Stefan Pesko *

MINIMAL TOTAL AREA CONVEX SET PARTITIONING PROBLEM

The linear binary model for finding set partitioning of the points in the plane is studied.

We present heuristic which generates partitioning of points to clusters for a given number of seeds with searched seeds of clusters in a plane. For this cluster the convex hulls are constructed via Graham's scan algorithm. This approach is demonstrated on the real instance of Florida area with 235 points for 10 and 13 clusters.

Keywords: Convexity constraints, set partitioning problem, location problem, convex hull

1. Introduction

The set-partitioning problem is a well known NP-hard combinatorial problem. A real-world problem of this class for the vehicle routing problem in Florida area was formulated by Juricek [5]. He requires partitioning of customers to clusters. The clusters should be constructed as "optimal" areas for 1- vehicle routing. An intuitive criterion for the well-formed area of the cluster for one vehicle is the smallest convex polygon that encloses all points of cluster – convex hull.

We present two problems:
- *Convex set partitioning problem:* A set of points in a plane have to be partitioned to a given number of clusters so that every two convex hulls of cluster members have empty intersection and the total area of cluster's convex hulls is minimal.
- *Convex location problem:* A set of q seeds of clusters are searched with a bounded capacity of cluster and convex constraint for a pair of assigned points.

2. Convex hull problem

Finding the convex hull of a set of points in plane is one of most interesting elementary problems in computational geometry [1].

Let $P = [p_1, p_2, ..., p_n]$, $n \geq 3$ be a set of distinct points in the Euclidean plane. The *convex hull* of P is the minimal convex area (bordered by points in P) that contains every point of P. We will denote this area by $convh$ (P). For convenience we say that a point $p \in P$ is a boundary point of the convex hull of P if p is a boundary vertex of the convex hull of P. The *convex hull problem* is the problem of computing the convex hull of P and reporting the boundary points of the convex hull in the order in which they appear on the hull – we will note this set $convh$ (P).

Fig. 1 shows an example of the convex hull of set $P = [p_1, p_2, ..., p_{10}]$. The points $p_4, p_6, p_7, p_8, p_9 \in (P - \underline{convh}(P))$ lie in the convex hull but they are not boundary points and the points $p_1, p_2, p_3, p_5, p_{10} \in \underline{convh}(P)$ are boundary points of the convex hull.
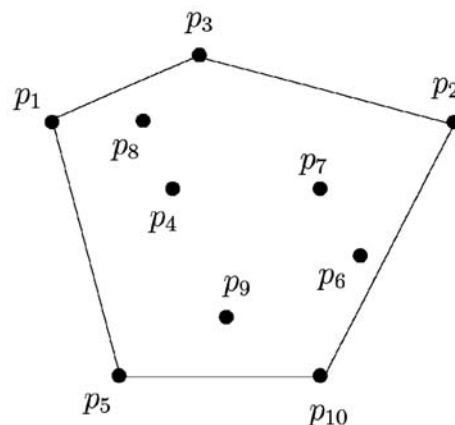

Fig. 1 The points on the convex hull $\underline{convh}$ (P) = [$p_1, p_3, p_2, p_{10}, p_5$]

As early as 1972, Graham gave a convex hull algorithm with $O(nlog(n))$ worst-case running time. In our application we used the Python program [3] implementing Graham's scan algorithm.

3. Convex set partitioning problem

The problem of partitioning customers (represented by the points on the plane) to the convex hulls of vehicle areas can be formulated as the following optimization problem (CSP):

Let us have the set of n points $P = [p_1, p_2, ..., p_n]$ in the plane and the integer q where $n \geq 6$ and $2 \leq q \leq [n/3]$. The goal is to

---

* **Stefan Pesko**
  Department of Mathematical Methods, Faculty of Management Science and Informatics, University of Zilina, Slovakia,
  E-mail: stefan.pesko@fri.uniza.sk

find a set partitioning of the set P to q partitions - clusters of points - $P(q) = (P_1, P_2, ..., P_q)$ such that every two clusters have empty intersection and the total area of the convex hulls of partitioning members is minimal.

Fig. 2 shows an example of the convex set partitioning of set P to three clusters P_1, P_2, P_3 of points. Every two convex hulls
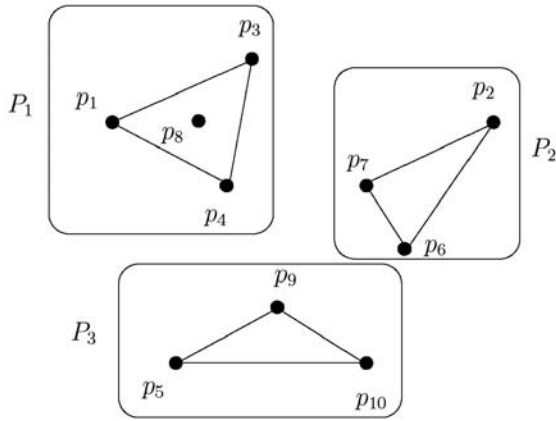

Fig. 2 Convex set partitioning P(3) = (P_1, P_2, P_3) of points **P**

of partitioning have empty intersection i.e. $\overline{convh}(P_i) \cap \overline{convh}(P_j) = = \varnothing$ for $1 < i < j < 3$ and the total area of the convex hulls $|\overline{convh}(P_1)| + |\overline{convh}(P_2)| + |\overline{convh}(P_3)|$ is minimal, where $|p|$ denotes the area o the polygon p

Let $\mathcal{H} = \{H_1, H_2, ..., H_m\}$ be the set of all subsets of the boundary points from P which lie on the convex hulls. We will use the abbreviate notations $M = \{1, 2, ..., m\}$ and $N = \{1, 2, ..., n\}$. We can now define the binary variables

$$x_j = \begin{cases} 1 & \text{if solution has clusters with convex hull of points } H_j \in \mathcal{H} \\ 0 & \text{if otherwise} . \end{cases}$$

We define the point-to-cluster incidence matrix $A = (a_{ij})$ where

$$a_{ij} = \begin{cases} 1 & \text{if } p_i \in \underline{convh}(H_j), \\ 0 & \text{if otherwise} \end{cases}$$

and the cluster-to-cluster incidence matrix $B = (b_{ij})$ where

$$b_{ij} = \begin{cases} 1 & \text{if } \overline{convh}(H_i) \cap \overline{convh}(H_j) \neq \emptyset, \\ 0 & \text{otherwise} \end{cases}$$

Let us remind that $|A|$ denotes the area of object A. Now we can formulate and solve the CSP problem as the following binary programming problem (BCSP):

$$\sum_{j \in M} \left| \overline{convh}(H_j) \right| \cdot x_j \to \min, \tag{1}$$

$$\sum_{j \in M} x_j = q \tag{2}$$

$$\sum_{j \in M} a_{ij} x_j = 1 \qquad i \in N \tag{3}$$

$$\sum_{j \in M-\{i\}} b_{ij} x_j = 0 \qquad i \in M \tag{4}$$

$$x_j \in \{0, 1\} \qquad j \in M \tag{5}$$

The objective function (1) minimizes the total area of the convex hulls in the solution. The cardinality constraint (2) states that the partitioning contains exactly q clusters. The set partitioning constraints (3) ensure that every point of the set P lies in exactly one cluster. Constraint (4) tells us that every two convex hulls are disjunct.. The logical constrain (5) chooses a set of clusters in the solution $x_{jk} = 1$, $k = 1, 2, ..., q$. The optimal solution of the CSP problem has the following form

$$\mathcal{P}(q) = (P_1, P_2, ..., P_q),$$

$$P_k = \left\{ p \in \mathcal{P} : p \in \overline{convh}(H_{jk}), x_{jk} = 1 \right\}.$$

For the exact solution of the BCSP we have the known set H which has an exponential growth. Its size is too large for the solving of practical problems via accessible bivalent programming solvers.

One approach how to try to find an approximate solution is the reduction of the set H. The natural way how to do it is based on the capacities c_i of the points p_i. Let C be the capacity of vehicles. We can define the capacity of a cluster as the sum of capacities of its points. The cluster of the points P is then a *feasible cluster* if

$$\sum_{p_i \in P} c_i \leq C . \tag{6}$$

Now we can formulate the problem which helps to reduce the set H.

4. Convex location problem

When we approximate q convex hulls of the CSP problem as q-median of seeds of clusters we have the following optimization problem (CL):

Let us have the given set of n *points* $P = \{p_1, p_2, ..., p_n\}$ *in the plane, integer* q *where* $n \geq 6$ *and* $2 \leq q \leq \lceil n/3 \rceil$ *and positive integer numbers* $c_1, c_2, ..., c_n, C$. *The goal is to find a* q *seeds of feasible clusters of points -* $S(q) = \{s_1, s_2, ..., s_q\}$ *such every two cluster hulls of cluster have empty intersection and the sum of Euclidean distances of points to seeds of clusters is minimal.*

This model measures the cluster by a sum of distances from the seed of cluster to the points of cluster as we can see in Fig. 3. The *convexity constraint* assigns the point p to the cluster P_1 because the distance from the point s_1 to point p is shorter than the distance from the point s_2 to point p.

Let $D = (d_{ij})$ be the matrix of Euclidean distances between the points P let us denote the maximum distance by d_{max}. We can define the bivalent variables $y_i = 1$ if the point p_i is a seed of
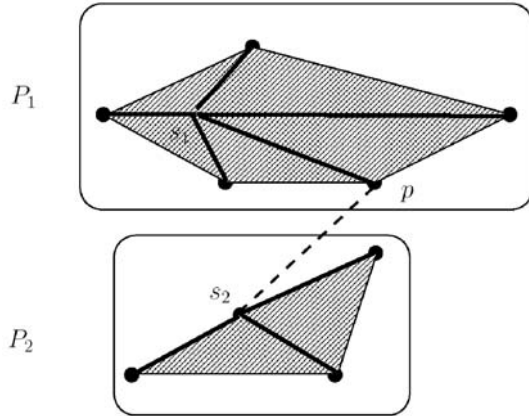
Fig. 3 Convexity constraint for clusters P_1 and P_2 with seeds s_1, s_2

the same cluster and $y_i = 0$ otherwise. We also define the bivalent variables $x_{ij} = 1$ if the point pj is assigned to the seed pi. It is easy to construct the set $S = [p_i \in P : y_i = 1]$.

Now we can formulate and solve the CL problem as the bivalent-programming problem (BCL):

$$\sum_{i \in N}\sum_{j \in N} d_{ij}x_{ij} \to \min, \tag{7}$$

$$\sum_{i \in N} y_i = q \tag{8}$$

$$\sum_{i \in N} x_{ij} = 1 \qquad\qquad j \in N, \tag{9}$$

$$\sum_{j \in N} c_j x_{ij} \leq C y_i \qquad\qquad i \in N, \tag{10}$$

$$d_{ij}x_{ij} = (d_{kj} - d_{\max})y_k \leq d_{\max} \qquad i, j, k \in N, i \neq j, j \neq k \tag{11}$$

$$x_{ij}, y_i \in \{0, 1\} \qquad\qquad i, j \in N. \tag{12}$$

The objective function (7) minimizes the total distance of the points to the nearest seed. The cardinality constraint (8) states that the solution has q clusters. The assignment constraint (9) ensures that every point of the set P is assigned exactly to one computed seed. The capacity constraint (10) is valid for feasible clusters only. The convexity constraint (11) ensures that the constructed convex hulls are really of shape of a convex polygon. Finding such polygon is similar to the same phase in Voronoi algorithm. The constaraint describes a situation when we need to decide to which cluster a point belongs. When there is more possibilities it is always assigned to a cluster with nearest seedThe last logical constraint (12) chooses seeds and defines clusters with its seed.

It can be proved that the BCL problem is NP-hard. Janacek and Gabrisova studied the possible approach to an approximate solution of the capacitated location problem based on lagrangean relaxation in [4]. Computational experiments with program *glpsol* for solution of mixed linear programming problems show that using Open Source library GLPK (GNU Linear Programming Kit) [7] we can find a good approximate solution of a real-world problem.

5. Instances of Florida area

We verified the models on the two instances of Florida area with $n = 234$ points and the number of clusters $q = 10$ and $q = 13$. The capacities of points are daily weights of customer's demands. For every q we solved 7 instances of BCSP problems with

$$C \in \left\{ C_k : k \in \{0, \dots, 6\}, C_0 > C_1 > \dots > C_6 \right\}$$

$$where\ C_0 = \left| \frac{\sum_{i=1}^{n} c_i}{q+2} \right|$$

and $C_k + 1$ is the maximum of cluster capacities in the solution of instance with the vehicle capacity C_{k-1} for $k = 1, \dots, 6$. The points on convex hulls define the reduced set $H_q = [H_1, H_2, \dots, H_{7 \cdot q}]$. By solving the BCSP problem, we got the clusters for ten vehicles as we can see in Fig. 4. The convex hulls were constructed by Graham's scan algorithm [3] implemented in *Python* [2], [6], [8]. An interesting property of the obtained solution is that the clusters have a relatively uniform distribution of the cluster capacity defined by the left side of inequality (6). Note that the total time of computing via [7] program *glpsol* was between 15 – 16 hours for one q.
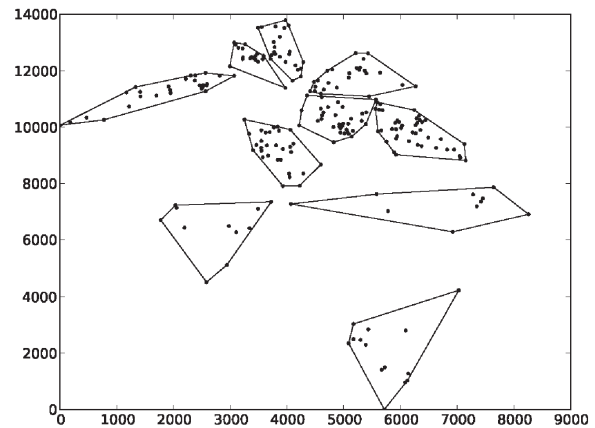

Fig. 4 Convex hulls for 10 clusters of Florida area

6. Open questions

The experiments with Florida instances show that real world BCSP problems with several hundreds of points can require long computational times. Reduction of the problem by solving some associated BCL problems is possible. Open questions are
- the approximate ratio for presented heuristic,
- the applicability of estimation of vehicle service with the aid of the area and weight of cluster,
- the possibility of weakening the formulation of the convexity constraint (11) in the BCL problem,

- the development of special methods for solving the BCSP and BCL problems.
- possibility of formulation and solution of BCSP via Voronoi diagrams

References

[1] BERG, M.: *Computational Geometry: Algorithms and Applications,* 3rd rev. ed., Springer-Verlag, 2008, ISBN 978-3-540-77973-5, 386 pp.

[2] DALE, D., DROETTBOOM, M., FIRING, E., HUNTER, J.: *Matplotlib*, 2008, 754 pp.

[3] GHERMAN, D.: *Recipe 66527: Finding the convex hull of a set of 2D points,* http://code.activestate.com/recipes/66527/

[4] JANACEK, J., GABRISOVA, L.: *Lagrangean Relaxation Based Approximate Approach to the Capacitated Location Problem.* In: Scientific Letters of the University of Zilina, Communications 3, 2006, ISSN 1335-4205, pp. 19–24

[5] JURICEK, M.: *Private communication,* Florida, USA

[6] KAUKIC, M.: *Basics of Programming in Pylab (in Slovak),* FEI TU Kosice, 2006, ISBN 80-8073-634-0, pp. 59

[7] MAKHORIN, A.: *GNU Linear Programming Kit,* Reference Manual, Version 4.29, Moscow, 2008, pp.154

[8] VAN ROSSUM, G., DRAKE, F.: *Python Tutorial,* Python Software Foundation, 2009, pp. 124