Csomagküldési szolgáltatás útvonal-optimalizálása

Oláh Sarolta
Témavezető: Jüttner Alpár

2022. december 18.

1. Bevezetés

Ebben a félévben a járműirányítási feladat (VRP - Vehicle Routing Problem) egy speciális esetével, egy csomagküldési szolgáltatással foglalkoztam.

A járműirányítási feladat olyan problémákra keres megoldást, melyekben a cél termékek szállítása egy vagy akár több raktártól a vásárlókig.

A csomagküldési szolgáltatásnál minden termékhez tartozik egy feladó (forrás) és egy vásárló (nyelő), az útvonalakat ilyen forrás-nyelő párok között kell megfelelően megadni. A járművek száma korlátozott. Célunk többféle lehet, akár az útvonalak költségének minimalizálása, vagy a a kiszállított csomagok számának maximalizálása.

2. Megoldás

A problémát egy gráfon szemléltethetjük. A feladók, a vásárlók és az autók kiinduló pontjai lesznek a csúcsok, éleknek pedig a közöttük menő utak felelnek meg. Költségük az utazási költség. Legyen x_{ijk} változó 1 pontosan akkor, ha a megoldásban az i csúcsból a j csúcsba megyünk a k járművel. Legyen t_i az i csúcs kiszolgálásának időpontja, és jelölje σ_l az l-edik forrás, τ_l az l-edik nyelő csúcsot. Ekkor a modellünk:

$$\min \sum_{i,j,k} c_{ij} x_{ijk} \tag{1}$$

$$t_j - t_i \geq 1 - M(x_{ijk}) \qquad \forall\, k \in K, (i,j) \in A \tag{2}$$

$$\sum_j x_{\sigma_l jk} - \sum_i x_{i\tau_l k} = 0 \qquad \forall\, k \in K, (\sigma_l, \tau_l) \text{ párra} \tag{3}$$

$$t_{\sigma_l} \leq t_{\tau_l} \qquad \forall (\sigma_l, \tau_l) \text{ párra} \tag{4}$$

$$\sum_{j,k} x_{ijk} = 1 \qquad \forall\, i \in V \tag{5}$$

$$x_{ijk} \in \{0,1\} \qquad \forall\, k \in K, (i,j) \in A. \tag{6}$$

A változók száma csökkenthető, ha a járművekre nem tartunk fent külön indexelést. Előjön ekkor a probléma, hogy hogyan különböztessük meg az utakat. Egyik megoldás erre, ha a járműveket különböző időpontokban indítjuk és korlátozzuk utazási idejüket. Így ha az indulási idők kellő távolságra vannak a visszaérkezésektől, akkor az, hogy az adott csúcsot melyik jármű szolgálja ki, meghatározható az elérésének időpontjából.

2.1. IP

A feladat megoldásának egyik módja, a modell közvetlen leprogramozása és egy IP-solver használata. C++-ban, Lemon és Cplex segítségével ezt én is megtettem. Sajnos sok csomag és nagy feladatok esetén a program már megengedett megoldást is nehezen talál, sőt, kisebb értékekre is lassan fut le, ez az egészértékű programokra általánosan is jellemző hátrány. Használata a gyakorlatban nem célravezető.

Ha a feladatból elengedjük az egészértékűségre vonatkozó feltételt, a program lényegesen gyorsabban megold több száz csomagos problémákat is. A gyorsaságért cserébe viszont igen távol kerülhetünk az optimális megoldástól.

2.2. Oszlopgenerálás

Másik megoldás, oszlopgenerálás [1] alkalmazása. Ötlete az, hogy nem tároljuk az egész feladatot, csak egy részét, és erre a részfeladatra határozzuk meg az optimumot. Az így kapott eredmény nem feltétlenül optimális megoldása az eredeti feladatnak, ellenőriznünk kell rá a duál-megengedettséget. Ha a megoldásunk nem duál-megengedett, akkor bővítjük a részfeladatot, a feladat mátrixának egy duált sértő változóhoz tartozó oszlopával, és újból megoldjuk.

A mátrix oszlopai a mostani problémában egy jármű útvonalának felelnek meg, így egy x_p változó pontosan akkor lesz 1, ha az adott útvonalat bevesszük a megoldásba. Ahhoz, hogy egy megoldás duál-megengedettségét ellenőrizzük egy autó költségének és hozamának különbségét kell minimalizálnunk. Ezt az egy autós modellt egy IP-solverrel, vagy heurisztikával megoldva és szubrutinként alkalmazva, már minden adott az oszlopgenerálás használatához.

3. Összefoglalás

Az IP-solveres megoldás kis problémákra is lassan működött, az oszlopgenerálás ezen segíthet, további cél az implementálása. Érdemes lehet még különböző heurisztikákat tesztelni, így gyorsítani a programot.

Hivatkozások

[1] Barnhart, Cynthia, et al. "Branch-and-price: Column generation for solving huge integer programs." Operations research 46.3 (1998): 316-329.