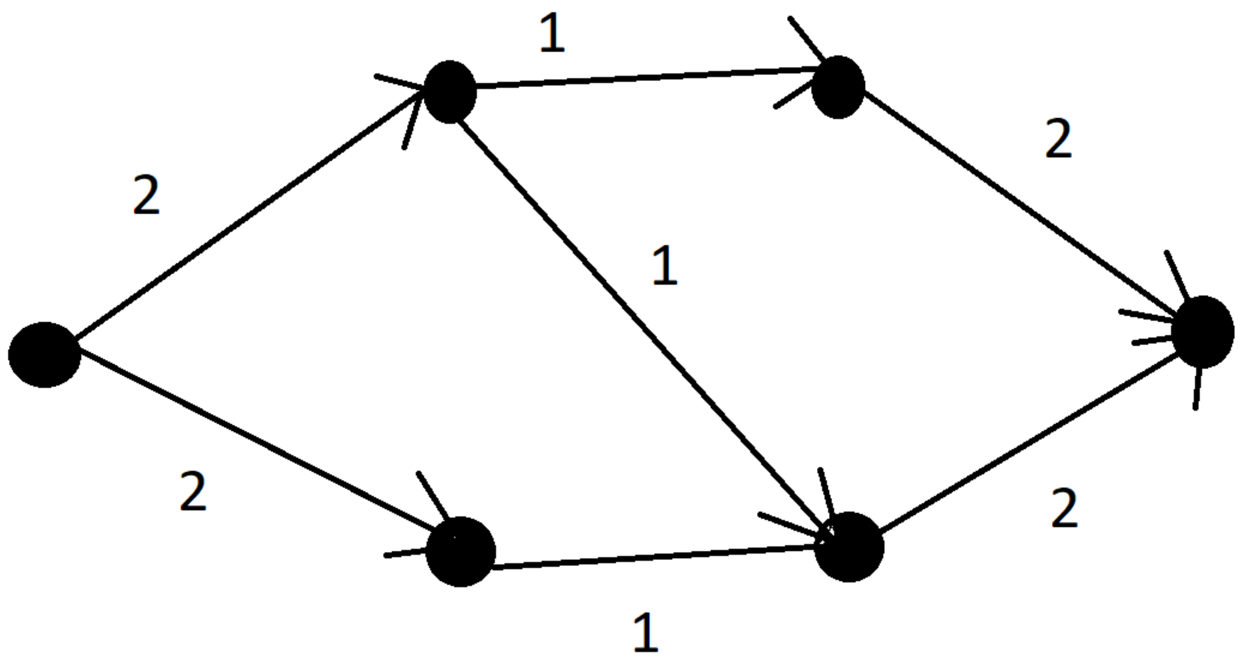
- FORD-FULKERSON-METHOD*(G, s, t)*

initialize flow *f* to 0

while there exists an augmenting path *p*

        do augment flow *f* along *p*

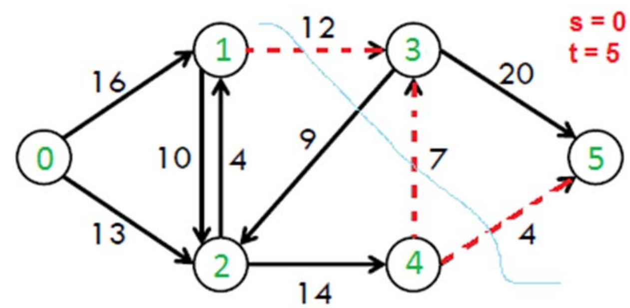
return *f*


2
1
2
1
2
1
2

Max-flow min-cut theorem

If f is a flow in a flow network $G = (V, E)$ with source s and sink t, then the following conditions are equivalent:

1. f is a maximum flow in G.
2. The residual network G_f contains no augmenting paths.
3. $|f| = c(S, T)$ for some cut (S, T) of G.



basic Ford-Fulkerson algorithm

expands on the FORD-FULKERSONMETHOD

runs in time $O(E\,|f^*|)$

```
FORD-FULKERSON(G, s, t)
for each edge (u, v) ∈ E[G]
    do f[u, v] ← 0
       f[v, u] ← 0
while there exists a path p from s to t in the residual network G_f
    do c_f(p) ← min {c_f(u, v) : (u, v) is in p}
       for each edge (u, v) in p
           do f[u, v] ← f[u, v] + c_f(p)
              f[v, u] ← −f[u, v]
```

The Edmonds-Karp algorithm

- implement the computation of the augmenting path p in line 4 with a breadth-first search (each edge has unit distance (weight))

the augmenting path is a *shortest* path from *s* to *t* in the residual network

the running time of the Edmonds-Karp algorithm is $\boldsymbol{O}(\boldsymbol{V}\ \boldsymbol{E}^{2})$

Theorem 26.9

If the Edmonds-Karp algorithm is run on a flow network $G = (V, E)$ with source s and sink t, then the total number of flow augmentations performed by the algorithm is $O(V\ E)$.

Maximum bipartite matching

+solve using flow network

The maximum matching problem

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

a ***matching*** is a subset of edges $M \subseteq E$ such that for all vertices $v \in V$, at most one edge of M is incident on v.

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

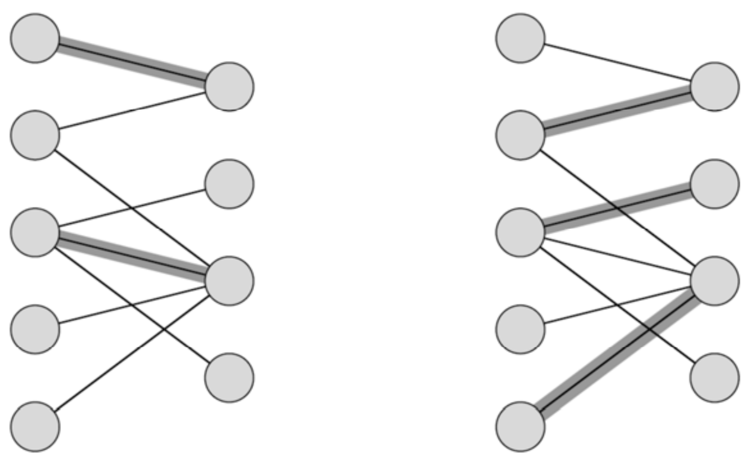
a vertex $v \in V$ is ***matched*** by matching M if some edge in M is incident on v;

otherwise, v is ***unmatched***.

The maximum-bipartite-matching problem

- **Bipartite graphs**: the vertex set can be partitioned into $V = L \cup R$, where L and R are disjoint and all edges in E go between L and R.

(further assume that every vertex in V has at least one incident edge.)



Finding a maximum bipartite matching

- We can **use the Ford-Fulkerson method** to find a maximum matching in an **undirected bipartite graph** $G = (V, E)$ in time **polynomial in |*V*| and |*E*|**.

- First:

define the ***corresponding flow network*** $G' = (V', E')$ for the bipartite graph G

corresponding flow network

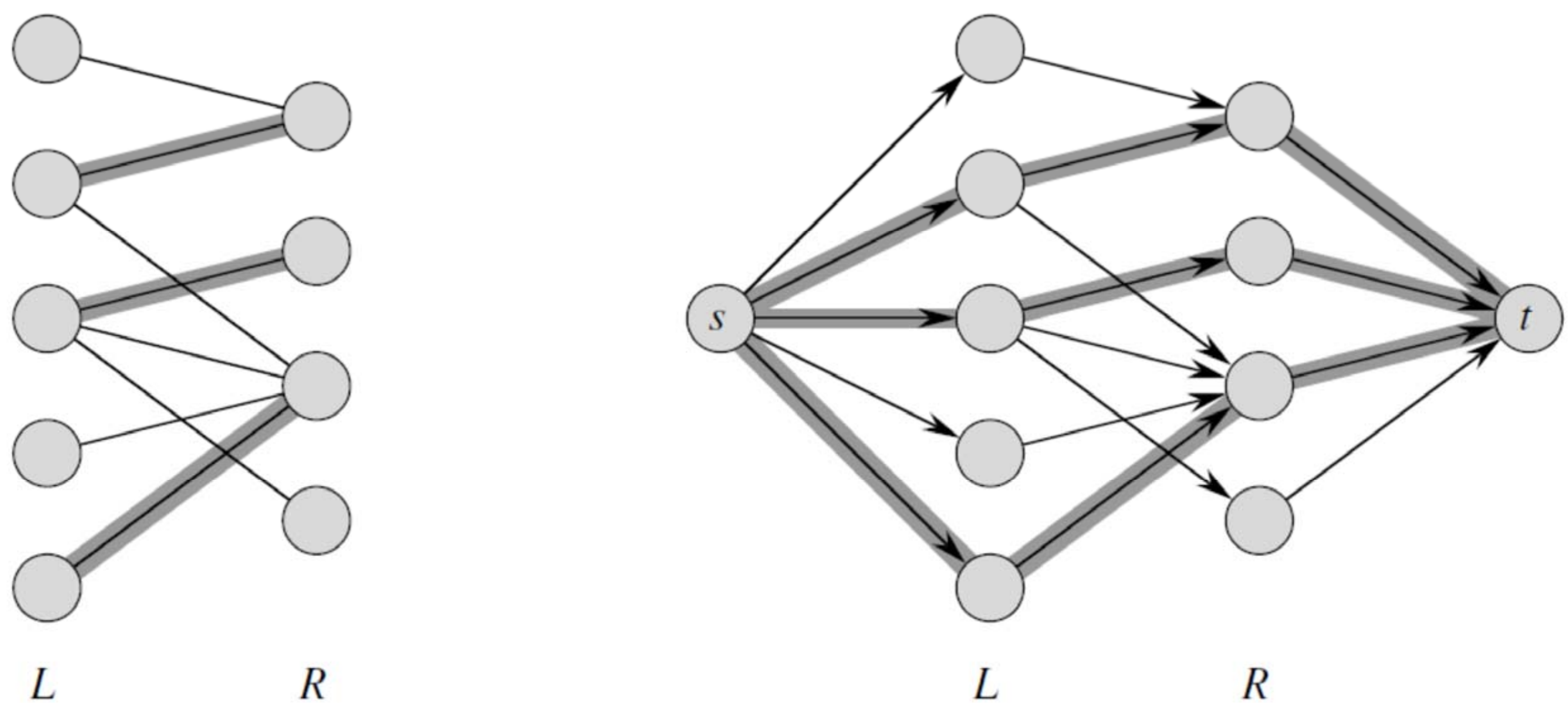
- Add source *s* and sink *t* as new vertices
  - let $V' = V \cup \{s, t\}$.

- The directed edges of G' are :
  - the edges of *E*, directed from L to R ($V = L \cup R$), along with *V* new edges:

$$
\begin{aligned}
E' \; = \; & \{(s, u) : u \in L\} \\
& \cup \{(u, v) : u \in L,\ v \in R,\ \text{and } (u, v) \in E\} \\
& \cup \{(v, t) : v \in R\} \ .
\end{aligned}
$$

  - assign unit capacity to each edge in E'.

The flow network corresponding to a bipartite graph



The order of E' ?

$$|E| \leq |E'|$$

$$|E'| = |E| + |V| \leq 3|E|, \ (|E| \geq |V|/2)$$

$$\boldsymbol{so}\ |\boldsymbol{E'}| = \boldsymbol{\Theta}(\boldsymbol{E}).$$

a **matching** in *G* corresponds directly to
a **flow** in *G*'s corresponding flow network

Lemma 26.10
Let $G = (V, E)$ be a bipartite graph with vertex partition $V = L \cup R$, and let $G' = (V', E')$ be its corresponding flow network. If M is a matching in G, then there is an integer-valued flow f in G' with value $|f| = |M|$. Conversely, if f is an integer-valued flow in G', then there is a matching M in G with cardinality $|M| = |f|$.

maximum matching in a bipartite graph *and* the value of a **maximum flow**

Theorem 26.11 (Integrality theorem)

If the capacity function c takes on only integral values, then the maximum flow f produced by the Ford-Fulkerson method has the property that $|f|$ is integer-valued. Moreover, for all vertices u and v, the value of $f(u, v)$ is an integer.

Corollary 26.12

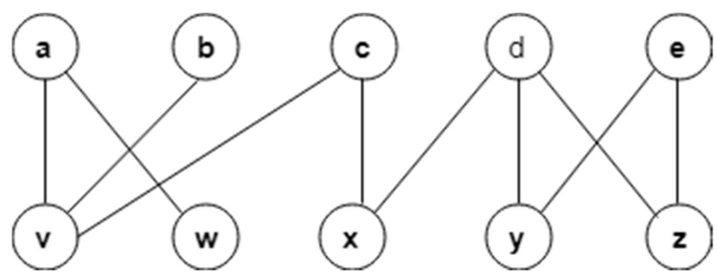
The cardinality of a maximum matching M in a bipartite graph G is the value of a maximum flow f in its corresponding flow network G'.

Whole picture

- given a bipartite undirected graph G,
- creating the flow network G'
- running the Ford-Fulkerson method
- Obtaining a maximum matching *M* from the integer-valued maximum flow

- the value of the maximum flow in G' is $O(V)$.
  - Since any matching in a bipartite graph has cardinality at most $\min(L, R) = O(V)$,
- time complexity of finding a maximum matching in a bipartite graph (since $|E'| = \Theta(E)$.)
  - $\boldsymbol{O(V\ E)}$.

Compute the maximum bipartite matching



Sample problem set:

If all edges in a graph have distinct capacities, is there a unique maximum flow?

- Describe an efficient algorithm to determine whether a given flow network contains a *unique* maximum (*s*, *t*)-flow.
- Describe an efficient algorithm to determine whether a given flow network contains a *unique* minimum (*s*, *t*)-cut.
- Describe a flow network that contains a unique maximum (*s*, *t*)-flow but does not contain a unique minimum (*s*, *t*)-cut.
- Describe a flow network that contains a unique minimum (*s*, *t*)-cut but does not contain a unique maximum (*s*, *t*)-flow.

Let $G = (V, E)$ be a bipartite graph with vertex partition $V = L \cup R$, and let G' be its corresponding flow network. Give a good upper bound on the length of any augmenting path found in G' during the execution of FORD-FULKERSON.

A ***perfect matching*** is a matching in which every vertex is matched.

Let $G = (V, E)$ be an undirected bipartite graph with vertex partition $V = L \cup R$, where $|L| = |R|$. For any $X \subseteq V$, define the ***neighborhood*** of X as $N(X) = \{y \in V : (x, y) \in E$ for some $x \in X\}$, that is, the set of vertices adjacent to some member of X.

Prove ***Hall's theorem***: there exists a perfect matching in G if and only if $|A| \leq |N(A)|$ for every subset $A \subseteq L$.