On Objective Function Representation Methods in Global Optimization

Panos M. Pardalos

Center for Applied Optimization, Department of Industrial and Systems Engineering, University of Florida

Gainesville, FL USA

and

LATNA, National Research University Higher School of Economics

http://www.ise.ufl.edu/pardalos

This talk is dedicated to the memory of my friend and colleague Chris Floudas

China, 2013

University of Florida, 2015

Decomposition Techniques

General

- Decomposition techniques for solving optimization problems have been used as early as in the 1960's for linear mixed integer and convex optimization

- Well known techniques include:
  - Dantzig-Wolfe
  - Benders

- The choice of the decomposition (of objective function) influences the choice of the algorithm used to solve the corresponding optimization problem

Decomposition Techniques

Overview

- Separable optimization:

$$\min_{x \in \mathbb{R}^n} F_0(x)$$
$$\text{s.t. } F_i(x) \le b_i \,, i = 1, \dots, m$$
$$l_i \le x_i \le u_i \,, i = 1, \dots, m$$

where each $F_i(x) = \sum_{j=1}^{n} F_{ij}(x_j), i = 0,1, \dots, m$

- Factorable optimization:

  See book:

  Garth, McCormick, "Nonlinear Programming: Algorithms and Applications", 1983

Decomposition Techniques

Overview

- Almost Block Separable Optimization:

$$\min_{x \in \mathbb{R}^n} f(x) = f_1(u, y) + f_2(v, y)$$

where $x = (u, v, y) \in \mathbb{R}^n$

and $u \in \mathbb{R}^{n_1}, v \in \mathbb{R}^{n_2}, y \in \mathbb{R}^{n_3}$, $n_1 + n_2 + n_3 = n$

y are called complicated variables [usually $n_1, n_2 \gg n_3$]

- Let $\varphi_1(y) = \min_u f_1(u, y)$, $\varphi_2(y) = \min_v f_2(v, y)$

Then the above problem is equivalent to:

$$\min_y \varphi_1(y) + \varphi_2(y)$$

Note that if f_1, f_2 are convex, then $\varphi_1(y)$ and $\varphi_2(y)$ are convex

DC Optimization Problems

Overview

- Many powerful techniques in global optimization are based on the fact that many objective functions can be expressed as the difference of two convex functions (so called d.c. functions)
- If $D(x)$ is an objective function in $\mathbb{R}^n$, then the representation $D(x) = p(x) - q(x)$, where p, q are convex function is said to be a d.c. decomposition of D
- The space of d.c. functions is closed under many operations frequently encountered in optimization (*i.e.*, sum, product, max, min, etc)
- Hartman 1959: Every locally d.c. function is d.c.
- For simplicity of notation, consider the d.c. program:

$$\min f(x) - g(x)$$
$$\text{s.t. } x \in D$$

where D is a polytope in $\mathbb{R}^n$ with nonempty interior and f and g are *convex functions* on $\mathbb{R}^n$

DC Optimization Problems

Overview

- By introducing an additional variable t, the above problem can be converted into the equivalent problem of **Global Concave Minimization**:

$$\min t - g(x)$$
$$\text{s.t. } x \in D, f(x) - t \leq 0$$

with concave objective function $t - g(x)$ and convex feasible set $\{(x,t) \in \mathbb{R}^{n+1}: x \in D, f(x) - t \leq 0\}$.

- If (x^*, t^*) is an optimal solution of the above program then x^* is an optimal solution of the initial d.c. program and $t^* = f(x^*)$

- Therefore, any D.C. program can be solved by an algorithm for minimizing a concave function over a convex set.

UF | UNIVERSITY of FLORIDA

Continuous DC programming

- DC function:

  A real-valued function $f: \mathbb{R}^n \longrightarrow \mathbb{R} \cup \{+\infty, -\infty\}$ s.t.

$$f(x) = g(x) - h(x), \forall x \in \mathbb{R}^n$$

  where $g: \mathbb{R}^n \longrightarrow \mathbb{R} \cup \{+\infty\}$ and $h: \mathbb{R}^n \longrightarrow \mathbb{R} \cup \{+\infty\}$ are convex functions

- DC program:

  A program of the form:

$$\begin{aligned} &\min f_0(x) \\ &\text{s.t. } f_i(x) \leq 0\ , i = 1,2, \dots, n \end{aligned}$$

  where $f_i(x)$ are convex functions $(i = 0,1,2, \dots, n)$

  - It is equivalent to the unconstrained DC program:

$$\inf_{x \in \mathbb{R}^n} f(x) = g(x) - h(x)$$

UF UNIVERSITY of FLORIDA

Continuous DC programming

- Subgradients:

  A vector x^* is a subgradient of a convex function h at a point x if:

$$h(z) \geq h(x) + \langle x^*, z - x \rangle$$

  - The subdifferential of $h(x)$ is the set of all subgradients

- Conjugate functions:

  A conjugate function $h^*: \mathbb{R}^n \longrightarrow \mathbb{R} \cup \{+\infty\}$ of a convex function $h: \mathbb{R}^n \longrightarrow \mathbb{R} \cup \{+\infty\}$ is :

$$h^*(p) \coloneqq \sup_{y \in \mathbb{R}^n} \{\langle y, x \rangle - h(x)\}$$

  - The conjugate function $h^*(y)$ of a function $h(x)$ is convex
  - If $h(x)$ is a closed proper convex function, then $h^{**} = h$
  - Given closed convex functions $g, h: \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$, it is:

$$\inf_{x \in \mathbb{R}^n} \{g(x) - h(x)\} = \inf_{p \in \mathbb{R}^n} \{h^*(p) - g^*(p)\}$$

Continuous DC programming

- DC Algorithm:
  - *Step 0 Find an inintial solution* $x^0 \in dom_R(g)$. *Set* $t \coloneqq 0$.
  - *Step 1 Find* $p^{(t)} \in \partial_R h(x^{(t)})$.
  - *Step 2 Find* $x^{(t+1)} \in \partial_R g^*(p^{(t)})$, *where* g^* *is the conjugate of* g.
  - *Step 3 If* $f\left(x^{(t+1)}\right) = f(x^{(t)})$, *stop. Otherwise, set* $t \coloneqq t+1$, *go to Step 1.*

where $x^{(t+1)} = \arg\min_{y \in \mathbb{R}^n} \{g(y) - h(x^t) - \langle p, y - x^{(t)} \rangle\}$

Continuous relaxations for discrete DC programming

- The positive support of $x \in \mathbb{Z}^n$ is:

$$\mathrm{supp}^+(x) \coloneqq \{i \in \{1,2,\dots,n\}: x_i > 0\}$$

- The indicator vector χ_S is defined by:

$$\chi_S(i) = \begin{cases} 1, i \in S \\ 0, i \notin S \end{cases}$$

- There are two common discrete functions:
- $M^\natural$-convex functions:
  - For all $x, y \in \mathbb{Z}^n$ and $i \in supp^+(x - y)$, function $h: \mathbb{Z}^n \to \mathbb{Z} \cup \{+\infty\}$ is $M^\natural$-convex if it satisfies:

$$h(x) + h(y) \geq \min\{h(x - \chi_i) + h(x + \chi_i)\},$$
$$\min_{j \in supp^+(x-y)} \{h(x - \chi_i + \chi_i) + h(y + \chi_i - \chi_j)\}$$

- $L^\natural$-convex functions:
  - For all $x, y \in \mathbb{Z}^n$, function $h: \mathbb{Z}^n \to \mathbb{Z} \cup \{+\infty\}$ is $L^\natural$-convex if it satisfies:

$$h(x) + h(y) \geq h\left(\left\lceil \frac{x+y}{2} \right\rceil\right) + h\left(\left\lfloor \frac{x+y}{2} \right\rfloor\right)$$

UF UNIVERSITY of FLORIDA

Continuous relaxations for discrete DC programming

- Given two functions $g, h: \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$,
  - The effective domain of g is $dom_Z g \coloneqq \{x \in \mathbb{Z}^n : g(x) < +\infty\}$
  - The convex closure $\bar{g}(x): \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ of g is:

  $\bar{g}(x) = \sup\{s(x): s \text{ is an affine function}, s(y) \le g(y)(y \in \mathbb{Z}^n)\}$
  - A convex extension $\hat{g}: \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}$ of g is a convex function with the same function value on $x \in \mathbb{Z}^n$
  - Let $\tilde{f}(x) \coloneqq \bar{g}(x) - \hat{h}(x)$. Then $\tilde{f}(x) \coloneqq g(x) - h(x), \forall x \in \mathbb{Z}^n$. Thus:

$$\inf_{x \in \mathbb{Z}^n} g(x) - h(x) = \inf_{x \in \mathbb{Z}^n} \tilde{f}(x) \ge \inf_{x \in \mathbb{R}^n} \tilde{f}(x)$$

- For convex extensible functions $g, h: \mathbb{Z}^n \to \mathbb{R} \cup \{+\infty\}$ with $dom_Z g$ bounded and $dom_Z g \subseteq dom_Z h$, it is:

$$\inf_{z \in \mathbb{Z}^n} \{g(z) - h(z)\} = \inf_{x \in \mathbb{R}^n} \{\bar{g}(x) - \hat{h}(x)\}$$

where $\bar{g}(x)$ is the linear closure of $g(x)$ and $\hat{h}(x)$ is any convex extension of $h(x)$

DI Optimization Problems

- Monotonicity with respect to some variables (partial monotonicity) or to all variables (total monotonicity) is a natural property exhibited by many problems encountered in applications. The most general problem of d.i. monotonic optimization is:

$$\min f(x) - g(x)$$
$$\text{s.t. } f_i(x) - g_i(x) \leq 0\,, i = 1, \ldots, m$$

  where all functions are increasing in $\mathbb{R}_+^n$

- Assume without loss of generality that $g(x) = 0$

- $\{\forall i\ f_i(x) - g_i(x) \leq 0\} \Leftrightarrow \max_{1 \leq i \leq m} \{f_i(x) - g_i(x)\} \leq 0 \Leftrightarrow F(x) - G(x) \leq 0$, where:

$$F(x) = \max_i \{f_i(x) + \sum_{i \neq j} g_j(x)\}\ , G(x) = \sum_i g_i(x)$$

- $F(x)$ and $G(x)$ are both increasing functions

DI Optimization Problems

- Problem reduces to:

$$\begin{aligned} &\min f(x) \\ \text{s.t.}\quad & F(x) + t \leq F(b), \\ & G(x) + t \geq F(b), \\ & 0 \leq t \leq F(b) - F(0), \\ & x \in [0, b] \subset \mathbb{R}_+^n \end{aligned}$$

- A set $G \subseteq \mathbb{R}_+^n$ is normal if for any two points x, x' such that $x' \leq x$, if $x \in G$, then $x' \in G$

- Numerous global optimization problems can be reformulated as monotonic optimization problems. Such problems include multiplicative programming, nonconvex quadratic programming, polynomial programming and Lipschitz optimization problems

Decomposition and Multi-objective Optimization

Overview

- Consider the following problems:

$$\min_{x \in D \subseteq R^n} F(x) = f_1(x) + \cdots + f_m(x) \qquad [1]$$

$$\min_{x \in D \subseteq R^n} f(x) = (f_1(x), \dots, f_m(x)) \qquad [2]$$

  [2] is a multi-objective optimization problem

  Let $E(f, D) \subseteq D$ be the set of all Pareto optimal solutions in D

- <u>Theorem</u>: If $\bar{x}$ is an optimal solution of problem [1], then $\bar{x} \in E(f, D)$ of problem [2]

- <u>Theorem</u>: Let $h_i(t)$ be monotonic increasing functions for $i = 1, \dots, m$. Consider the multi-objective optimization problem

$$\min_{x \in D \subseteq R^n} h(x) = (h_1(f_1(x)), \dots, h_m(f_m(x))) \qquad [3]$$

  Then $E(f, D) = E(h, D)$.

Kolmogorov's Superposition

Overview

- Theorem: For any integer $n \geq 2$ there are continuous real functions $\psi^{p,q}(x)$ on the closed unit interval $E^1 = [0,1]$ such that each continuous real functions $f(x_1, \dots, x_n)$ on the n-dimensional unit cube E^n is representable as

$$f(x_1, \dots, x_n) = \sum_{q=1}^{2n+1} \chi_q\Big[\sum_{p=1}^{n} \psi^{pq}(x_p)\Big]$$

where $\chi_q(y)$ are continuous real functions

- For $n = 3$, by setting:

$$\varphi_q(x_1, x_2) = \psi^{1q}(x_1) + \psi^{2q}(x_2)$$
$$h_q(y, x_3) = \chi_q[y + \psi^{3q}(x_3)]$$

we obtain from the above:

$$f(x_1, x_2, x_3) = \sum_{q=1}^{7} h_q[\varphi_q(x_1, x_2), x_3]$$

MAX-DFSS Problem

- An undirected connected graph $G = (V, E)$ with at least $k + 1$ vertices is k-vertex-connected if it remains connected whenever fewer than k vertices are removed
- The problem of finding a minimum weight k-vertex-connected subgraph is NP-hard when $k \geq 2$
- The minimal k-connected spanning subgraph problem is considered instead
- Boros *et al* considered how to generate all minimal k-vertex-connected spanning subgraphs in incremental polynomial time
- We consider a routing system characterized by a minimal 2-vertex-connected spanning subgraph with some high-degree vertices.
  - We name it the degree-concentrated maximum fault-tolerant spanning subgraph problem (MAX-DFSS)
  - Finding a maximum weight of minimal 2-vertex-connected spanning subgraph is also hard to solve

MAX-DFSS Problem

Formulation

- Given an undirected graph $G = (V, E)$, we need to find a minimal 2-connected spanning subgraph. The aim is to maximize the sum of the squares of the degree of each vertex:

$$\max \sum_{v \in V} \deg_S(v)^2$$

s.t. S : minimal 2-connected spanning subgraph

- This problem is the fault tolerant version of the problem of fault tolerant version for the degree-concentrated spanning tree problem (DST) of (Maehara et al, 2015)

- It can be applied in spanning tree routing system (Tanenbaum, 2010).

- In (Maehara et al, 2015), the monitoring of network communications is considered

MAX-DFSS Problem

Algorithms

- Greedy algorithm for the MAX-DFSS problem:

  - *Input.* 2-connected graph $G = (V, E)$
  - *Output.* A minimal 2-connected spanning subgraph $S \subseteq G$
  - *Step 1* $S \leftarrow G$
  - *Step 2* Repeat

    Step 2.1 If S is minimal 2-connected, go to step 3.

    Step 2.2 Select an edge $e^* \in$

$$\underset{\substack{e \in E(S) \\ S-e \text{ is } 2-\text{connected}}}{\arg\max} \sum_{v \in V(S-e)} \deg_{S-e}(v)^2$$

    Step 2.3 $S \leftarrow S - e^*$

  - *Step 3* Output S

MAX-DFSS Problem

DC Formulation

- Now we formulate the problem as a DC program.

  Let $M \in \mathbb{R}^{|V|\times|E|}$ be the incidence matrix for graph G.
- Let $x_e \in \{0,1\}, \forall e \in E$ indicate whether edge e is chosen into the subgraph or not.

  For any $x \in \{0,1\}^{|E|}$ we obtain a subgraph $S = \{e \in E: x_e = 1\}$

  $$Mx = (\deg_S(v_1), \deg_S(v_2), \ldots, \deg_S(v_n))^T$$
- The objective function $\sum_{v\in V} \deg_S(v)^2$ can be formulated as $x^T Ax$, where $A = M^T M \in \mathbb{R}^{|E|\times|E|}$. A is positive semi-definite.
- The problem becomes:

  $$\max \ x^T Ax$$

  s.t. $\{e \in E: x_e = 1\}$is a minimal 2-connected spanning subgraph
- If $g(x) = \begin{cases} 0, \{e \in E: x_e = 1\}\text{is a minimal 2−connected spanning subgraph} \\ +\infty, otherwise \end{cases}$

  the problem is equivalent to $\min_{x\in\{0,1\}^n} g(x) - x^T Ax$

MAX-DFSS Problem

DC Algorithm

- DC algorithm for the MAX-DFSS problem:

  - *Step 0 Choose* $x^{(0)} \in dom_R(g)$ *to be an initial solution. Set* $t \coloneqq 0$
  - *Step 1 Compute* $p^{(t)} \in \partial h(x^{(t)})$
  - *Step 2 Compute* $x^{(t+1)} \in \partial g^*(p^{(t)})$*, i.e.,,*
  $$x^{(t+1)} = \arg\min_y \{g(y) - \langle p^{(t)}, y\rangle\}$$
  - *Step 3 If* $f\left(x^{(t+1)}\right) = f\left(x^{(t)}\right)$*, go to Step 4. Otherwise, set* $t \coloneqq t + 1$*, go to Step 1*
  - *Step 4 Call Greedy algorithm for the MAX-DFSS on the graph represented by* $x^{(t)}$.

MAX-DFSS Problem

Complexity Evaluation

- $x^{(t+1)}$ can be obtained by solving a maximum weighted minimal 2-connected spanning subgraph with edge weight $p_e^{(t)}$ for $e \in E$

- Instead, we can find a 2-connected subgraph with at most $2|V| - 2$ edges and the total weight is at least 0.5 times the maximum total weight of minimal 2-connected spanning subgraph.

- Lemma: Suppose S^* is a maximum weight minimal 2-connected spanning subgraph of G and T^* is a maximum weight spanning tree of S^*. Then $p(T^*) \geq p(S^*)/2$.

- Therefore, the complexity of the following algorithm is $O(|V|^3)$

MAX-DFSS Problem

Algorithms

- Algorithm for the MAX-DFSS problem:

  - *Input* 2-connected graph $G = (V, E, p)$ *where* p *is the edge weight vector*
  - *Output* 2-connected spanning subgraph $S \subseteq G$ *with at most* $2|V|$ *edges*
  - *Step 1 Find out a maximum spanning tree* T *of* G *by Prim algorithm*
  - *Step 2* $S \leftarrow T$
  - *Step 3 Repeat*

    Step 3.1 Perform block decomposition for S. *If it has only one block, go to step 4.*

    Step 3.2 Search the maximal weighted edge $e \in E \backslash S$ *striding over different blocks*

    Step 3.3 $S \leftarrow S + e$
  - *Step 4 Output* S

MAX-DFSS Problem

Computational results

- Computational results can be found in:

Solving the degree-concentrated fault-tolerant spanning subgraph problem by DC programming Chenchen Wu, Yishui Wang, Panos M. Pardalos, Dachuan Xu, Zhao Zhang, Ding-Zhu Du (submitted, 2017)