Multiterminális minimális út feladat megoldási módszerei és alkalmalzásai

Bakó András

Két pont közötti legrövidebb út probléma számítástechnikailag leggyorsabb és legkönnyebben programozható megoldását Ford [4] és Dijkstra [7] adta. További speciális minimális út feladat megoldása található az irodalomjegyzékben [10], [11], [15]. A gyakorlati szempontból igen fontos multiterminális minimális út probléma megoldásával igen sok szerző foglalkozott. A dolgozatban összefoglaljuk és számítástechnikai szempontból értékeljük a megoldási módszereket. Végül néhány, gyakorlati szempontból fontos problémát sorolunk fel, amelyek a leírt módszerek valamelyikével megoldhatók.

1. Jelölések, definiciók

Legyen $N = (x, y, \dots)$ véges sok pontból álló halmaz és E az N halmaz bizonyos pontjait összekötő irányított élek halmaza. Jelöljük az x pontból az y pontba vezető élt (x,y)-nal. Az (N, E)-t irányított gráfnak vagy digráfnak nevezzük. Tekintsük az E halmazon értelmezett $t(x,y) \geq 0$ egészértékű függvényt. A $t(x,y)$ függvényt szokás az y pont x ponttól való távolságának nevezni. A $t(x,y)$ függvényről nem tesszük fel, hogy szimmetrikus. Az (N,E,t)-t hálózatnak nevezzük.

A multiterminális minimális út feladat: meghatározni az összes pontpárt összekötő minimális távolságú utakat.

Ha a gráf nem volt teljes, teljessé tesszük a t függvény kiterjesztésével: $t(x,x) = 0$ és $t(x,y) = \infty$, ha $(x,y) \in E$. A fenti feladat megoldását mátrix műveletek, speciális mátrix szorzások egy sorozataként kapjuk.

2. A feladat megoldási algoritmusai

Az első megoldási algoritmust Bellman [1] és Simbell [16] adta. A módszer alapötlete dinamikus programozási elven nyugszik: egy út akkor és csakis akkor minimális, ha mindenegyes részútja is az a megfelelő pontok között.

Az x pontból az y pontba menő legfeljebb r élből álló minimális távolságú utat r-optimálisnak nevezzük. Az optimális, a fenti definiciónak megfelelően $(n-1)$-optimális utakat $n-1$ lépésben kapjuk meg, ahol n az N halmaz pontjainak a száma. A számolásnál kiindulunk a $t(x,y)$ függvényértékekből alkotott $L^{(1)}_{n \times n}$ mátrixból. Ebből meghatározzuk az $L^{(2)}_{n \times n} = (t^{(2)}_{ij})$ 2-optimális utakat. Az $L^{(1)}$ és $L^{(2)}$ mátrixból a 3-optimális utakat. A fenti lépéseket egészen addig ismételjük, ameddig az $L^{(n-1)}$ mátrixot megkapjuk. Az algoritmust a következő mátrix művelettel adjuk meg:

$$L^{(k)}_{n \times n} = L^{(k-1)}_{n \times n} \otimes L^{(1)}_{n \times n} \tag{1}$$

ahol

$$L^{(1)} = (\ell_{ij}^{(1)}) = (t(p_i, p_j)) \qquad p_i, p_j \in N$$

és a $\otimes$ művelet

$$\ell_{ij}^{(k)} = \min_r (\ell_{ir}^{(1)} + \ell_{rj}^{(k-1)})$$

Az $L^{(r)}$ mátrixot kiszámoljuk $r = 2, 3, \ldots, n-1$ értékre az 1 -gyel megadott mátrixművelettel. Ez a módszer módosítható úgy, hogy az út hosszán kívül még az útvonalat is megkapjuk.

A feladat megoldására Warschall [17] 1962-ben közölte le módszerét. A módszer ALGOL programját Floyd [6] írta le. Lényegében ugyanezt az eljárást írja le Murchland [13] és ezt használja fel dolgozatában Hu [8].

Warschall az $L^{(0)} = (\ell_{ij}^{(0)}) = (t(p_i, p_j))$ mátrixból az optimális $L^{(n)}$ mátrixot n egymásutáni mátrix transzformációval kapja.

Az $L^{(k)}$ mátrixot az $L^{(k-1)}$ mátrixból határozzuk meg a következőképp:

$$L^{(k)} = L^{(k-1)} \otimes L^{(k-1)},$$

ahol a $\otimes$ művelet

$$\ell_{ij}^{(k)} = \begin{cases} \ell_{ij}^{(k-1)} & i = k \text{ vagy } j = k \text{ esetén} \\ \min (\ell_{ij}^{(k-1)}, \ell_{ik}^{(k-1)} + \ell_{kj}^{(k-1)}) \end{cases}$$

A Bellman – Simbell módszert Narahary [14] 1961-ben majd Hu [9] 1967-ben továbbfejleszti. Az általuk leírt módszer matematikailag elegáns, könnyen programozható, azonban az eredeti módszerrel szemben csak az útvonal hosszát adja meg, magát az útvonalat nem.

A dinamikus programozási módszer egy más irányú módosítását közli Dantzig [3] 1966-ban. Eljárásában a k. lépésben egy $k \times k$ méretű mátrixot határoz meg, amely a $p_1, p_2, \ldots, p_k$ pontokon átmenő minimális hosszúságú utat adja meg a p_i pontból a p_j $(i, j = 1, 2, \ldots, k)$ pontba.

A $k+1$ -edik mátrix együtthatóját a k. mátrixból a következőképp határozza meg:

a./ kiszámolja az $\ell_{i,k+1}^{(k+1)}$ és az $\ell_{k+1,i}^{(k+1)}$ $(i = 1, 2, \ldots, k)$ elemeket:

$$\ell_{i,k+1}^{(k+1)} = \min_{1 \le j \le k} (\ell_{ij}^{(k)} + \ell_{j,k+1}),$$

$$\ell_{k+1,i}^{(k+1)} = \min_{1 \le j \le k} \left(\ell_{ij}^{(k)} + \ell_{k+1,j} \right)$$

b./ a kiszámolt $k+1$ -edik sor és oszloppal transzformálja az $\ell_{ij}^{(k)}$ $(i = 1,2,\ldots,k,\ j = 1,2,\ldots,k)$ elemeket:

$$\ell_{ij}^{(k+1)} = \min \left(\ell_{ij}^{(k)},\ \ell_{i,k+1}^{(k+1)} + \ell_{k+1,j}^{(k+1)} \right) .$$

Módszerével az optimális úthosszakat tartalmazó mátrixot $n-1$ lépés után kapjuk meg.

3. Számítástechnikai megjegyzések

A Bellmann-Simbell módszernél $(n-1)^4$ összeadást, illetve összehasonlítást kell végezni. Eredményképp az útvonalak hosszát kapjuk. Magát az útvonalat az algoritmus némi módosításával meg tudjuk határozni.

Az eljárást egy kis módosítással gyorsíthatjuk. E célból a $\otimes$ műveletet a következőképp definiáljuk:

$$L^{(m+n)} = L^{(m)} \otimes L^{(n)} ,$$

azaz

$$\ell_{ij}^{(m+n)} = \min_{k} \left(\ell_{ik}^{(m)} + \ell_{kj}^{(n)} \right) .$$

A fenti képlet az optimalizációs elv miatt igaz. A legcélszerűbb 2 hatványai szerint haladni, azaz az $L^{(1)}, L^{(2)}, L^{(4)}, L^{(8)} \ldots$ mátrixokat meghatározni az $L^{(2k)} = L^{(k)} \otimes L^{(k)}$ képletből. Az optimális utak hosszát akkor kapjuk meg, ha valamely k -ra $L^{(2k)} = L^{(k)}$.
A módosítás után az összehasonlítások, illetve összeadások száma $k(n-1)^2$, ahol $k = [\log_2(n-1)] + 1$.

Ez a módszer viszont nem módosítható úgy, hogy az úvonalat is meg tudjuk határozni.

Elektronikus számológéppel való számoláshoz tárolni kell az $L^{(k)}$ és $L^{(2k)}$ mátrixokat – azaz $2n^2$ tárolóhelyre van szükség. FORTRAN nyelven írt program esetén – kihasználva a CDC gépen megengedett 24 bites opciót – elegendő $2n^2$ rövid szó tárolóhely. Számítástechnikai trükkel ez a szám n^2 -re csökkenthető. Tudniillik lehet a két mátrixot ugyanazon a helyen tárolni. Ehhez viszont a mátrix elemei egyik lépésben sem lehetnek 999 -nél nagyobbak.

A Hu [9] algoritmushoz $2n(n-1)$, a Dantzig módszerhez $n(n-1)(n-2)$ összehasonlítása, illetve összeadásra van szükség.

A Warschall eljárásnál az összeadások, illetve az összehasonlítások száma $n(n-1)(n-2)$. Ez több, mint a dinamikus programozási módszer számítási igénye. Viszont ezzel a módszerrel az

útvonalat is könnyen meg tudjuk határozni. Ekkor egy további $M_{n \times n}$ mátrixra van szükség. A mátrix a minimális utaknál szokásos cimkéket tartalmazza. Kiinduláskor $M^{(0)} = \left(m_{ij}^{(0)}\right) = J$.

A k-adik lépésben a mátrix elemeit a következőképp módosítjuk:

$$m_{ij}^{(k)} = \begin{cases} m_{ij}^{(k-1)}, & \text{ha } \ell_{ij}^{(k-1)} < \left(\ell_{ik}^{(k-1)} + \ell_{kj}^{(k-1)}\right) \\ m_{ik}^{(k-1)} & \text{egyébként.} \end{cases}$$

Az $M^{(k)}$ mátrix transzformálásával a számolási idő az eredeti számolási időhöz képest kis mértékben emelkedik. A fent említett módszereket és megjegyzéseket összegezve a következőre jutunk:

a./ Ha az útvonalak hosszára van szükség, a leggyorsabb eljárás a Hu eljárás, majd ezután a négyzetes hatványokkal számoló Bellman-Simbel eljárás.

b./ Ha az útvonalakat is meg akarjuk határozni, legcélszerűbb a Warschall módszert használni.

c./ Ha a megoldandó feladat mérete meghaladja a számológép belső memóriájának kapacitását, akkor a fenti módszerek nem alkalmazhatók. Ha a feladat mérete nem sokkal nagyobb a memóriaméretnél (kb. 50 %-kal nagyobb), akkor Dijkstra [7] módszerét minden pontpárra alkalmazva meg tudjuk határozni a multiterminális utat.

Ha a feladat mérete a fenti korlátnál is nagyobb, akkor a Warschall eljárást használjuk dekompozicióval. Az első dekompoziciós módszert Mills [12] közölte. Számítástechnikai szempontból sokkal célszerűbb és matematikailag is elegánsabb Hu [8] által közölt eljárás. Ezzel az eljárással többszörös dekompozicióval elméletileg tetszés-szerinti nagyságú feladatot meg tudunk oldani. Gyakorlatilag természetesen határt szab a számolási és input-output idő.

d/ Eddig feltettük, hogy $t(x,y) \geq 0$. Ezt a feltételt elhagyva a feladatot megoldhatjuk a Warschall eljárással. Ha negativ ciklus van a hálózatban, akkor a minimális út feladat nincs egyértelműen megfogalmazva. Ciklus keresésére felhasználhatjuk az $L^{(k)}$ mátrix fődiagonálisát. Ekkor azonban $\ell_{ii}^{(0)} = \infty$ az eredeti $\ell_{ii}^{(0)} = 0$ definició helyett. Ha az $\ell_{ii}^{(k)}$ értéke negativvá válik, akkor a hálózatban negativ ciklus van.

4. Megoldható gyakorlati problémák

Mint azt a bevezetésben is említettük, a multiterminális utat meghatározó algoritmusokat más feladatok részfeladatainak megoldásaira is felhasználhatjuk. Alább felsorolunk néhány ilyen jellegű feladatot. További hasonló tipusú alkalmazási lehetőségekkel foglalkozik a [2], [4], [5], [8].

a./ Tekintsük a következő valószínőségszámításban előforduló problémát. A feladatban az $(x,y) \in E$ élen nem a távolságfüggvény van megadva, hanem annak a valószínűsége, hogy az x pont az y ponttal össze van kötve. A feladat: meghatározni az összes pontpár között azt az utat, amely összekötésének a valószínűsége maximális. A $P = (p_0, p_1, \ldots, p_k)$

út létezésének valószínűségét az élek létezési valószínűségeinek szorzataként definiáljuk, azaz

$$v(P) = \prod_{i=0}^{k-1} t(p_i, p_{i+1}),$$

ahol $t(p_i, p_{i+1})$ a p_i és p_{i+1} él létezésének valószínűsége. A feladat megoldható a Warschal eljárással. A $\otimes$ műveletet a következőképp kell átfogalmazni:

$$\ell_{ij}^{(k)} = \max_j \left(\ell_{ij}^{(k-1)}, \ell_{ik}^{(k-1)} \cdot \ell_{kj}^{(k-1)} \right)$$

Megjegyezzük, hogy a feladat megoldható a dinamikus programozási elven alapuló Bellman – Simbell módszerrel is, és ha az útvonal meghatározása érdektelen, a feladat megoldása sokkal gyorsabb.

b./ Egy ciklus-nélküli hálózatban sokszor előfordul, hogy meg akarjuk határozni a leghosszabb utat. Például CPM vagy PERT esetén nem akarjuk megoldani a teljes feladatot, csak a minimális elvégzési időt akarjuk meghatározni.

Az $L^{(0)} = (\ell_{ij}^{(0)})$ mátrixot definiáljuk a következőképp:

$$\ell_{ij}^{(0)} = t(p_i, p_j), \quad \text{ha} \quad (p_i, p_j) \in E \qquad i \neq j$$

$$\ell_{ii}^{(0)} = 0$$

$$\ell_{ij}^{(0)} = -M \qquad (p_i, p_j) \in E$$

esetén, ahol $M \geq 0$ a leghosszabb út egy felső korlátja. A feladat a $\otimes$ műveletet megfelelően definiálva szintén megoldható az 1. alatt elmondott módszerek bármelyikével.

c./ Városi vagy országúti útvonalak, új utak beállításánál, a forgalom hálózatra való ráterhelésnél és egész sor egyéb, ezen témakörhöz kapcsolódó feladatnál részfeladatok megoldására használjuk a multiterminális út probléma valamelyik megoldási algoritmusát.

Irodalom

[1] R. Bellman, "On a Routing Problem", Quart. Appl. Math. 1, 16-20 (1958).

[2] C. Berge, Theory of Graphs, Methuen, London 1962, p. 138.

[3] G.B. Dantzig, All Shortest Routes in a Graph, Operations Research House, Stanford University, Technical Report 66-3, November 1966.

[4] L.R. Ford Jr., Network Flow Theory, The Rand Corporation, P-923, Augus 1956.

[5] B.A. Farbey,– A.H. Land – J.D. Murchland, The cascade Algorithm for finding the minimum distance, Report LSE – TVT – 19, London School of Economics, 1965.

[6] R.W. Floyd, "Algorithm 97, Shortest Path", COM. ACM 5, 345 (1962).

[7] E.W. Dijkstra, "A Note on Two Problems in Connection with Graphs", Numerische Math. 1. 269-271 (1959).

[8] T.C. Hu, "A Decomposition Algorithm for Shortest Path in a Network", Opns. Res. 16, 91-102 (1968).

[9] T.C. Hu, "Revised Matrix Algorithm for Shortest Path", SIAM J. 15, 207-218 (1967).

[10] Klafszki E., "Legrövidebb út meghatározása időtől függő élhosszakkal bíró hálózatban", MTA SzK Közlemények 3, 29-35 (1968).

[11] Komáromi É., "Hálózati feladatok, MTA SzK Szemináriumi füzetek (sajtó alatt).

[12] G. Mills, "A Decomposition Algorithm for the Shortest-Route Problem", Opns. Res. 14, 279-291 (1966).

[13] J.D. Murchland, A new Method for Finding All Elementary Path in a Complete Directed Graph, Transport Network Theory Unit, London School of Economics Report LSE TNT 22 October 1965.

[14] S.N. Narahary Pandit, "The shortest route Problem – an addendum", Opns. Res. 9, 129-132 (1961).

[15] J.P. Saksena and Kumar, "The Routing Problem with "K" Specified Nodes", Opns. Res. 14, 909-913 (1966).

[16] A. Simbell, "Structure in Communication Nets", Proceedings of the Simposium on Information Networks, Brooklyn Polytechnic Institute, New York (1965).

[17] S. Warsall, "A Theorem on Boolean Matrices", J. ACM 9, 1-12 (1962).

Summary

The solution methods and applications of the multiterminal minimal path problem.

Ford [4] and Dijkstra [7] have given the fastest and most easily programmable solution of the minimal path problem between two points as regards the computational technique. Further solutions of special minimal path problems can be found in the list of literature [10], [11], [15]. Many authors dealt with the solution of the minimal path problem, so important from a practical point of wiev. The solution methods summarized and assessed from the angle of computational technique, liable to be solved by one of the described methods, are enumerated.

Р е з ю м е

Методы решения и приложения задач мультитерминального пути.

Найбыстрейшее с точки зрения вычислительной техники и легко программируемое решение проблемы кратчайшего пути между двумя точками даны Фордом [4] и Дейкстра [7] Дальнейшие решения специальных задач минимальных путей даны в списке литературы [10], [11], [15] . Проблема мультитерминального минимального пути с точки зрения практики очень важна и много авторов занимались ей.

В настоящей работе суммируем и с точки зрения вычислительной техники оцениваем методы решений. Наконец перечисляем несколько практически важных проблем, которые решаются одним из вышеупомянутых методов.