Kétfázisú robusztus optimalizálás

Marosvári Ágnes

2020. december. 8.

A való életbeli optimalizálási problémák adatai gyakran tartalmazhatnak bizonytalanságot. Ez a bizonytalanság számos különböző okból eredhet, ez lehet mérési hiba, vagy egyszerűen az, hogy azt az adatot nem is lehet pontosan felvenni (például egy hozzzárendelési feladat esetén nem feltétlenül lehet előre tudni, hogy egyes termékekre mekkora lesz a kereslet), de az is előfordulhat, hogy a megoldás megengedettsége függ a jövőtől is (ismét a hozzárendelési feladatot példaként használva, egy természeti katasztrófa megbéníthat egy kiszolgáló pontot). A bizonytalanság kezelésére kétféle megközelítés létezik: a sztochasztikus és a robusztus optimalizálás. Az előbbi módszer a valószínűségszámításon alapul, míg az utóbbinál feltesszük, hogy a bizonytalan esetek egy előre ismert halmazból kerülnek ki, és olyan megoldást keresünk, amely megoldás marad, bármely bizonytalan eset is következzen be, és ezek közül a legjobbat szeretnénk. Ennek viszont olyan hátránya van, hogy az a megoldás, amely a legrosszabb esetre is megengedett, túlságosan konzervatív lehet. Többféle módon is kezelhető ez a probléma, például az [2]-ben szereplő modellben a bizonytalan adatok száma van korlátozva, és ennek a paraméternek a változtatásával lehet a modell konzervatívságát szabályozni. Ebben a félévben egy másik megközelítéssel foglalkoztunk a [3] cikk alapján.

1. A feladat

[3]-ban egy kétfázisú modell szerepel, ahol először egy y döntést hozunk, majd ezután derül ki a bizonytalan vektor, u értéke. Az y és az u vektor függvényében az erre az esetre adott válasz x lesz, összességében pedig a $cy + bx$ értéket szeretnénk minimalizálni, valamilyen c és b költésgfüggvények mellett (ezekről ezentúl feltesszük, hogy $b \geq 0$ és $c \geq 0$). A robusztusság érdekében a legrosszabb esetet nézzük, ezért szerepel a költségfüggvényben az U-ra vett maximum.

$$\min_{y \in S_y} cy + \max_{u \in U} \min_{x \in F(y,u)} bx$$

$$Ay \geq d$$

$$F(y,u) = \{x \in S_x : Gx \geq h - Ey - Mu\}$$

$$S_y \subseteq \mathbb{R}^n_+, \; S_x \subseteq \mathbb{R}^m_+$$

Egy ilyen probléma optimális megoldásának megkereséséről belátható, hogy NP-nehéz. A számítási nehézségek leküzdésére több megoldás is létezik, ezek közül a [3] cikkben szereplő algoritmus következik, amely az optimalizálás folyamán vágásokat (a primál oldalon) és oszlopokat is generál (column-and-constraint generation method - C&CG).

2. A C&CG algoritmus

Az algoritmus alapötlete a következő: először tegyük fel, hogy U egy véges halmaz, elemei $u_1, ..., u_r$, és az ezekhez tartozó változók a második fázisban $x_1, ..., x_r$. Ekkor a feladatunk az alábbi alakot ölti:

$$\min_y cy + \eta$$

$$Ay \geq d$$

$$\eta \geq bx_l \ \ \forall l = 1, ..., r$$

$$Ey + Gx_l \geq h - Mu_l \ \ \forall l = 1, ..., r$$

$$y \in S_y, \ x_l \in S_x \ \forall l = 1, ..., r, \ \eta \in \mathbb{R}_+, \ u_l \in U$$

Ha U számossága nem túl nagy, akkor ezt az LP-t könnyen megoldhatjuk. Ha viszont nem így van, például $U = \{u : Lu \leq v\}$ egy poliéder, akkor a benne lévő vektorok számba vétele, és így az összes feltétel felsorolása a gyakorlatban nem megoldható. Azonban ha csak a feltételek egy részét soroljuk fel, akkor az eredeti feladatunk egy relaxációját, és így optimumának egy alsó korlátját kapjuk. A cél tehát az, hogy megtaláljuk a szignifikáns $u \in U$ vektorokat, és így egyre erősebb alsó korlátokat kapjunk az optimumra. Az algoritmus lépései a következők.

1. Legyen az alsó korlát $LB = -\infty$, a felső korlát $UB = +\infty$, $k = 0$ és $O = \emptyset$.

2. Oldjuk meg a mester problémát (MP):

$$\min_{y,\eta} cy + \eta$$

$$Ay \geq d$$

$$\eta \geq bx_l \ \ \forall l \in O$$

$$Ey + Gx_l \geq h - Mu_l^* \ \ \forall l \leq k$$

$$y \in S_y, \ x_l \in S_x \ \forall l \leq k, \ \eta \in \mathbb{R}_+, \ u_l^* \in U$$

   Legyen MP optimális megoldása $(y_{k+1}^*, \eta_{k+1}^*, x_1^*, ..., x_k^*)$, és $LB = cy_{k+1}^* + \eta_{k+1}^*$. Ez alsó korlát, hiszen a minimalizálási feladatunk feltételeinek csak egy részhalmazát véve optimalizáltunk.

3. Definiáljuk a részproblémát (SP-t) a következő módon.

$$\mathcal{Q}(y) = \{\max_{u \in U} \min_{x} bx : Gx \geq h - Ey - Mu, x \in S_x\}$$

Ez tehát egy konkrét y-ra a célfüggvény második tagjának optimális értékét adja meg, így készíthetünk belőle felső korlátot. Oldjuk meg SP-t az y^*_{k+1} vektort használva, és legyen UB az eddigi felső korlát és $cy^*_{k+1} + \mathcal{Q}(y^*_{k+1})$ közül a kisebbik.

4. Ha $UB - LB \leq \epsilon$, akkor álljunk meg. Különben:

   (a) Ha $\mathcal{Q}(y^*_{k+1}) < +\infty$, vegyünk fel egy új x_{k+1} változót, és adjuk a következó feltételeket az MP-hez:

   $$\eta \geq bx_{k+1}$$

   $$Ey + Gx_{k+1} \geq h - Mu^*_{k+1}$$

   Itt u^*_{k+1} az az U-beli vektor, melyre $\mathcal{Q}(y^*_{k+1})$ optimális. Legyen $k = k + 1$, és $O = O \cup \{k + 1\}$. Menjünk a 2. lépéshez.

   (b) Ha $\mathcal{Q}(y^*_{k+1}) = +\infty$, azaz a részprobléma nem megoldható, akkor is vegyünk fel új x_{k+1} változót a következő feltétellel az MP-hez:

   $$Ey + Gx_{k+1} \geq h - Mu^*_{k+1}$$

   Itt u^*_{k+1} az az U-beli vektor, amelyre SP nem megengedett. Legyen $k = k + 1$, és menjünk a 2. lépéshez.

1. Tétel. *Tegyük fel, hogy rögzített y és u vektor mellett létezik x megoldása SP-nek. Legyen p az U poliéder extrém pontjainak száma, vagy ha U véges halmaz, akkor a számossága, ekkor az algoritmus az optimális megoldást találja meg $\mathcal{O}(p)$ lépésben.*

3. Implementálás

A félév során főként azzal foglalkoztunk, hogy a C&CG algortimust implementáljuk az Xpress segítségével, hogy a program írása közben felmerülő problémákat (például hogy hogyan keressük meg u^*_{k+1}-ot az adott iterációban) megoldjuk, illetve hogy különböző inputokon teszteljük az elkészült programot.

3.1. Egy módszer az optimális u megtalálására

Ahhoz, hogy SP ne egy maxmin probléma legyen, egy vele ekvivalens alakra írjuk át. A belső min bx-et úgy kényszeríthetjük arra, hogy az optimális értéket vegye fel adott u mellett, hogy a primál feltétel mellé felvesszük a duális, illetve a komplementaritási feltételeket.

$$\max bx$$

$$Gx \geq h - Ey - Mu$$

$$\pi G \leq b$$

$$(Gx - h + Ey + Mu)_i \pi_i = 0 \ \ \forall i$$

$$(b - \pi G)_j x_j = 0 \ \ \forall j$$

Ahhoz, hogy a k-adik iterációban kiderítsük, hogy $Q(y_k^*)$ esetleg $+\infty$-t vesz-e fel, elegendő SP-nek csak egy részét megnézni, és eldönteni, hogy lehet-e olyan u-t mondani, hogy már ez a rész sem oldható meg. Elég, ha a primál feltételről ($Gy \geq h - Ey - Mu$) döntjük el, hogy megoldható-e. Ha ugyanis meg tudjuk oldani, akkor a dualitás tétel alapján tudjuk, hogy a primál és a duál optimum megegyezik. Ebben a feladatban egy minimalizálási probléma van elrejtve, tehát az még előfordulhatna, hogy a primál optimum $-\infty$, és így a duális feltétel, $\pi G \leq b$ nem elégíthető ki. Ez azoban mégsem történhet meg, hiszen x-ről és b-ről is feltettük, hogy nemnegatív, tehát a 0 mindenképpen alsó korlát. Így tehát ha az első feltételre van megoldás, akkor véges optimumot kapunk, és van olyan x, π primál-duál pár, amely kielégíti a komplementaritási feltételeket, és a teljes részproblémának megoldása.
A $Gx \geq h - Ey - Mu$ feltétel megolhatóságának tesztelését a következőképpen írhatjuk fel.

$$\max_{u \in U} \min \sum_i s_i$$

$$(Gx)_i + s_i \geq (h - Ey - Mu)_i \ \ \forall i$$

$$x \in S_x, \ s \geq 0$$

Amennyiben az optimum értéke 0, akkor a $(Gx)_i + s_i \geq (h - Ey - Mu)_i \ \ \forall i$ feltétel minden u esetén kielégíthető, és továbblépünk a teljes SP megoldásának keresésére. Ha viszont pozitív optimumot kapunk, akkor létezik egy olyan u, ahol legalább az egyik s_i pozitív értéket vesz fel, és így olyan u-t találtunk, amelyre SP nem megoldható.
Ahhoz viszont, hogy ez szintén ne egy maxmin feladat legyen, át kell írnunk. Ha a belső minimalizálási problémának írnánk fel a duálisát α változókkal, akkor ugyan a két max-ot egyesíthetnénk, de a célfüggvényben két változó szorzata állna ($\max \alpha(h - Ey - Mu)$), amelyet az Xpress segítségével nem tudnánk megoldani. Ezért más technikához folyamodunk, és hasonlóan járunk el, mint amikor SP-t írtuk át.

$$\max_{u \in U} \sum_i s_i$$

$$(Gx)_i + s_i \geq (h - Ey - Mu)_i \ \ \forall i$$

$$(\alpha G)_j \leq 0 \ \ \forall j$$

$$\alpha_j \leq 1 \ \ \forall j$$

Továbbá a komplementaritási feltételek:

$$x_i > 0 \implies (\alpha G)_i = 0 \ \ \forall i$$

$$s_i > 0 \implies \alpha_i = 1 \ \ \forall i$$

$$\alpha_i > 0 \implies (Gx)_i + s_i = (h - Ey - Mu)_i \ \ \forall i$$

Ezt a három feltételt a nagy M-módszer segítségével linearizálhatjuk, ahol $\mathbb{M}$ egy megfelelően nagy pozitív szám, χ^1, χ^2, χ^3 pedig bináris vektorok.

$$x_j \leq \mathbb{M} \cdot (1 - \chi_j^1) \ \ \forall j$$

$$(\alpha G)_j \geq -\mathbb{M} \cdot \chi_j^1 \ \ \forall j$$

$$s_i \leq \mathbb{M} \cdot (1 - \chi_i^2) \ \ \forall i$$

$$\alpha_i \geq 1 - \mathbb{M} \cdot \chi_i^2 \ \ \forall i$$

$$\alpha_i \leq \mathbb{M} \cdot (1 - \chi_i^3) \ \ \forall i$$

$$(Gx)_i + s_i \leq (h - Ey - Mu)_i + \mathbb{M} \cdot \chi_i^3 \ \ \forall i$$

3.2. Futásidők

Az algortimus hatékonyságát egy hozzárendelési feladaton teszteltük, ahol n gyárunk és m kliensünk van. Ebben y_i azt határozza meg, hogy az i-edik gyárat megnyitjuk-e, amelynek f_i megnyitási költsége és K_i kapacitása van, x_{ij} pedig azt, hogy az i-edik gyárból mennyit szállítunk c_{ij} költséggel a j-edik klienshez, akinek d_j igénye van.

$$\min fy + cx$$

$$\sum_j x_{ij} \leq K_i y_i \ \ \forall i$$

$$\sum_i x_{ij} \geq d_j \ \ \forall j$$

$$x_{ij} \geq 0, y_i \in \{0, 1\}$$

A bizonytalanságot az adja, hogy nem ismerjük előre a d_j értékeket, csak egy intervallumot, ahová esik. Mivel a legrosszabb eset olyankor következik be, ha az igények növekednek (ekkor ugyanis nőnek a szállítási költségek, de az is lehet, hogy a növekedés újabb gyárak megnyitását teszi szükségessé), a modellbe a következő feltételek kerültek bele:

$$\sum_i x_{ij} \geq d_j + u_j \tilde{d}_j \quad \forall j$$

$$u_j \leq 1 \quad \forall j$$

$$\sum_j u_j \leq \Gamma$$

Azaz minden d_j megnőhet egy előre ismert $\tilde{d}_j \geq 0$ értékkel, de összesen legfeljebb Γ igény változik meg. A tesztelésnél $K_i \equiv 5000$ volt, hogy a kiinduló probléma biztosan mindig megengedett legyen. Γ értékét $0.25m$-nek választottuk, f_i az $[500, 1000]$, c_{ij} a $[0, 100]$, d_j a $[0, 10]$ intervallumon vett egyenletes eloszlásból került ki, és végül $\tilde{d}_j$ megegyezett d_j-vel. A kapott eredmények az 1. táblázatban láthatók.

$n \times m$	Futásidő (mp)	Iterációk száma
5×5	0.078	3
5×5	0.063	3
5×5	0.062	3
5×5	0.114	4
5×5	0.21	5
10×10	0.674	5
10×10	0.297	4
10×10	1.237	7
10×10	0.407	4
10×10	0.508	4
15×15	2.106	5
15×15	1.215	4
15×15	2.444	6
15×15	8.577	13
15×15	5.442	13
20×20	1.697	3
20×20	3.076	5
20×20	6.213	5
20×20	4.349	6
20×20	2.293	3
25×25	37.572	5
25×25	81.219	6
25×25	96.503	7
25×25	25.324	6
25×25	15.558	4

1. táblázat. Futásidők

3.3. Egy hozzárendelési feladat

A C&CG algoritmus egy általános módszer, szeretnénk egy konkrét és a tesztelésnél használt problémánál nehezebb feladatra is alkalmazni, illetve megnézni, hogy az adott feladat szerkezete hogyan használható ki arra, hogy esetleg még hatékonyabbá tegyük az algoritmust. [1]-ben szerepel egy olyan hozzárendelési feladat modellje, amelyben a bizonytalanságot egyrészt az adja, hogy a megnyitott gyárak közül néhányban, de legfeljebb k-ban (k egy előre megadott paraméter) leállhat a termelés például egy természeti katasztrófa vagy sztrájk következtében, másrészt az igények sem ismertek előre. A következő félévben ezzel szeretnénk folytatni, megnézni, hogyan gyorsítható az algoritmus, ha egy ilyen feladaton futtatjuk, illetve magyar adatokat felhasználva megoldani egy ilyen való életbeli problémát.

Hivatkozások

[1] Y. An, B. Zeng, Y. Zhang, and L. Zhao. Reliable p-median facility location problem: two-stage robust models and algorithms. *Transportation Research Part B*, 64:54–72, 2014.

[2] D. Bertsimas and M. Sim. The price of robustness. *Operations research*, 52:35–53, 2004.

[3] B. Zeng and L. Zhao. Solving two-stage robust optimization problems using a column-and-constraint generation method. *Operations Research letters*, 41:457–461, 2013.