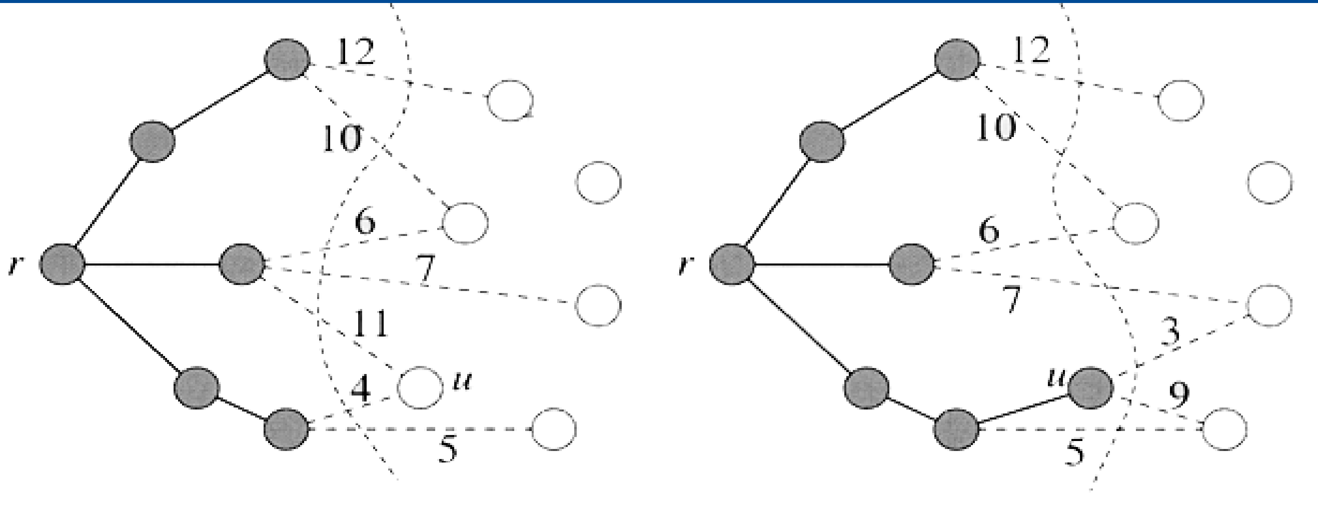
Announcements

- **Weekly Reading Assignment: Chapter 23, 24**
- **Quiz 5 (last one) today**
- **Homework #7 is due on Monday, Dec. 12, 2005**

Intuition behind Prim's Algorithm

- Consider the set of vertices S currently part of the tree, and its complement (V-S). We have a cut of the graph and the current set of tree edges A is respected by this cut.
- Which edge should we add next? *Light edge!*



Basics of Prim 's Algorithm

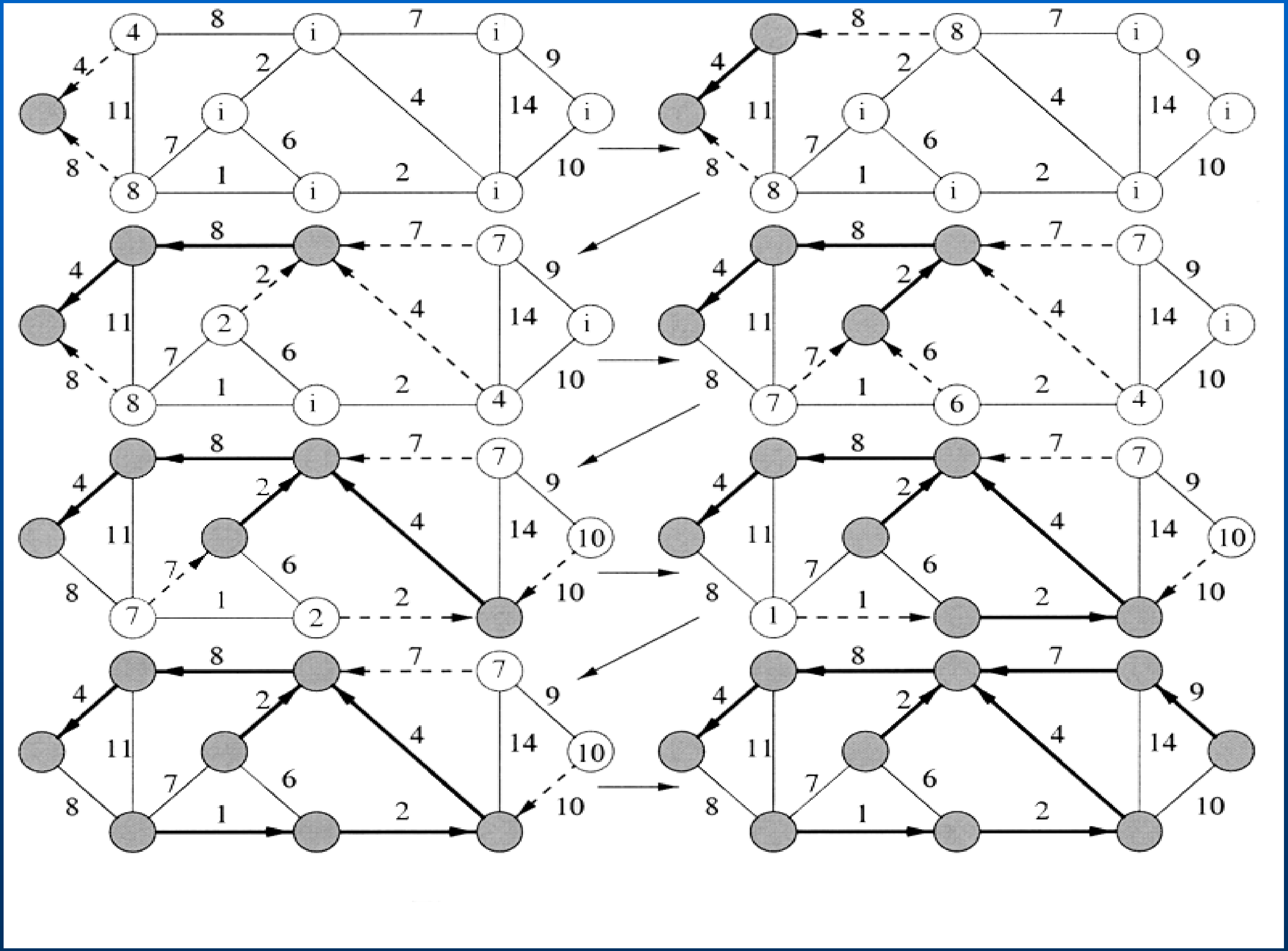
- It works by adding leaves on at a time to the current tree.
  - Start with the root vertex *r* (it can be any vertex). At any time, the subset of edges *A* forms a single tree. *S* = vertices of *A*.
  - At each step, a light edge connecting a vertex in *S* to a vertex in *V*- *S* is added to the tree.
  - The tree grows until it spans all the vertices in *V*.
- Implementation Issues:
  - How to update the cut efficiently?
  - How to determine the light edge quickly?

Implementation: Priority Queue

- **Priority queue implemented using heap can support the following operations in $O(lg\ n)$ time:**
  - Insert (Q, u, key): Insert u with the key value key in Q
  - u = Extract_Min(Q): Extract the item with minimum key value in Q
  - Decrease_Key(Q, u, new_key): Decrease the value of u's key value to new_key

- **All the vertices that are *not* in the S (the vertices of the edges in A) reside in a priority queue Q based on a key field. When the algorithm terminates, Q is empty. $A = \{(v, \pi[v]): v \in V - \{r\}\}$**

Example: Prim's Algorithm



MST-Prim(G, w, r)

$Q \leftarrow V[G]$
for each vertex $u \in Q$ — // initialization: $O(V)$ time
    do $key[u] \leftarrow \infty$
$key[r] \leftarrow 0$ — // start at the root
$\pi[r] \leftarrow \mathrm{NIL}$ — // set parent of r to be NIL
while $Q \neq \emptyset$ — // until all vertices in MST
    do $u \leftarrow$ Extract-Min(Q) — // vertex with lightest edge
        for each $v \in adj[u]$
            do if $v \in Q$ and $w(u,v) < key[v]$
                then $\pi[v] \leftarrow u$
                    $key[v] \leftarrow w(u,v)$ — // new lighter edge out of v
                    decrease_Key(Q, v, $key[v]$)

Analysis of Prim

- **Extracting the vertex from the queue: $O(\lg n)$**
- **For each incident edge, decreasing the key of the neighboring vertex: $O(\lg n)$ where $n = |V|$**
- **The other steps are constant time.**
- **The overall running time is, where $e = |E|$**

$$T(n) = \sum_{u \in V}(\lg n + \deg(u) \lg n) = \sum_{u \in V} (1 + \deg(u)) \lg n$$
$$= \lg n \,(n + 2e) = O((n + e) \lg n)$$

Essentially same as Kruskal's: $O((n+e) \lg n)$ time

Correctness of Prim

- Again, show that every edge added is a safe edge for *A*
- Assume (*u*, *v*) is next edge to be added to *A*.
- Consider the cut (*A*, *V*-*A*).
  - This cut respects *A* (why?)
  - and (*u*, *v*) is the light edge across the cut (why?)
- Thus, by the MST Lemma, (*u*,*v*) is safe.