PRIM'S ALGORITHM for MST

R. PRIM - 1957

(V. JARNIK - 1930)
PRIM'S ALGORITHM for MST

Uses basic principle:

Given a subtree $T$ of MST, the "lightest" edge connecting to a vertex not in $T$ can be added to $T$. 
PRIM'S ALGORITHM for MST

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 (and vertex)
PRIM'S ALGORITHM for MST

Every vertex not in $T$ has a score $=$ lightest edge weight connecting it to $T$. 
PRIM'S ALGORITHM for MST

Every vertex not in T has a score = lightest edge weight connecting it to T

Identify lightest edge crossing cut: How?
PRIM'S ALGORITHM for MST

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

Identify lightest edge crossing cut:
1) identify min-score vertex,
PRIM’S ALGORITHM for MST

Every vertex not in $T$ has a score = 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 $T$
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

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
... but we must still update scores after adding $x$ to $T$
**PRIM'S ALGORITHM** for MST

Update scores when $x$ joins $T$:

For each neighbor $v_i$ of $x$

if $v_i$ not in $T$

$$c = \text{score}(v_i)$$

$$\text{score}(v_i) \leftarrow \min \{c, w(x, v_i)\}$$

new option
**PRIM'S ALGORITHM for MST**

Update scores when \( x \) joins \( T \):

For each neighbor \( v_i \) of \( x \)

if \( v_i \) not in \( T \)

\[
\begin{align*}
C &= \text{score}(v_i) \\
\text{score}(v_i) &\leftarrow \min \{ C, w(x,v_i) \}
\end{align*}
\]

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

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM’S ALGORITHM for MST
PRIM'S ALGORITHM for MST

Priority queue
PRAIM'S ALGORITHM for MST
PRIM'S ALGORITHM for MST
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST
PRIM'S ALGORITHM for MST

Priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

Priority queue
PRIM'S ALGORITHM for MST

Priority queue
PRIM'S ALGORITHM for MST
PRIM'S ALGORITHM for MST
PRIM'S ALGORITHM for MST

Priority queue
PRIM'S ALGORITHM for MST

priority queue
**PRIM'S ALGORITHM for MST**

Priority queue
PRIM'S ALGORITHM for MST

[Diagram of a network with labeled nodes and edges, and a priority queue is shown.]

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

Note: a vertex can keep track of its "best" edge, so when it is added to $T$, we don't need to find min.
PRIM'S ALGORITHM for MST

Priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

current best for vertex 24

priority queue
PRIM’S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST

Priority queue
PRIM'S ALGORITHM for MST

priority queue
PRIM'S ALGORITHM for MST
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

$|V|$ 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
   - extract-min & add edge to \( T \)
   - mark \( x \rightarrow \) in \( T \).

("add edge" \( \rightarrow \) find an edge from \( x \) to \( T \), 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(≠s) = ∞ & put all in pr.queue
3) while pr.queue not empty
   x: extract-min & add edge to T
   mark x → 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
   - \( x \): 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
   $x$: 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

$N$ rounds

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

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
   \( x: \text{extract-min} \) & add edge to \( T \)
   mark \( x \to \) 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(≠s) = ∞ & put all in pr.queue
3) while pr.queue not empty
   x: extract_min & add edge to T
   mark x → in T.
   for each unmarked neighbor q of x
   if w(q) > w(q,x) then decrease.
PRIM'S ALGORITHM for MST

\( \sum_{x \in V} (O(\log V) + O(\text{degree}(x))) = O(V \log V) + O(E) \) rounds

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
   - $x$: extract-min & add edge to $T$
   - mark $x \to$ 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 $w(s) = \infty$ & 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

**$|V|$** rounds

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

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

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

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

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.

**Total cost?**
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
   \[ x: \text{extract-min} \ & \text{add edge to } T \]
   \[ \text{mark } x \rightarrow \text{ in } T. \]
   \[ \text{for each unmarked neighbor } q \text{ of } x \]
   \[ \text{if } w(q) > w(q,x) \text{ then decrease.} \]

$|V|$ rounds

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

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

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

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

dominates

Using adjacency list

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

with Fibonacci heap
(beyond scope of COMP160)

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

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
   \[ x: \text{ extract-min } \] & 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

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

Using adj. matrix w/ weighted entries & no pr. queue

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 \( \exists v \) not in \( T \)
   \( x: \) 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 array
3) While pr. queue not empty: \( \exists v \) not in \( T \)

\[ \forall v \in V \{ \] 
\[ x: \text{extract-min} \& \text{add edge to } T \] 
\[ \text{mark } x \to \text{ in } T. \] 
\[ \text{for each unmarked neighbor } q \text{ of } x \] 
\[ \text{if } w(q) > w(q, x) \text{ then decrease.} \] 
\[ \text{O}(V^2) \text{ time & space} \]