

# Topics in Algorithms 2005

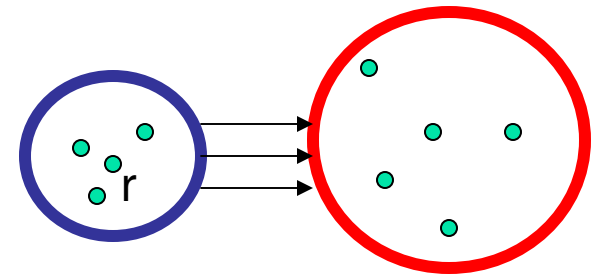
Edge Splitting, Packing Arborescences

---

Ramesh Hariharan

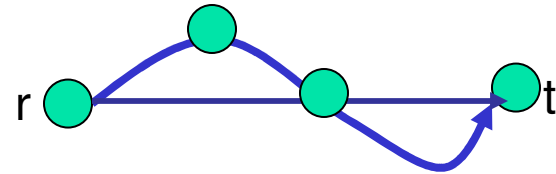
# Directed Cuts

- **r-Cut:** Partition of the vertex set
- **r-Cut Size:** Number of edges from the side containing root  $r$  to the other side
- Remember, we needed to direct an undirected graph to show Menger's theorem



# Edge Disjoining Paths

- **Edge Connectivity between  $r$  and  $t$ :**  
The number of edge-disjoint paths from  $r$  to  $t$ .
- **Exercise: Show Menger's theorem for the directed case for  $r$ - $t$  edge connectivity???**

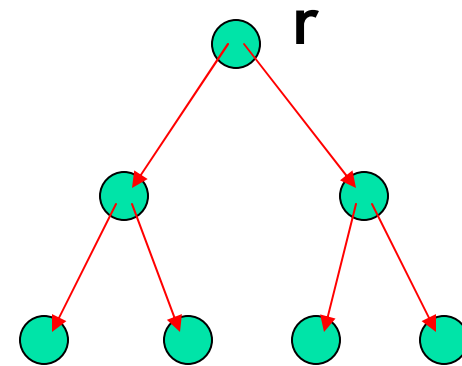




# Arborescences

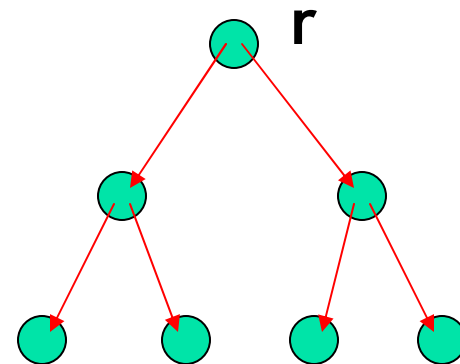
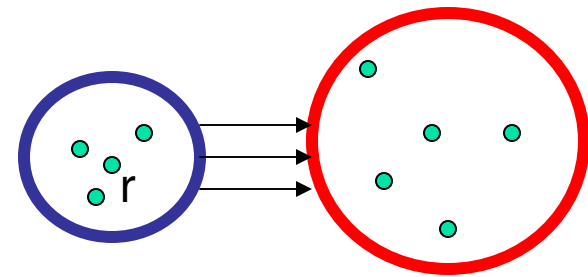
---

- Spanning Trees rooted at  $r$  with all edges directed away from  $r$



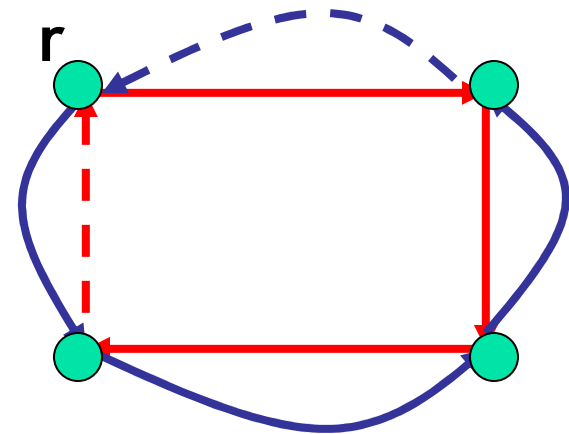
# Edmonds's Arborescence Packing Theorem

- The size of the smallest  $r$ -cut equals the number of edge disjoint arborescences rooted at  $r$
- Clearly, applies to undirected graphs as well by just directing edges as before
- Global version of Menger's Theorem;



# Edmonds's Arborescence Packing Example

- Take a simple cycle with edges running in both directions (e.g., obtained from an undirected cycle)
- Ignoring the dotted lines gives the two required arborescences, one red, one blue

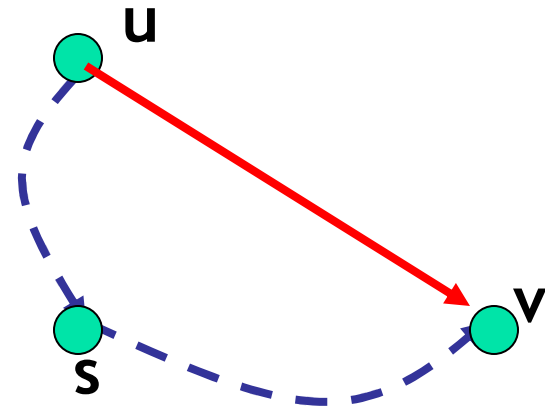




# The Edge Splitting Method

---

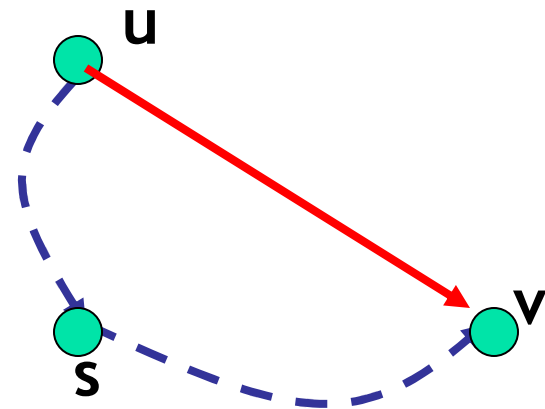
- Provides an inductive tool to prove connectivity properties
- Given vertex  $s$ , there is an incoming edge  $us$  and an outgoing edge  $sv$  such that removing  $us,sv$  and adding  $uv$  maintains connectivities between all pairs of vertices  $x,y$  where  $x,y \neq s$
- Assumption: The graph is Eulerian, i.e., the number of edges going into a cut equals the number of edges going out of a cut



# Proving Edmond's Theorem Using Edge Splitting

Proof of Edmond's theorem for Eulerian Graphs (includes graphs obtained from an undirected graph)

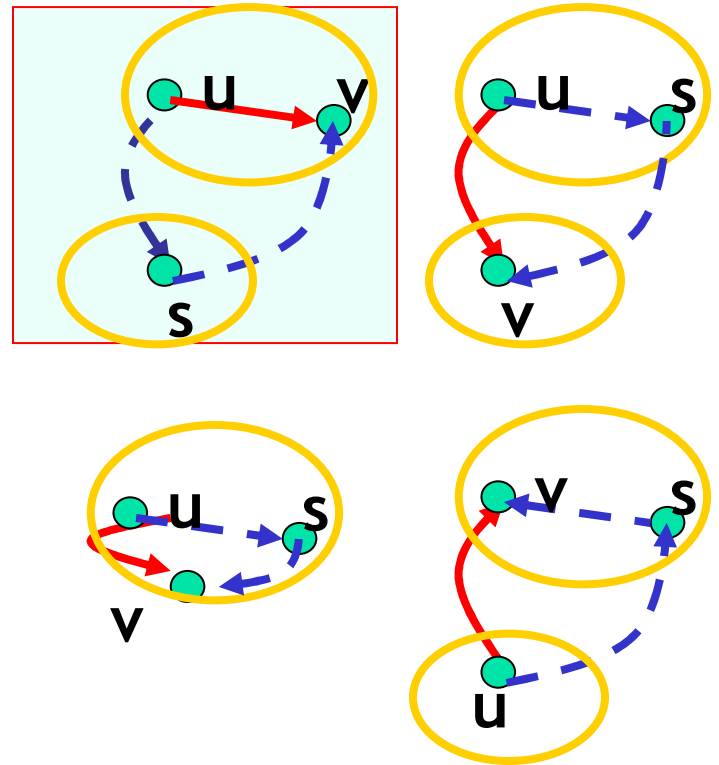
- Start with at least  $k$  paths from root  $r$  to all other vertices
- Repeatedly apply edge splitting on some chosen vertex  $s$ ; note that eulerianness is maintained at each step.
- Now remove  $s$  from the resulting graph and assume what remains satisfies Edmond's theorem by induction (because there are  $k$  paths still between  $r$  and every other vertex, and one less vertex)
- Now link up  $s$  to each of the arborescences as follows. Associate each of the up to  $k$  shortcut edges with a unique arborescence. If the short cut edge  $uv$  occurs in its associated arborescence, replace it with  $us$  and  $sv$ . If  $uv$  does not appear in its associated arborescence, add  $us$  to that arborescence. And if there aren't sufficiently many shortcut edges because you ran out of outedges at  $s$ , the unused inedges play the role of the edge  $us$ .



# Proving Edge Splitting

## Proof of Edge Splitting

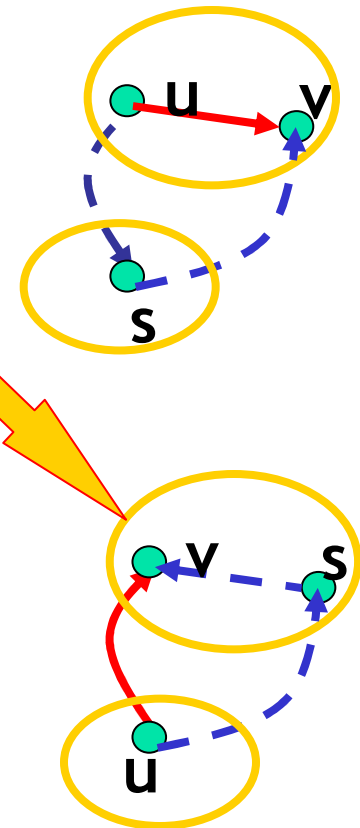
- For a fixed  $us$ , we need to choose  $v$  carefully
- Choose  $v$  so that affected cuts are not tight, i.e., for no pair of vertices  $x \neq s, y \neq s$  does the cut have just as many edges as the connectivity between  $x$  and  $y$
- What cuts are affected?



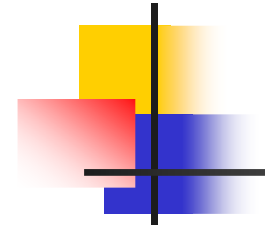
# Proving Edge Splitting

## Proof of Edge Splitting

- Given  $us$ , we need to choose  $v$  so that it is not on the same side as  $u$  in any of the tight cuts separating  $u$  and  $s$
- There could be lots of tight cuts separating  $u$  and  $s$ .
- Consider one particular  $u$ - $s$  tight cut. Can all possible  $v$ 's be on the same side as  $u$  in this tight cut. No. (Why??)
- Can it be that each  $v$  appears in the same side as  $u$  in SOME  $u$ - $s$  tight cut. **This is the problem case.**

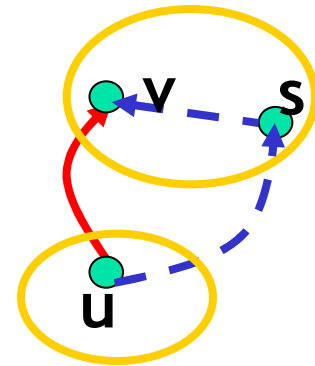


# Proving Edge Splitting



## Proof of Edge Splitting

- There is a unique maximal  $u$ - $s$  tight cut for the eulerian case
- Work with respect to this cut. Not all possible  $v$ 's can be in the same side as  $u$  in this cut. Pick one  $v$  which is outside this cut.





# Proving Edge Splitting

---

## Proof of u-s tight cut maximality for the Eulerian Case

$C(X)$ : the degree of the cut, i.e., number of incoming/outgoing edges into set  $X$

$R(X)$ : the demand of the cut, i.e., max connectivity over all pairs of vertices separated by  $X$

$E(X) = C(X) - R(X) = 0$  for tight sets; this is the excess capacity of a cut

$D(A, B)$ : number of edges between sets  $A$  and  $B$

### SubModularity of Excess: One of these holds

- $E(X) + E(Y) \geq E(X - Y) + E(Y - X) + D(X \cap Y, U - (X \cup Y))$
- $E(X) + E(Y) \geq E(X \cap Y) + E(X \cup Y) + D(X - Y, Y - X)$

Unique Maximality follows: Why?? **Exercise??**

Caveat: **what if maximal u-s tight cut has only s and no other vertices on one side? Exercise??**



# Proving SubModularity of Excess

---

**C() is sub-modular: Both of these hold**

- $C(X)+C(Y) \geq C(X \cap Y) + C(X \cup Y) + D(X-Y, Y-X)$
- $C(X)+C(Y) \geq C(X - Y) + C(Y-X) + D(X \cap Y, U-(X \cup Y))$

**Proof: Exercise??**



# Proving SubModularity of Excess

---

**R()** is super-modular: One of these holds

- $R(X) + R(Y) \leq R(X-Y) + R(Y-X)$
- $R(X) + R(Y) \leq R(X \cap Y) + R(X \cup Y)$

**Proof: Exercise??**

Submodularity of  $C()$  and Supermodularity of  $R$  gives Submodularity of Excess by subtraction



# Algorithmic Constructions of Edmonds Theorem

---

## Gabow's Modification to Edmond's Theorem

- Key Idea:
  - Give up edge directions; edges can go towards or away from the root
  - But insist of overall indegree to be  $k$  for each vertex over all “arbs”
- Call these new “arbs”, directionless arbs.
- Why do this? Because it makes construction easier, as we will see.



# Constructing directionless arbs

---

## Gabow's Algorithm

- Suppose you have constructed  $k-1$  edge-disjoint arbs with total indegree  $k-1$  for each vertex
- The  $k$ th arb is partly constructed, so there are several connected components; each component has exactly one vertex whose in-degree is still  $k-1$  and hasn't reached  $k$  yet.
- Find an unused edge directed into one such vertex  $v$  and connecting this vertex to another connected component. If you can find such an edge for each connected component then the number of components will come down by half. You can repeat this procedure until all components are connected. *If you can't find such an edge, then what??*



# Constructing directionless arbs

---

## Gabow's Algorithm

- $v$  has at least one unused incoming edge (why??)
- This edge stays within the same component, so adding it will create a cycle; this cycle can be broken by pulling out an edge from the cycle. maybe we can add this edge to one of the already constructed arbs and pull out an edge from there and this edge will connect two components in the partly built arb; this can be iterated multiple times, take an edge out from one arb, put it into another arb and pull out and so on.. Call this the closure process
- All indegrees stay the same in the process except for  $v$  which gains 1.
- Directionlessness crucial in this process. Why??

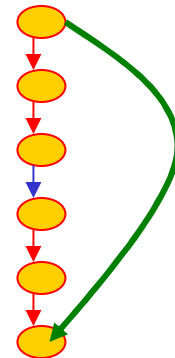


# Constructing directionless arbs

---

## Directionlessness is Crucial

- You can add an edge and pull out any edge in the cycle
- This ensures that vertices in the closure are all contiguous in all the trees on unsuccessful termination





# Constructing directionless arbs

---

## Gabow's Algorithm

What if you can never find an edge which connects two connected components in the new partial arb?

- Show that all vertices encountered in the closure process occur contiguously in all  $k$  arbs. Exercise??
- Now count the in-degrees over all trees to show that the total indegree of this set of vertices is  $k-1$ , which is a contradiction. Exercise??



# Constructing directionless arbs

---

## Gabow's Algorithm: Timing Analysis

- When constructing the last arb, the closure process for each component stays within the nodes of that component, so the time taken over all components in a round  $O(n+m)$ .
- The number of components at least halves in a round
- Total time is  $O((n+m)\log n)$  per arb or  $O(mk\log n)$  on the whole
- By Nagamuchi and Ibaraki's construction,  $m$  is at most  $O(nk)$  (non-trivial)
- So total time is  $O(nk^2 \log n)$



# Constructing arborescences

---

## Going from Directionless Trees to Arborescences

- Gabow shows how to modify the above algorithm to run in  $O(n^2k^2)$  time
- Key bottleneck: Progress on the last arg only one by one and not by halving of components; directions interfere is component reduction.
- Open problem: Show how to construct arbs in sub-quadratic time, even for the case  $k=2$



# Constructing arborescences

---

## Eulerianness

- So Gabow's algorithm is a constructive proof of Edmonds Theorem even when the graph is not Eulerian
- The Edge Splitting proof used Eulerianness. This can be relaxed a bit, see the Bang Jensen reference.



# Extension to Edmond's Theorem

---

Gabow's modification of Edmond's Thm:

If and only if a directed graph has connectivity  $k$  from  $r$  to every other vertex



There exist  $k$  edge disjoint arborescences



There exist  $k$  edge disjoint directionless trees with total indegree  $k$  for each vertex



# Another Extension to Edmond's

---

Extensions beyond the min-cut:

Given a directed eulerian graph,  
if and only if it has connectivity  $c(v)$  from  $r$  to vertex  $v$



There exist edge disjoint (non-spanning) arborescences such that each vertex  $v$  occurs in exactly  $c(v)$  arborescences; **prove this using edge splitting.**



There exist edge disjoint directionless trees with total number of occurrences and total in-degree  $c(v)$  for each vertex  $v$  [Cole,Ramesh]; these can be constructed in  $O(nk^3 \log n)$  time



# Related Problems

---

How fast can one compute the Global MinCut?

- The Karger-Stein algorithm:  $O(n^2 \log n)$
- Gabow's Method using Directionless Trees:  $O(nk^2 \log n)$
- For large networks with small bottlenecks (the practical case), the latter works better

How fast can one compute the Global Steiner MinCut?

- Min-Cut separating a specified subset of vertices
- Using the Cole, Ramesh Method:  $O(nk^3 \log n)$
- Again works well in the practical case of large network, small bottlenecks

# Next Class



---

## Application of Cut Structure

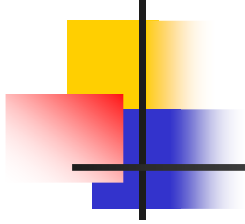
- Constructing networks with sufficient connectivity and fault tolerance
- Intro to convex optimization



# References

---

- Edge Splitting: Lecture Notes by Goemans on the web
- Edge Splitting: Paper by Bang-Jensen, Frank et al on Preserving and increasing edge connectivity of mixed graphs, available on the web
- Gabows paper on computing edge connectivity
- Extensions: Paper on my website on Steiner Edge connectivity



Thank You

---