

LOCAL  $p$ -MEDIAN PROBLEM  
 SA405, FALL 2012  
 INSTRUCTORS: FORAKER AND PHILLIPS

We now describe the *local  $p$ -Median problem*, a facility location problem that is similar to the *general  $p$ -Median problem*, formulated in the text on p. 136 as Integer Program 4.5, but has a crucial difference in that demand can only be satisfied “locally”. Let  $\mathcal{G} = (\mathcal{V}, \mathcal{E})$  be a given undirected graph. The crucial assumption is that a demand node  $i \in \mathcal{V}$  can only be served by a facility at node  $j \in \mathcal{V}$  if either  $(j, i) = (i, j) \in \mathcal{E}$ , or  $i = j$ .

Also given are:

- a set of *demand nodes*, denoted by  $I \subseteq \mathcal{V}$ ;
- a population at each demand node, denoted by  $h_i$ ;
- a set of possible *location nodes* where facilities might be built, denoted by  $J \subseteq \mathcal{V}$ ;
- a positive integer  $p$ , which is the maximum number of facilities that can be built;
- a minimum amount of population whose demand must be satisfied, denoted by  $T$ ;
- arc distances, denoted by  $c_{ij}$ ; and
- for each  $i \in I$ , a set of *neighborhood nodes*, denoted by  $N_i$ , and defined formally by

$$N_i = \{j \in J \mid (i, j) = (j, i) \in \mathcal{E} \text{ or } i = j\},$$

i.e.,  $N_i$  consists of location nodes that can serve  $i$ .

We note that  $c_{ij}$  are **different** than the book’s  $d_{ij}$  as  $d_{ij}$  represent shortest path distances, whereas  $c_{ij}$  are arc distances (and may even be greater than the shortest path distances).

The problem is as follows: Where should the  $p$  facilities be located so as to minimize the total population-weighted distance between demand nodes served and facilities? For the formulation, we use two sets of binary variables,

$$y_{ij} = \begin{cases} 1 & \text{if } i \text{ is served by } j \\ 0 & \text{otherwise.} \end{cases}$$

for every demand node  $i$  and location node  $j \in N_i$ , and

$$x_j = \begin{cases} 1 & \text{if a facility is open at location node } j \\ 0 & \text{otherwise.} \end{cases}$$

for every location node  $j \in J$ . The formulation is

$$\begin{aligned} \min \quad & \sum_{i \in I} h_i \left( \sum_{j \in N_i} c_{ij} y_{ij} \right) & (a) \\ \text{s.t.} \quad & \sum_{j \in J} x_j \leq p & (b) \\ & y_{ij} \leq x_j & \forall i \in I, \forall j \in N_i \quad (c) \\ & \sum_{i \in I} h_i \left( \sum_{j \in N_i} y_{ij} \right) \geq T & (d) \\ & \sum_{j \in N_i} y_{ij} \leq 1 & \forall i \in I \quad (e) \\ & x_j, y_{ij} \in \{0, 1\} & \forall i \in I, \forall j \in N_i. \quad (f) \end{aligned}$$

The objective, (a), multiplies the population by the distance from the facility location used (if any) and adds these products to calculate the population-weighted sum of distances. Note that the order of summation matters as  $N_i$  depends on the index  $i$ , which is also true in the constraints (c), (d), and (f). The upper bound on facilities is enforced in constraint (b) – note that in the book, the assumption is that **exactly**  $p$  facilities must be opened. Constraints (c) enforce that if demand node  $i$  is satisfied by a facility at location node  $j$ , then a facility must be open at location  $j$ . The lower bound on demand satisfied is enforced by constraint (d). Constraints (e) ensure that at most one facility is counted as serving a given demand node. Constraints (f) enforce that the variables are binary.

Some notes on the formulation and problem:

- The crucial differences between the local  $p$ -Median problem and the  $p$ -Median problem are that in the local problem a facility at location node  $j$  can only serve demand node  $i$  if  $(i, j) = (j, i) \in \mathcal{E}$  or  $i = j$ . In the general  $p$ -Median problem, all demand nodes can be served from any location node so long as a facility is open there. Thus, in the general  $p$ -Median, the book assumes **every** demand node is satisfied. In the local  $p$ -Median, we assume that a minimum amount of demand must be satisfied.
- Note that bounds on meaningful  $p$  and  $T$  can be determined from the Set-Cover Location problem (p. 132, Integer Program 4.2), and Maximal Covering Location problem (p. 133, Integer Program 4.3), respectively.