Dynamic programming

Lecture 3: Examples

Giovanni Righini

Doctoral course, 2023

UNIVERSITÀ DEGLI STUDI DI MILANO

Motivation

A company sells small appliances and must provide spare parts to several sale points.

The types of spare parts to be stocked and distributed have different prices.

To simplify administrative procedures, the company wants to group them into clusters, such that all spare parts in the same cluster are sold at the same price.

The price assigned to each cluster must be chosen so that the expected annual income remains equal to a predefined value, estimated on the basis of the expected demand for each type of spare part.

The goal is to keep the price of each cluster as close as possible to the original price of the spare parts in it.

The number of clusters is a user-defined parameter.

Weight-constrained clustering on a line: data

Data.

- a set N of n points on a the real line;
- a weight $w_i \in \Re_+$ for each point $i \in N$;
- a value $p_i \in \Re_+$ for each point $i \in N$ (its position on the line);
- a coefficient $\alpha \geq 1$.

Decision variables

Decisions.

- partition the points into K clusters $\{C_1, \ldots, C_K\}$, with $C_k \subset N \; \forall k = 1, \ldots, K$;
- assign a position q_k to the centroid of each cluster $C_k \; \forall k = 1, \ldots, m$.

Variables.

- A binary assignment variable x_{ik} represents the assignment of point $i \in N$ to cluster C_k.
- A continuous variable $q_k \in \Re_+$ indicates the position of the centroid of cluster C_k on the real line.

Constraints

- Assignment constraints:

$$\sum_{k=1}^{m} x_{ik} = 1 \quad \forall i \in N$$

- Total weight:

$$\sum_{k=1}^{K} \sum_{i=1}^{n} w_i q_k x_{ik} \geq \alpha \sum_{i=1}^{n} w_i p_i . \tag{1}$$

Additional restrictions.

- clusters cannot overlap;
- each centroid must lie within its cluster.

Objectives

The problem has two conflicting objectives.

1. Minimize the maximum difference between p_i and q_k for any point $i \in N$ in cluster C_k (*offset*, for brevity).
2. Minimize the number of clusters.

Objective 2 can be replaced by a constraint $x_{ik} = 0 \; \forall k > K$, so that the optimal solution of the problem is computed with respect to Objective 1 for different values of K in a used-defined range.

This allows enumerating all Pareto-optimal solutions of the two-objectives problem.

$$
\begin{aligned}
\text{minimize } & z \\
\text{s.t. } & z \geq q_k - \overline{p} + (\overline{p} - p_i)x_{ik} && \forall i \in N, \forall k = 1, \dots, K \\
& z \geq p_i x_{ik} - q_k && \forall i \in N, \forall k = 1, \dots, K
\end{aligned}
$$

where $\overline{p} = \max_{i \in N}\{p_i\}$.

Classification

- The model is mixed-01 programming, with binary assignment variables x and continuous variables q;
- The model is non-linear, owing to constraints

$$\sum_{k=1}^{K} \sum_{i=1}^{n} w_i q_k x_{ik} \geq \alpha \sum_{i=1}^{n} w_i p_i .$$

- Instances are expected to be large-sized (n: tens of thousands; m: hundredths).

The use of general-purpose MINLP solvers cannot be considered a viable option to find provably optimal solutions.

Additional constraints

Constraint 1: Non-overlapping clusters. After sorting the points such that $i < j$ implies $p_i \leq p_j$ and sorting the clusters so that $h < k$ implies $q_h \leq q_k$,

$$x_{ik} + x_{jh} \leq 1 \quad \forall i < j \in N, h < k \in K.$$

Constraint 2: Bounded centroids.

$$\min_i \{\overline{p} - (\overline{p} - p_i)x_{ik}\} \leq q_k \leq \max\{p_i x_{ik}\}.$$

In particular, when a cluster $k \in K$ includes only one point $i \in N$, then Constraint 2 imposes $q_k = p_i$.

Property 1

Property 1. *If Constraint 2 is not imposed, then there exists an optimal solution complying with Constraint 1.*

Proof. Consider a solution X in which $i \in N$ is assigned to $k \in K$, $j \in N$ is assigned to a $h \in K$ and $p_i < p_j$ and $q_k > q_h$. Let $z(X)$ be the maximum offset in X.
Re-assigning j to C_k produces a non-worse and feasible solution.
Since $q_k > q_h$, then $p_j - q_k < p_j - q_h$.
Since $p_i < p_j$, then $q_k - p_j < q_k - p_i$.
Therefore, $|p_j - q_k| < \max\{p_j - q_h, q_k - p_i\} \leq z(X)$.
Since $q_k > q_h$, the re-assignment of j yields a strictly positive profit increase. In the new solution the number of clusters is not larger and all points remain assigned. Therefore, the new solution is feasible and non-worse than the original one.
By repeated re-assignments, Constraint 1 can always be eventually enforced (Q.E.D.).

Property 1

When Constraint 2 is imposed, then Property 1 does not hold any more, in general.

Consider $n = 4$, $p = [100, 140, 150, 205]$, $w = [4, 2, 1, 1]$, $m = 2$. Total profit is 1185. Assume target profit to be 1220.

Solution A**:** $C_1 = \{1, 2\},\ C_2 = \{3, 4\}$ (complying with Constraint 1). $q_1 = 140, q_2 = 190$ with $z = 40$.

Solution B**:** $C_1 = \{1, 3\},\ C_2 = \{2, 4\}$ (violating Constraint 1). $q_1 = 137.5, q_2 = 177.5$ with $z = 37.5$.

Solution C**:** $C_1 = \{1\},\ C_2 = \{2, 3, 4\}$ (complying with Constraint 1). $q_1 = p_1 = 100$ (owing to Constraint 2), $q_2 = 205$, with $z = 65$.

Solution D**:** $C_1 = \{1, 2, 3\},\ C_2 = \{4\}$ (complying with Constraint 1). $q_1 = 145$, $q_2 = p_4 = 205$ (owing to Constraint 2), with $z = 45$.

Property 2

Property 2. *There exists an optimal solution in which* $q_k \geq \frac{p_k^+ + p_k^-}{2}$ $\forall k = 1, \ldots, K$, *where* $p_k^+ = \max_{i \in C_k}\{p_i\}$ *and* $p_k^- = \min_{i \in C_k}\{p_i\}$.

Proof. Consider a feasible solution with $q_k < \frac{p_k^+ + p_k^-}{2}$.

Then, the maximum offset of the points in C_k is given by $p_k^+ - q_k$.

Therefore q_k can be increased up to $\frac{p_k^+ + p_k^-}{2}$ yielding a positive increase in the profit and a decrease in the offset of the cluster.

In this way the objective function value does not increase and no constraint can be violated (Q.E.D.).

Corollary. In optimal solutions, the offset of each cluster C_k is given by $q_k - p_k^-$.

Three problems

Owing to Properties 1 and 2, there are three significant variations of the problem:

A) Constraint 2 not imposed: Constraint 1 is redundant.

B) Constraint 2 imposed and Constraint 1 imposed.

C) Constraint 2 imposed and Constraint 1 not imposed.

D.P. algorithms for Problem A and Problem B.

Problem C is open.

Problem A

Constraint 2 is not imposed.
Constraint 1 can be imposed wlog (wloo).

A D.P. algorithm: step 1

Step 1: define a sequence of decisions.

Points in N are renumbered by non-decreasing order of their position p_i.

Owing to Constraint 1, the partition of a subsequence is uniquely defined by the rightmost points of the clusters.

Notation: $W_i^j = \sum_{\ell=i}^{j} w_\ell$.

A D.P. algorithm: step 2

Step 2: define the state.

A label is associated with each pair (i, k), where

- $i \in N$ is a point;
- k is the number of clusters used to partition $\{1, \ldots, i\}$, with $k \leq i$.

The label has the form $(z(i, k), v(i, k))$:

- $z(i, k)$: minimum offset that can be achieved when $\{1, \ldots, i\}$ is partitioned into k clusters;
- $v(i, k)$: maximum profit that can be collected from $\{1, \ldots, i\}$ partitioned into k clusters, when their offsets do not exceed $z(i, k)$.

A D.P. algorithm: step 3

Step 3: define the extension function.

Initialization.

$$z(i,1) = \frac{p_i - p_1}{2} \qquad v(i,1) = \frac{p_i + p_1}{2} W_1^i \quad \forall i \in N.$$

Extension. To guarantee an implicit enumeration of all possible sets of disjoint clusters, all possible ways to partition $\{1, \ldots, i\}$ are considered, such that $\{1, \ldots, j\}$ is partitioned in $k-1$ clusters and $\{j+1, \ldots, i\}$ is assigned to one cluster, for any possible choice of $j < i \in N$.

Extension rule from $(j, k-1)$ to (i, k) for offset:

$$z(i,k)^{(j)} = \max\left\{ z(j, k-1), \frac{p_i - p_{j+1}}{2} \right\}.$$

A D.P. algorithm: step 3

$$v(i,k)^{(j)} = v(j,k-1) + W^i_{j+1}\frac{p_i + p_{j+1}}{2} +$$
$$+\max\left\{\left(z(j,k-1) - \frac{p_i - p_{j+1}}{2}\right) W^i_{j+1}, \left(\frac{p_i - p_{j+1}}{2} - z(j,k-1)\right) W^j_1\right\}.$$

The profit $v(j,k-1)$ is the contribution of clusters C_1 to C_{k-1} when their offset is bounded by $z(j,k-1)^{(j)}$.

The contribution of C_k is $W^i_{j+1}q_k$, where q_k is initially set to $\frac{p_i+p_{j+1}}{2}$ by Property 2, when the cluster offset is bounded by $\frac{p_i-p_{j+1}}{2}$.

If $z(j,k-1) > \frac{p_i-p_{j+1}}{2}$, then q_k can be increased by the difference.
If $\frac{p_i-p_{j+1}}{2} > z(j,k-1)$, then all values q_1 to q_{k-1} can be increased by the difference.

Dominance

Consider two labels $(z^A(i,k), v^A(i,k)) = (z(i,k)^{(j')}, v(i,k)^{(j')})$ and $(z^B(i,k), v^B(i,k)) = (z(i,k)^{(j'')}, v(i,k)^{(j'')})$, corresponding to the same state (i,k) reached by different precedecessors.

If $z^A(i,k) \leq z^B(i,k)$ and $v^A(i,k) \geq v^B(i,k)$ and at least one of the inequalities is strict, then $(z^A(i,k), v^A(i,k))$ dominates $(z^B(i,k), v^B(i,k))$.

This condition can be generalized:
if $z^A(i,k) \leq z^B(i,k)$ and $z^B(i,k) - z^A(i,k) \geq (v^B(i,k) - v^A(i,k))W_1^i$ and at least one of the inequalities is strict, then $(z^A(i,k), v^A(i,k))$ dominates $(z^B(i,k), v^B(i,k))$.

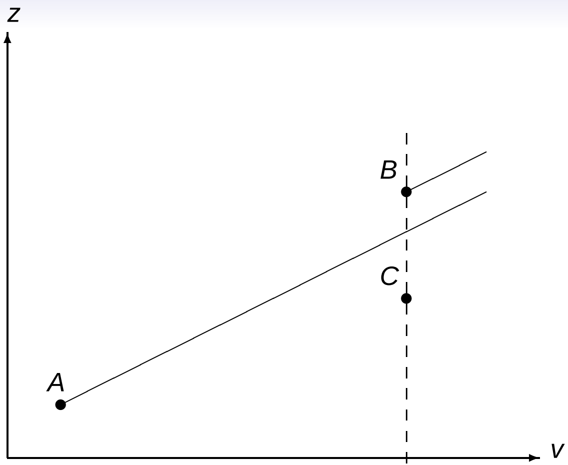

Figura: Comparison between labels: A dominates B, but not C.

If the centroids of $C_1^A, \ldots, C_k^A$ are moved to increase the profit v^A to match v^B, then z^A remains below z^B.

Termination

The optimal solution with K clusters is given by a label of state (n, K).

Given a label $(z(n,K), v(n,K))$, if $v(n,K) \geq \alpha \sum_{i \in N} w_i p_i$, then $z(n,K)$ is the final offset value.

Otherwise, $z(n,K)$ must be increased by $\dfrac{\alpha \sum_{i \in N} w_i p_i - v(n,K)}{W_1^n}$ to satisfy the profit constraint.

After possibly correcting the labels to make them feasible, the one with minimum offset is optimal.

The optimal labels of states (n, K) for all K provide a complete description of the Pareto-optimal frontier of the two-objectives problem, allowing the decision-maker tuning the trade-off between the number of clusters and the minimum offset.

Computational complexity

Each state (i, k) can have $O(i^k)$ non-dominated associated labels.

The worst-case computational complexity is exponential.

It is polynomial for any K fixed.

In practice, it is not likely that many non-dominated labels exist for each state.

Problem B

Constraint 2 is imposed.
Constraint 1 is imposed.

The model

The mathematical model is the same as for Problem A with the additional restrictions

$$p_k^- \leq q_k \leq p_k^+ \quad \forall k \in K$$

and

$$x_{ik} + x_{jh} \leq 1 \; \forall i < j \in N, \forall h < k \in K.$$

A D.P. algorithm: step 1

Step 1: define a sequence of decisions.

Same as for Problem A.

Points on the real line are sorted in ascending order of their positions.

Clusters are sorted in ascending order of their centroids.

Since overlaps are not allowed, the ordering is well-defined.

A D.P. algorithm: step 2

Step 2: define the state.

Each policy that partitions $\{1, \ldots, i\}$ in k clusters is characterized by two values, $z(i, k)$ and $v(i, k)$ with the same meaning as before.

However, they are no longer enough to establish dominance.

Constraint 2 may produce different effects on different policies when the offset is increased to meet the profit requirement.

When $q_k = p_k^+$ (C_k is *saturated*), no further increase in profit can be obtained from C_k.

A D.P. algorithm: step 2

Consider a generic policy leading to a state (i, k); it corresponds to a set of clusters, indexed by $h = 1, \ldots, k$.

The correspondence between an increase in the allowed offset and the corresponding increase in the achievable profit is no longer a linear function as in Problem A, this is now a piecewise linear function.

Given a label $(z(i, k), v(i, k))$, for each cluster $h = 1, \ldots, k$ we define

- its total weight $W_h = \sum_{\ell \in C_h} w_\ell$
- its residual allowed offset increase $r_h = \max\{p_h^+ - (p_h^- + z(i, k)), 0\}$.

Notation:

- Δ: a generic offset increase of the policy;
- $\Psi(i, k, \Delta) = \sum_{h=1}^{k} (W_h \min\{r_h, \Delta\})$ is the corresponding profit increase.

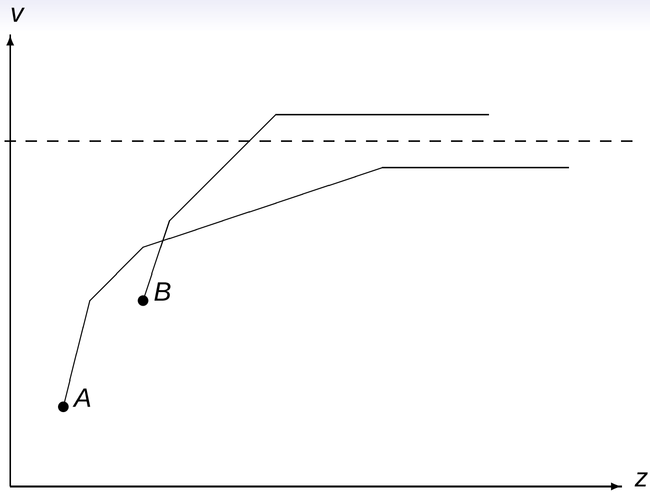

Figura: There is no dominance betweeen the piecewise linear functions of policies *A* and *B*.

A D.P. algorithm: step 3

Initialization.

$$z(i,1) = \frac{p_i - p_1}{2} \quad v(i,1) = \frac{p_i + p_1}{2} \sum_{j=1}^{i} w_j \quad \forall i \in N.$$

Extension.
Extension of the offset:

$$z(i,k)^{(j)} = \max\left\{ z(j,k-1), \frac{p_i - p_{j+1}}{2} \right\},$$

when a state (i,k) is reached from a state $(j,k-1)$.

A D.P. algorithm: step 3

Extension of the profit:

$$v(i,k)_j = v(j,k-1) + W^i_{j+1}\frac{p_i + p_{j+1}}{2} + \max\{\Psi^+, \Psi^-\}.$$

When $\frac{p_i - p_{j+1}}{2} > z(j,k-1)$, then the offset of C_1 to C_{k-1} can be increased by $\Delta^+ = \frac{p_i - p_{j+1}}{2} - z(j,k-1)$. The corresponding profit increase is $\Psi^+ = \sum_{h=1}^{k-1}(W_h \min\{r_h, \Delta^+\})$.

When $z(j,k-1) > \frac{p_i - p_{j+1}}{2}$, then the offset of C_k can be increased by $\Delta^- = \min\{z(j,k-1) - \frac{p_i - p_{j+1}}{2}, \frac{p_i - p_{j+1}}{2}\}$. The corresponding profit increase is $\Psi^- = \Delta^- W^i_{j+1}$.

Termination

A final solution with K clusters is provided by each policy reaching state (n, K).

If $v(n, K) \geq \alpha \sum_{i \in N} w_i p_i$, then $z(n, K)$ is the final value.

Otherwise, $z(n, K)$ must be increased so that the profit increases by $\overline{\Psi} = \alpha \sum_{i \in N} w_i p_i - v(n, K)$.

The necessary offset increase $\overline{\Delta}$ corresponding to a profit increase of $\overline{\Psi}$ is determined from the piecewise linear function $\Psi(n, K, \Delta)$.

The policy providing the feasible solution with the minimum final value of $z(n, K)$ is the optimal one.

Dominance

Dominance between labels occurs only when one of the two piecewise linear functions is completely above the other.

To limit the combinatorial explosion of the number of feasible policies to be recorded, for each pair (i, k) the algorithm records a unique non-dominated piecewise linear function, defined as the maximum of $v(i, k) + \Psi(i, k, \Delta)$ for each value of $z(i, k) + \Delta$.

In general this function corresponds to different policies in different intervals of Δ and it is neither convex nor concave.

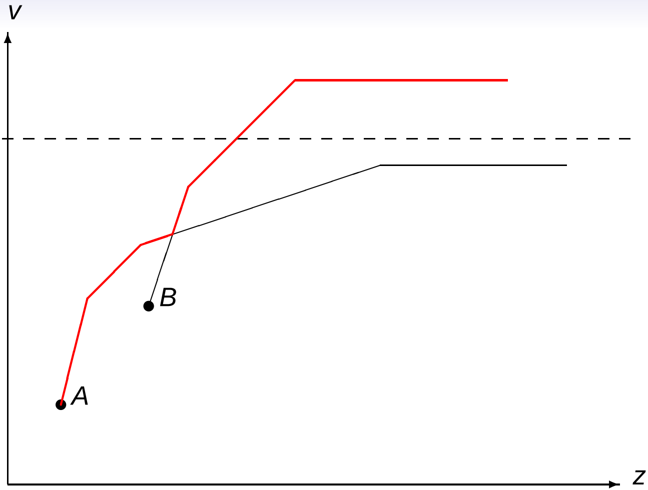

Figura: Dominance between piecewise linear functions.

Complexity

Given two piecewise linear functions, made of s_1 and s_2 segments respectively, the non-dominated piecewise linear function resulting from their comparison can be made by up to $s_1 + s_2$ segments.

The number of segments associated with a state (i, k) grows exponentially with k in the worst-case.

The D.P. algorithm for Problem B has exponential worst-case time complexity.