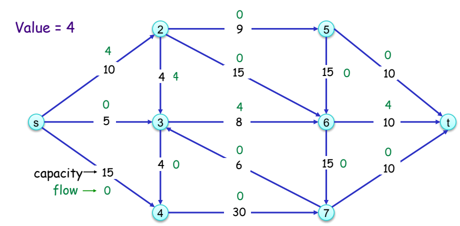
Objectives

- Network Flow Applications
  - Bipartite Matching
  - Circulation

Review: Flows

- The value of a flow f is $v(f) = \sum_{e \text{ out of } s} f(e)$

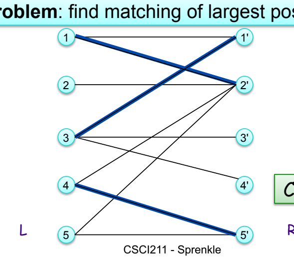


BIPARTITE MATCHING

Bipartite Matching

- Input: undirected, bipartite graph G = $(\underbrace{L \cup R}_{V}, E)$
  - Edges: one end in L, one end in R
- Matching $M \subseteq E$ such that each node appears in at most 1 edge in M.

Problem: find matching of largest possible size

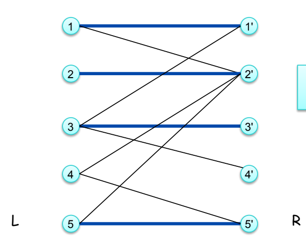


matching
1-2', 3-1', 4-5'

Can we do better?

Bipartite Matching

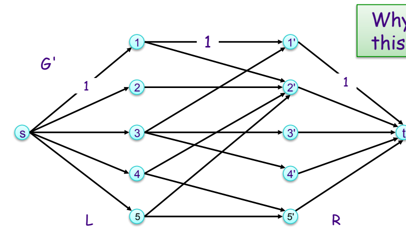
- Input: undirected, bipartite graph G = $(\underbrace{L \cup R}_{V}, E)$
  - Edges: one end in L, one end in R
- Matching $M \subseteq E$ such that each node appears in at most 1 edge in M.



max matching
1-1', 2-2', 3-3' 4-4'

Max Flow Formulation

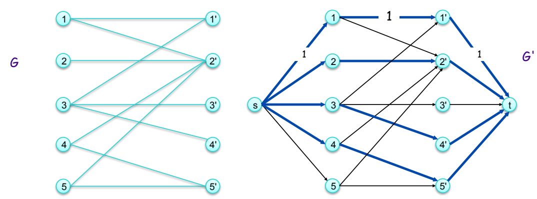
- Create digraph G' = $(L \cup R \cup \{s, t\}, E')$
- Direct all edges from L to R, and assign unit capacity
- Add source s, and unit capacity edges from s to each node in L
- Add sink t, and unit capacity edges from each node in R to t



Why does this work?

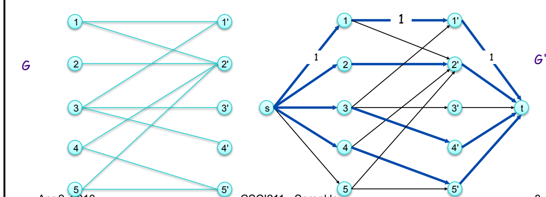
Bipartite Matching: Proof of Correctness

- Theorem. Max cardinality matching in G = value of max flow in G'.
- Proof: Need to show in both directions



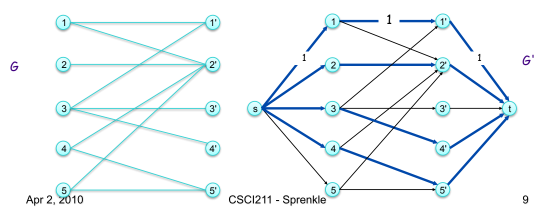
Bipartite Matching: Proof of Correctness

- Theorem. Max cardinality matching in G = value of max flow in G'.
- Pf. →
  - Given max matching M of cardinality *k*.
  - Consider flow f that sends 1 unit along each of *k* paths.
  - *f* is a flow and has cardinality *k*. ▪



Bipartite Matching: Proof of Correctness

- Theorem. Max cardinality matching in G = value of max flow in G'.
- Pf. ←
  - Let f be a max flow in G' of value k.
  - Integrality theorem ⇒ k is integral and can assume f is 0-1.
  - Consider M = set of edges from L to R with f(e) = 1.
    - each node in L and R participates in at most one edge in M
    - |M| = k: consider cut (L ∪ s, R ∪ t) ▪



Perfect Matching

- Def. A matching M ⊆ E is *perfect* if each node appears in *exactly one* edge in M.

How could we figure out if a matching is perfect?

When does a bipartite graph have a perfect matching?

Perfect Matching

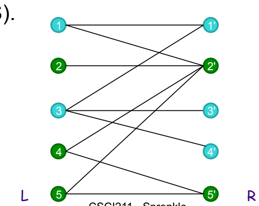
- Def. A matching M ⊆ E is *perfect* if each node appears in *exactly one* edge in M.

When does a bipartite graph have a perfect matching?

- Structure of bipartite graphs with perfect matchings:
  - Clearly we must have |L| = |R|.
  - What other conditions are necessary?
  - What conditions are sufficient?

Perfect Matching

- Let S be a subset of nodes, and let Γ(S) be the set of nodes adjacent to nodes in S.
- Observation. If a bipartite graph G = (L ∪ R, E) has a perfect matching, then |Γ (S)| ≥ |S| for all subsets S ⊆ L.
- Pf. Each node in S has to be matched to a *different* node in Γ(S).

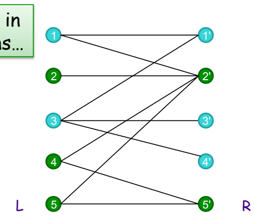


No perfect matching:
S = { 2, 4, 5 }
Γ(S) = { 2', 5' }.

Marriage Theorem [Frobenius 1917, Hall 1935]

- Let G = ($L \cup R$, E) be a bipartite graph with |L| = |R|. Then, G has a perfect matching iff $|\Gamma(S)| \geq |S|$ for *all subsets* $S \subseteq L$.

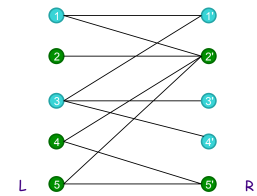
Need to prove in both directions…



No perfect matching:
$S = \{ 2, 4, 5 \}$
$\Gamma(S) = \{ 2', 5' \}$.

Marriage Theorem [Frobenius 1917, Hall 1935]

- Show: G has a perfect matching → $|\Gamma(S)| \geq |S|$ for all subsets $S \subseteq L$.
- Pf. $\Rightarrow$ This was the previous observation.

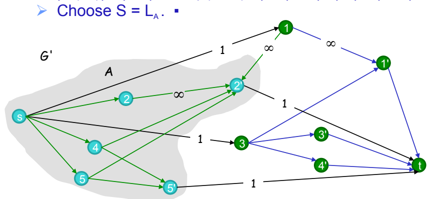


No perfect matching:
$S = \{ 2, 4, 5 \}$
$\Gamma(S) = \{ 2', 5' \}$.

Proof of Marriage Theorem

If max flow < n, then $|\Gamma(S)| < |S|$

- Pf. $\Leftarrow$ Suppose G does not have a perfect matching
  - Formulate as a max flow problem and let (A, B) be min cut in G'
  - By max-flow min-cut, cap(A, B) < | L |
  - Define $L_A = L \cap A$, $L_B = L \cap B$, $R_A = R \cap A$
  - cap(A, B) = $|L_B| + |R_A|$.
  - Since min cut can't use ∞ edges: $\Gamma(L_A) \subseteq R_A$.
  - $|\Gamma(L_A)| \leq |R_A| = cap(A, B) - |L_B| < |L| - |L_B| = |L_A|$.
  - Choose $S = L_A$. ▪



$L_A = \{2, 4, 5\}$
$L_B = \{1, 3\}$
$R_A = \{2', 5'\}$
$\Gamma(L_A) = \{2', 5'\}$

Bipartite Matching: Running Time

- Which max flow algorithm to use for bipartite matching?
  - Generic augmenting path: $O(m\ val(f^*)) = O(mn)$
  - Capacity scaling: $O(m^2 \log C) = O(m^2)$
  - Shortest augmenting path: $O(m\ n^{1/2})$

EXTENSIONS TO MAX FLOW

Power of Max Flow Problem

- Some problems with non-trivial combinatorial searches can be formulated as max flow or min cut in a directed graph

Circulation with Demands

- Circulation with demands
  - Directed graph G = (V, E)
  - Edge capacities c(e), e ∈ E
  - Node supply and demands d(v), v ∈ V

demand if d(v) > 0; supply if d(v) < 0;
transshipment if d(v) = 0

Circulation with Demands

- Circulation with demands
  - Directed graph G = (V, E)
  - Edge capacities c(e), e ∈ E
  - Node supply and demands d(v), v ∈ V

demand if d(v) > 0; supply if d(v) < 0; transshipment if d(v) = 0

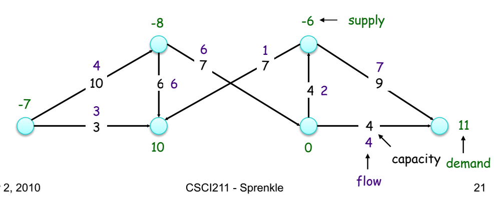
- Def. A *circulation* is a function that satisfies:
  - For each e ∈ E: $0 \le f(e) \le c(e)$ (capacity)
  - For each v ∈ V: $\sum_{e \text{ in to } v} f(e) - \sum_{e \text{ out of } v} f(e) = d(v)$ (conservation)

Circulation problem: given (V, E, c, d), does there exist a circulation?
(Can we satisfy demand with supply?)

Circulation with Demands

- Necessary condition:
  sum of supplies = sum of demands

$$\sum_{v : d(v) > 0} d(v) = \sum_{v : d(v) < 0} -d(v) =: D$$

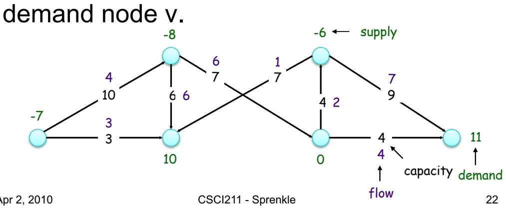


Circulation with Demands

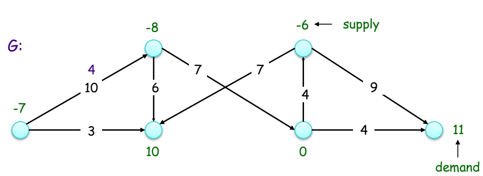
- Necessary condition:
  sum of supplies = sum of demands

$$\sum_{v : d(v) > 0} d(v) = \sum_{v : d(v) < 0} -d(v) =: D$$

- Pf. Sum conservation constraints for every demand node v.



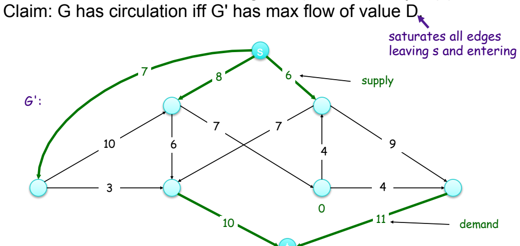
Circulation with Demands: Towards Max Flow Formulation



Circulation with Demands: Max Flow Formulation

- Add new source s and sink t.
- For each v with d(v) < 0, add edge (s, v) with capacity -d(v).
- For each v with d(v) > 0, add edge (v, t) with capacity d(v).
- Claim: G has circulation iff G' has max flow of value D.

saturates all edges leaving s and entering t



Circulation with Demands

- Integrality theorem. If all capacities and demands are integers, and there exists a circulation, then there exists one that is integer-valued.
- Pf. Follows from max flow formulation and integrality theorem for max flow.

Circulation with Demands: Characterization

- Given (V, E, c, d), there does **not** exist a circulation iff there exists a node partition (A, B) such that

$$\Sigma_{v \in B} d_v > \text{cap}(A, B)$$

demand by nodes in B exceeds supply of nodes in B + max capacity of edges going from A → B

- Pf idea. Look at min cut in G'.

Circulation with Demands and Lower Bounds

Force flow to make use of certain edges

- Feasible circulation.
  - Directed graph G = (V, E).
  - Edge capacities c(e) and lower bounds ℓ (e), e ∈ E.
  - Node supply and demands d(v), v ∈ V.

- Def. A *circulation* is a function that satisfies:
  - For each e ∈ E: $0 \le \ell(e) \le f(e) \le c(e)$ (capacity)
  - For each v ∈ V: $\sum_{e \text{ in to } v} f(e) - \sum_{e \text{ out of } v} f(e) = d(v)$ (conservation)

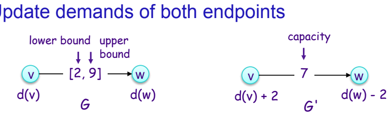
Circulation problem with lower bounds.
Given (V, E, ℓ, c, d), does there exists a circulation?

Circulation with Demands and Lower Bounds

- Model lower bounds with demands
  - Send ℓ(e) units of flow along edge e
  - Update demands of both endpoints

Circulation with Demands and Lower Bounds

- Model lower bounds with demands
  - Send ℓ(e) units of flow along edge e
  - Update demands of both endpoints



- Theorem. There exists a circulation in G iff there exists a circulation in G'. If all demands, capacities, and lower bounds in G are integers, then there is a circulation in G that is integer-valued.
- Pf sketch. f(e) is a circulation in G iff f'(e) = f(e) - ℓ(e) is a circulation in G'.