Applications of Matching

Jon Turner
Computer Science & Engineering
Washington University

www.arl.wustl.edu/~jst
Packet Switch Scheduling

- Internet routers often use “crossbar switches” to transfer packets from inputs to outputs
  - an input can send one packet at a time, and an output can receive one
  - packets transferred in one time step define matching in bipartite graph
  - packets transferred over several time steps define an edge coloring

- To find coloring using fewest colors
  - repeatedly find matching that includes an edge at vertices of maximum degree
Matching Max Degree Vertices

- Find path to extend matching by constructing a single tree rooted at a max degree vertex
  - if selected edge connects to unmatched vertex, we have an augmenting path and root becomes matched
  - if selected edge connects to matched vertex, extend tree; if new leaf has <max-degree, swap edges on path to root
    - this does not increase size of matching, but does match root
Alternate Approach

- Construct flow graph for matching as before
  - augment source/sink edges for max-degree vertices with minimum flow requirement of 1

- To find flow that satisfies min flow requirement
  - find max flow in modified graph
    - add sink/source edge of infinite capacity
    - add new source/sink vertices $s'$, $t'$
    - replace each lower-bound edge $(u,v)$ with ordinary edges $(s',v)$ and $(u,t')$

- Map back to original graph and augment, while retaining min flows
Observations

- Method using min flows can be used to construct matchings that require specific vertices
  - not just max-degree vertices
- Algorithm applies more generally
  - can be used with arbitrary graphs having arbitrary min flow requirements
  - useful in various application settings
- Not all sets of min flow requirements are feasible
  - given infeasible set of requirements, first phase of algorithm terminates without saturating $s'$ edges
- Other edge coloring methods
  - divide-and-conquer algorithm based on Euler partitions achieves running time of $O(m \log \Delta)$
Traveling Salesman Problem

- Given a complete graph with edge costs, $c(u,v)$
  - find min length “tour” that visits every vertex once

- Variants
  - TSP with triangle inequality – $c(u,w) \leq c(u,v) + c(v,w)$
  - Euclidean TSP: vertices are points in a plane, there’s an edge between every pair with length equal to distance between the points
  - asymmetric TSP – directed graph with $c(u,v) \neq c(v,u)$

- TSP is NP-complete, but can be approximated
  - worst-case approx bound of 3/2 with triangle inequality
  - no bound for asymmetric case, but can get near-optimal solutions with high probability for random instances
Approximating TSP Using MST

- If we discard an edge from a TSP solution, we get a spanning tree, so
  \[ \text{MST}(G) \leq \text{TSP}(G) \]

- Consider a depth-first traversal of an MST \( T \), from some arbitrary root
  - list each edge as we go “down” and again as we go back “up”
    - cost of list is \( 2 \text{MST}(G) \)
  - select sub-list by replacing repeat edges with “shortcuts”
    - this yields valid TSP tour and if edge lengths satisfy triangle inequality its total length is at most \( 2 \text{MST}(G) \leq 2 \text{TSP}(G) \)
Improving Approximation

- Can view previous procedure as constructing Eulerian graph
  - where all vertices have even degree
  - any Eulerian graph tour can be converted to a TSP tour using shortcuts
- Finding a better Eulerian graph by connecting odd-degree vertices
  - by finding a perfect matching in graph induced by odd-degree vertices
    - any graph has an even number of these
  - min weight perfect matching $\leq \frac{TSP(G)}{2}$
    - since alternate edges of “shortcut TSP tour” yield two matchings
    - so tour from MST+matching $\leq 1.5 TSP(G)$
Approximating Asymmetric TSP

- TSP tour is a single cycle spanning all vertices
  - can view as perfect matching on bipartite graph
- Any perfect matching defines collection of cycles in original graph
  - so min weight perfect matching provides lower bound on cost of TSP tour
  - for random edge weights, bound is very tight with high probability
Patching Algorithm for TSP

- Construct weighted bipartite graph and find min cost perfect matching
  - using min-cost flow method with costs=weights
  - let $C$ be set of cycles defined by matching
- While $|C| > 1$
  - select two cycles and "patch them" using edge pair that produces smallest increase in cost
    - $(c(u,v)+c(x,y)) - (c(u,v)+c(x,y))$
- For random edge weights
  - initial $C$ has small number of cycles
    - with high probability
  - so small number of patching operations
  - and small increase in cost, yielding near-optimal TSP tour
Applications of TSP

- Vehicle routing
  - selecting route for school bus or mail delivery truck
  - sub-problem of more general “fleet scheduling”

- Job sequencing
  - given set of jobs to be carried out on a complex machine tool, where each job requires some setup
    - setup time for one job depends on previous job
  - use TSP tour to select ordering of jobs to minimize setup

- Data clustering
  - let $A=[a_{ij}]$ were $a_{ij}$ represents the strength of a relationship between two properties
  - permute rows and columns to form “high value blocks”
  - use TSP to find best permutations