PRIM’S ALGORITHM for MST
(R. Prim 1957, but also V. Jarnik 1930)

Uses basic principle:
Given a subtree $T$ of MST, the "lightest" edge connecting to a vertex not in $T$
can be added to $T$.

Grow one tree, incrementally adding one edge (& vertex)
PRIM'S ALGORITHM for MST

Every vertex not in $T$ has a score $= \text{lightest edge weight connecting it to } T$

Identify lightest edge crossing cut:
1) identify min-score vertex, $x$
2) identify lightest edge from $x$ to $T$

Brute force: $O(v)$ per MST edge
PRIM'S ALGORITHM for MST

Update scores when $x$ joins $T$:
For each neighbor $v_i$ of $x$
  if $v_i$ not in $T$
    $c = \text{score}(v_i)$
    $\text{score}(v_i) \leftarrow \min \{c, w(x, v_i)\}$

Need to extract min score & decrease scores. How?
PRIM'S ALGORITHM for MST

For a detailed example of Prim's algorithm on this graph, please see full version of class notes.

Summary follows.
PRIM’S ALGORITHM for MST

1) start w/ any vertex s; set \( w(s) = 0 \)
2) set \( w(\neq s) = \infty \) & put all in pr. queue
PRIM'S ALGORITHM for MST

1) start w/ any vertex \( s \); set \( w(s) = 0 \)
2) set \( w(\neq s) = \infty \) & put all in pr. queue
3) while pr. queue not empty

\[ \forall \] rounds
PRIM'S ALGORITHM for MST

1) start w/ any vertex \( S \); set \( w(s) = 0 \)
2) set \( w(\neq s) = \infty \) & put all in pr. queue
3) while pr. queue not empty
   \( x \): extract-min & add edge to \( T \)
   mark \( x \rightarrow \) in \( T \).

("add edge" \( \rightarrow \) find an edge from \( x \rightarrow T \), w/ min weight)
(if \( x = s \), no edge to add)
PRIM’S ALGORITHM for MST

1) start w/ any vertex s; set \( w(s) = 0 \)
2) set \( w(\neq s) = \infty \) & put all in pr.queue
3) while pr.queue not empty
   x: extract-min & add edge to T
   mark \( x \rightarrow \) in T.
PRIM’S ALGORITHM for MST

1) Start w/ any vertex \( s \); set \( w(s) = 0 \)
2) set \( w(\neq s) = \infty \) & put all in pr.queue
3) while pr.queue not empty
   \( x \): extract-min & add edge to \( T \)
   mark \( x \rightarrow \) in \( T \).
   for each unmarked neighbor \( q \) of \( x \)
   if \( w(q) > w(q,x) \) then decrease.
PRIM'S ALGORITHM for MST

1) start w/ any vertex s; set $w(s) = 0$
2) set $w(\neq s) = \infty$ & put all in pr. queue
3) while pr. queue not empty
   \[ x: \text{extract-min} \; & \; \text{add edge to } T \]
   mark $x \rightarrow$ in $T$.
   for each unmarked neighbor $q$ of $x$
   if $w(q) > w(q,x)$ then decrease.

$|V|$ rounds
\[ \sum_{x \in V} (O(\log V) + O(\text{degree}(x))) \]
\[ = O(V \log V) + O(E) \]
\[ \sum_{x \in V} O(\text{degree}(x)) \cdot O(\log V) \]
\[ = O(E) \cdot O(\log V) \]

Using adjacency list
\[ \text{TOTAL} = O(E \log V) \]
PRIM'S ALGORITHM for MST

with Fibonacci heap
(beyond scope of COMP 160)

$N$ rounds

$\sum_{x \in V} (O(\log V) + O(\text{degree}(x)))$

$= O(V\log V) + O(E)$

$\sum_{x \in V} \text{degree}(x) \cdot O(\log V)$

$= O(E) \cdot O(\log V)$

1) start with any vertex $s$; set $w(s) = 0$
2) set $w(\neq s) = \infty$ & put all in pr. queue
3) while pr. queue not empty
   x: extract-min & add edge to $T$
   mark $x \rightarrow$ in $T$.
   for each unmarked neighbor $q$ of $x$
   if $w(q) > w(q, x)$ then decrease.

Using adjacency list

$\text{TOTAL} = O(E + V\log V)$
PRIM'S ALGORITHM for MST

Using adj. matrix
w/ weighted entries

& no pr. queue

\[ V \text{ rounds} \]

scan array

scan row(x) in matrix

1) start w/ any vertex \( s \); set \( w(s) = 0 \)
2) set \( w(\neq s) = \infty \) & put all in pr. queue

3) while pr. queue not empty
   \[ \exists v \text{ not in } T \]
   \[ O(V) \{ \]
   \[ x: \text{ extract-min } \& \text{ add edge to } T \]
   \[ \text{mark } x \rightarrow \text{ in } T. \]
   \[ O(V) \{ \]
   \[ \text{for each unmarked neighbor } q \text{ of } x \]
   \[ \text{if } w(q) > w(q,x) \text{ then decrease.} \]

\[ O(V^2) \text{ time } \& \text{ space} \]