Modified Minimimum Spanning Forest

This computation is used to find a minimum spanning forest connecting nodes.

It is currently being used in a least-cost supply model where demand at nodes can be met via "internal" supply (remains a standalone node) or via a networked supply (where it becomes part of a tree in a forest).

Function definitions

Distance (weight) between nodes of an edge e (e.g. euclidean distance):

$E_w(e) \to \mathbb{R}$

Weight of a node v:

$V_{max}(v) \to \mathbb{R}$

Minimum edge distance (weight) between subcomponents of a parent component c:

$E_{min}(c_i, c_j) \to \mathbb{R}$

Components connected by edge in a tree:

$E_c(e) \to c_i, c_j$

Component c lookup for a node v:

$C[v] = c$

Weight of a component tree c. This will be used in a constraint that determines the maximum length of an edge that can connect to this component.

$C_{max}(c) = \sum_{\forall v \in V(c)} V_{max}(v) - \sum_{\forall e \in E(c)} E_w(e)$

The above formula can be derived from the fact that as each edge e, with nodes v, u is added to a component c, $C_{max}(c)$ is updated via:

$C_{max}(c) = (C_{max}(C[v]) + C_{max}(C[u])) - E_w(e)$

Mathematical formulation

Given F is the set of all spanning forests in G, we want the set of components of a forest $C \in F$ that minimizes the sum of edge weights while meeting certain constraints among its subcomponents.

$MSF = min \sum_{c \in C} \sum_{e \in c} E_w(e)$

$s.t.$

$C_{max}(c_i) > E_{min}(c_i, c_j) \wedge C_{max}(c_j) > E_{min}(c_i, c_j) \ \forall c_i, c_j = E_c(e), \forall e \in C$

Modified Kruskal Minimum Spanning Forest Algorithm

Given graph $G(V, E)$

```
modKruskal(G)
  MSF = NULL
  for v in V
    C[v] = v

  for e(u, v) in sorted(E, E_w(e))
    if (C[u] != C[v] &&
        C_max(C[v]) > E_w(e) && C_max(C[v]) > E_w(e))

      # also update Component lookup structure
      MSF = MSF U (u, v)

  return MSF
```