```
int P = 2;
int L_count = 3;
int K_count = 3;

range P_num = 1..P;
range Locations = 1..L_count;
range ReferencePoints = 1..K_count;

float L[Locations][1..2] = [[2,6],[1,2],[3,3]];
float K[P_num][ReferencePoints][1..2] = [[[2,5.3],[1,2],[]], [[2,3], [2,8],[2,6]]];
int x[Locations] = [0, 0, 0];

dvar boolean xy[P_num][Locations];
dvar float+ total_distance;

minimize total_distance;

subject to {
   sum(p in P_num, i in Locations) xy[p][i] == P;

   forall(p in P_num) {
      sum(i in Locations) xy[p][i] == 1;
   }

   forall(i in Locations) {
      sum(p in P_num) xy[p][i] <= 1;
   }

   total_distance == sum(p in P_num, i in Locations) (
      max(j in ReferencePoints) (xy[p][i] * pow(pow(L[i][1] - K[p][j][1], 2) + pow(L[i][2] - K[p][j][2], 2), 0.5)));
}

execute DISPLAY {
   for (i in Locations) {
      for (p in P_num) {
         if (xy[p][i] == 1) {
            x[i] = 1;
         }
      }
   }

   writeln("Minimalna suma maksymalnych odległości: ", total_distance);
   write("Wybrane lokacje: ");
   for (i in Locations) {
      write(x[i], " ");
   }
//    write(xy);
}
```