Conference Paper for MAFIOK 2016.
Óbuda University Alba Regia Technical Faculty, Hungary, Székesfehérvár

Helyi buszközlekedés segítő szoftver speciális gráf szerkezetének tervezése

Zsobrák Krisztián*, Gugolya László**
* OE AMK, Székesfehérvár, Magyarország
** OE AMK, Székesfehérvár, Magyarország
zsobrak.krisztian@gmail.com, gugolya.laszlo@amk.uni-obuda.hu

Kulcsszavak: gráf, mobil alkalmazás, sql

***Kivonat*—Modern világunkban egyre nagyobb jelentősége az informatikai alkalmazásoknak. Különösen az olyan mobilalkalmazásoknak, melyek kényelmesebbé tehetik az életünket. Az informatikai alkalmazások gyakran használnak összetettebb matematikai struktúrákat, algoritmusokat. Az egyik jelentős, sokszor használt matematika tudományi terület a gráfelmélet, s így a gráfelméleti algoritmusok.**

A publikációban bemutatásra kerül egy speciális gráf kidolgozása. A gráf alapjául szolgál egy olyan szoftveres megoldásnak, amely a helyi járatos közlekedésen alapul. A megoldáshoz a menetrendi adatok tárolása egy központi adatbázisban történik. Ebből felépítve az aktuális gráfot lehet műveleteket végezni. A tervezéskor a székesfehérvári helyi járatos menetrend lett alapul véve. A rendszer úgy lett felépítve, hogy a megvalósításakor változtatások nélkül átültethető legyen bármely más város busz alapú helyi közlekedésének kezelésére is. Ismertetésre kerül a megfelelő adatstruktúrák kialakítása, a menetrendi ismeretek alapján ezek feltöltése. Majd részletezésre kerül, hogy az így eltárolt (SQL szerveren) adatok hogyan alakíthatók át olyan gráf adatszerkezetté, amelyen a keresés elvégezhető. A keresés alapja, hogy egy adott helyről (kiindulási állomás) milyen módon juthatunk el a célig (végállomás). A keresés során térben és időben is kell keresni. Ez okoz nehézséget a megvalósítás során. Hiszen az egyik helyről a másikra kell eljutni, de ezek a lépések csak bizonyos időpontokban lehetségesek. A megoldáshoz választott modell mind a két dimenziót egyben tudja kezelni.

Bemutatásra kerül az gráfépítő-tároló algoritmus, ami alapja egy olyan készülőben lévő szoftvernek, ami Androidra készül el. A szoftverfejlesztés során a Java, SQLite, mySql eszközök kerülnek felhasználásra. Az elkészült alkalmazás tervezetten a Google Play-be kerül.

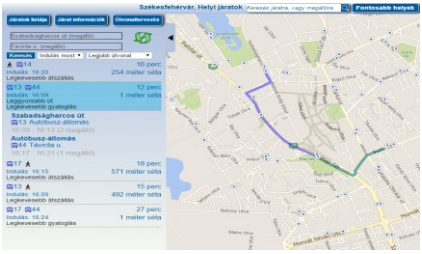
1 BEVEZETÉS

Rohanó világunkban egyre nagyobb szükségünk van olyan alkalmazásokra melyik megkönnyítik életünket. Amelyekkel úgy érezzük, hogy nem kell fölösleges információkat megjegyeznünk, s ha ezekre szükségünk van, akkor azt bárhol a mobiltelefonunk segítségével elérhessük. Ezért jutott eszünkbe az a gondolat, hogy a már meglévő közlekedéssegítő rendszerekhez hasonlóan Székesfehérvár helyi közlekedésére is megvalósítsuk az útvonalkeresést, tervezést.

Jelenleg a KNYKK Zrt. honlapján az egyik lehetőség szerint statikusan járatonként juthatunk az információkhoz.

1. ábra – székesfehérvári statikus járat kereső

A másik felület a térképes útvonaltervező, ahol ki lehet választani térképen, hogy melyik megállóból kívánunk indulni, illetve melyik a célállomás.

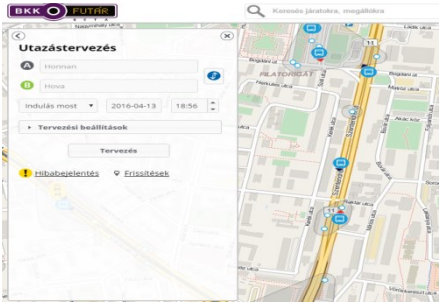


2. ábra – székesfehérvári térképes útvonaltervező

Ez a felület viszont erőforrás igényes, és egy táblagépen vagy mobiltelefonon is akadozik.

Követendő példának a budapesti BKK Futár szoftveres megoldást tekintettük.

Köszönhetően a letisztult megjelenése, és hasznos funkció miatt. A Futárnak van egy webes felülete, és egy mobil applikációja is, mindkettő azonos funkciókat nyújt a felhasználó számára. Az indulási és célállomást nem kell a térképen megjelölni, egyszerűen be lehet gépelni. Nem csak megállókra, de utca és házszám alapján is lehet vele keresni.



3. ábra – Képernyőkép a BKK Futár weboldalról

A szoftver megvalósításához a mobil platform okán a Java[1] nyelv lett választva, az adatbázis MySql[2], illetve SQLLite használatával valósul meg.

2 HELYI JÁRATOS MENETREND BEMUTATÁSA, ADATBÁZISTERV

Jelenleg Székesfehérváron kizárólag busz alapú helyijáratos közlekedés van. A buszmegállók a város egész területén helyezkednek el. A megállóhelyekhez általában két buszmegálló tartozik, az út két oldalán, így meg tud valósulni az oda-vissza történő közlekedés. Léteznek még úgynevezett körjáratok, ahol a kezdő és végállomás ugyan az. A menetrendben a

járatok elnevezése két számjegyből áll, némely esetben egy betűjellel ellátva. Az első számjegy a kiinduló állomást jelzi (Pl: 1x - Szedreskert ln., 2x - Jancsár, stb). A betűjel pedig az eredeti járat módosított útvonalán közlekedik (gyorsított-, iskolai járat, stb). A gráfhoz szükséges adatok tárolását a már említett relációs adatbázis-kezelővel lett tervezve, megvalósítva.[3]

A. Állomások, buszmegállók

A legtöbb esetben az út egy oldalán csak egy megálló helyezkedik el, de bizonyos állomásokon, ahol nagy az utasforgalom, illetve sok járat érinti azt, egy oldalon több megálló is épült, így biztosítva a zavartalan közlekedést. Az egyszerűség kedvéért menetrendben a buszmegállók nincsenek külön meghatározva, csak az állomások nevei szerepelnek benne. Amikor valaki útvonalat kíván tervezni egy kiválasztott állomástól egy másikig (például az Autóbusz pályaudvarról a Vasútállomásra), szintén nem számít, pontosan melyik buszmegállóból kell indulnia, neki bármelyik megfelelhet. Viszont amikor több megálló közül ki kell választani, hogy ténylegesen melyikben fog a járat megállni, akkor érdemes a felhasználónak megjeleníteni a térképen az adott buszmegállót. Egy másik lehetőség, amikor a mobil készülék jelenlegi helyzete alapján kell útvonalat keresni, akkor ott GPS koordináták alapján kell visszakövetni, hogy melyik buszmegálló van hozzá közel, hiszen csak az rendelkezik pontos pozícióval, míg az állomás nem. A megfelelő adatszerkezet állomások információinak tárolásához.

1. TÁBLÁZAT

Azonosítószám	id	egész szám
Név	nev	szöveges

Ez a név az, ami a menetrendben is szerepel, és közismert (például: Vasútállomás, Távirda utca, stb).

A buszmegállók adatait is kívánjuk tárolni, erre pedig a megfelelő adatstruktúra a következő.

2. TÁBLÁZAT

Azonosítószám	id	egész szám
Állomásazonosító	allomasid	egész szám
Szélességi fok	lat	valós szám
Hosszúsági fok	lon	valós szám

Az allomasid mezővel tudjuk hozzárendelni a buszmegállót ahhoz az állomáshoz, amelynek a részét képezi.

B. Járatok

Az utasok a számuk szerint ismerik a járatokat, például 10-es, vagy 23E. A legtöbb ilyen járat oda-vissza közlekedik, a két végállomás között. A tervezés során úgy döntöttem, hogy külön kell tárolni ezeket, a könnyebb kezelhetőség végett. Innentől kezdve az alábbi kifejezéseket használom a dolgozatban:

- **Vonal**
  Több járatot foglal magában, számjellel rendelkezik. Pl: 10, 11A, 15Y. A menetrendben ez fog megjelenni a felhasználónak.
- **Járat**
  Jól meghatározott útvonala van, állomásnevekkel, időpontokkal. Egyszerűsített neve nincs, általában: Szedreskert ln. - Autóbuszpályaudvar – Könnyűfémmű. Legalább egy járat tartozik minden vonalhoz. A szoftveren belül egyszerű betűjelekkel van jelölve, például: o, v (mint oda, vissza).

Bizonyos vonalakon az egyik járat egy megállóval többet érint, így más a menetideje is. Ezeket külön járatokként tárolja majd a rendszer.

A megfelelő adatstruktúra a vonalak tárolására:

3. TÁBLÁZAT

Azonosítószám	id	egész szám
Név	nev	szöveges (4 karakter)

A járatokat pedig a következő adatstruktúra tárolja:

4. TÁBLÁZAT

Azonosítószám	id	egész szám
Vonal azonosító	vonalid	egész szám
Betűjel	jel	szöveges (5 karakter)

C. Útvonal

A menetrendben a különböző járatokhoz megmutatja, hogy milyen állomásokat érint, illetve az indulástól számítva hány perc alatt fog oda eljutni. Mivel az állomások és a járatok adatszerkezete ismertetésre került, így az útvonal tárolásához már kevesebb adatot kell tárolni:

5. TÁBLÁZAT

Azonosító	id	egész szám
Vonalazonosító	vonalid	egész szám
Állomásazonosító	allomasid	egész szám
Perc	perc	egész szám

Itt az azonosító csak a bejegyzésnek ad egy egyedi kulcsot. A perc tárolja, hogy hány perc alatt lesz az adott megállóban az adott járat. A különböző járatok útvonalait lehet egy egységes adattáblában tárolni, mivel későbbi szűréssel visszanyerhető egy bizonyos járat adata. A megfelelő sorrend visszaállításáról a perc adattag szerinti növekvő rendezés gondoskodik.

D. Indulási időpontok

A menetrendi információ része még a különböző járatok indulási időpontja, melyek minden esetben órában és percben vannak megadva. Mivel az útvonalnál szintén elegendő csak perc adatokat tárolni, a számítások során felesleges egy percnél kisebb időegységeket figyelembe venni. Valamint az indulási információknál sem fontos számolni óránál nagyobb időegységgel. Így az időpontok tárolására a mindenkori éjféltől eltelt percek használatát választottam.

A számítás módszere a következő:

A “h” óra “m” perc időpont esetén a h*60+m érték lesz tárolva.

A tárolt p szám esetén pedig az óra és a perc értékek visszanyerhetők h=p/60 egészrésze, és m=p mod 60 alapján.

A különböző járatokat több ember veszi igénybe munkanapokon, és munkaszüneti vagy szabadnapokon. A menetrendben külön indulási időpontok tartoznak a munkaszüneti és szabadnapokhoz. Illetve egyes indulási időpontokhoz a munkanapon kívül jelölésre kerül, hogy csak iskolai előadások napján, a hét utolsó munkanapján, stb közlekedik. Ezt az alábbi adatszerkezet tárolja.

6. TÁBLÁZAT

Azonosítószám	id	egész szám
Betűjel	betu	szöveges (1 karakter)
Megnevezés	nev	szöveges (30 karakter)

Későbbiekben annak érdekében, hogy a program ismerje egy adott napon történő keresés esetében mely indulási időpontokat kell figyelembe vennie, naptárszerkezetben kell tárolni, hogy az adott naptári naphoz mely naptípus tartozik. Ez nem kerül bemutatásra ebben a dolgozatban.

Érdemes úgy feltölteni az adatokat, hogy a szabadnap és a munkanapokhoz tartozó típusok elkülönüljenek egymástól, például 1-es azonosítója a szabadnap, 2-es a munkanap, 3-as az iskolai előadási nap, stb. Ez alapján, ha a felhasználó munkanapokon közlekedő járatokra kíván keresni, a keresési feltétel szimplán id>=2 feltétellel megadható. Természetesen ekkor figyelmeztetni kell,

ha a felkínált járat csak bizonyos típusú napokon közlekedik.

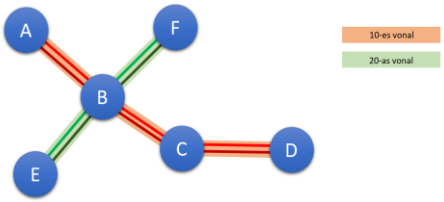
Az indulási időpontokhoz tartozó adatszerkezet a következő:

7. TÁBLÁZAT

Azonosítószám	id	egész szám
Járatazonosító	jaratid	egész szám
Időpont	ido	egész szám
Nap típus	ntp	egész szám

E. Minta menetrend

Szemléltetésre egy minta menetrend és a hozzátartozó adatbázis kerül bemutatásra.



4. ábra – A minta menetrendhez tartozó város vázlatos térképe

Egy városban hat állomás helyezkedik el, mindegyikhez két megálló tartozik. Két vonalon közlekednek a járatok: 10-es és 20-as. Továbbá mindkét vonalon egy-egy járat, oda-vissza közlekedik. Ezek az egyszerűség kedvéért az adatbázisban "o" és "v" betűvel lesznek jelezve.

A menetrendi információk a következők:

8. TÁBLÁZAT

10, A-ból D-be (o)	
A	0
B	2
C	6
D	8
8 óra	00, 20

10, D-ből A-ba (v)	
A	0
B	2
C	6
D	8
8 óra	10, 30

20, E-ből F-be (o)	
A	0
B	2
C	6
D	8
8 óra	05, 25

20, F-ből E-be (v)	
A	0
B	2
C	6
D	8
8 óra	15, 35

F. Gráf felépítése a tárolt információk alapján

Az adatok feltöltése után következik az útvonaltervezés kérdése. A kereséshez szükséges a kiinduló és cél állomást, valamint az indulási időpontot ismerni. Ez sokféleképpen megoldható. Egy nagyon egyszerű megoldást a következő:

1. Válasszuk ki azokat a járatokat, amelyek az induló és cél állomást is tartalmazzák.
2. Az adathalmaz további szűrése azokra a járatokra, amelyek előbb érintik az induló, és később a cél állomást.
3. Keressük ki azokat az indulási időpontokat, amelyek ezekhez a járatokhoz tartoznak.
4. Az indulási időpontokhoz adjuk hozzá a kiinduló állomáshoz tartozó perc értéket (mennyi idő alatt ér oda a járat).
5. Az eredményeket rendezzük kiindulási idő szerint növekvő sorrendbe.

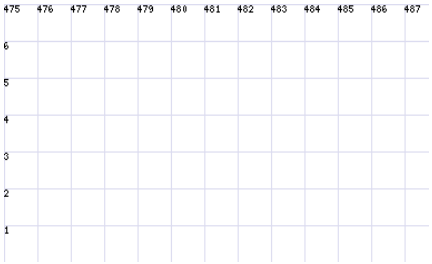
Ez az eljárás működik abban az esetben, ha az induló és cél állomás között van közvetlen járat. Viszont gyakran előfordul olyan eset, melyben egy vagy több alkalommal is át kell szállni.

A mintaadatbázisban szereplő menetrend szerint az "A" állomásról az

"F"-be csak átszállással lehet eljutni. A térképről egyértelműen leolvasható, hogy ez "B" fog történni, de kérdéses, hogy van-e értelme elindulni, tehát fog-e jönni csatlakozás.

A menetrend szerint ismerjük, hogy egy adott busz hol fog tartózkodni melyik időpontban. Hogy egyszerűbb legyen kezelni és ábrázolni térbeli helyzetet, elég ezt egy dimenzióban megtenni, az adott hely koordináta értéke az állomás azonosítója. A mintamenetrend szerint az "A" állomás koordinátája 1, a "B" állomásé 2, stb.

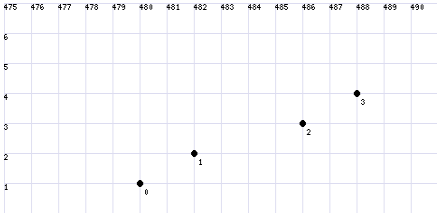
Az időbeliséget szintén elegendő egy dimenzióban ábrázolni, melynek az alapegysége a perc, a már ismertetett megfontolások alapján.



5. ábra – Koordinátarendszer a gráf ábrázolásához

Az ábrán az időbeliség ábrázolása a vízszintes tengelyen történik, míg a térbeliség a függőleges tengelyen.

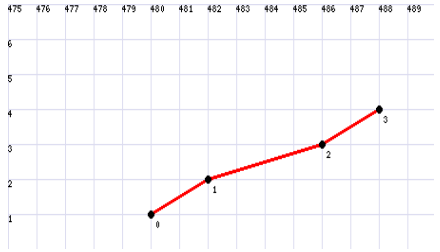
A gráf minden pontjához két érték tartozik, a térbeli (állomásazonosító) érték, és az időbeli (éjféltől eltelt percek) száma. Ez fogja szimbolizálni, hogy az adott időpontban az adott helyhez információ tartozik.



6. ábra – A járat által érintett állomások és időpontjuk

A gráfban a pontokat élek kötik össze, melyek azt jelölik, hogy az adott pontból el lehet jutni egy másik pontba, azaz egyik helyről a másikra, a megfelelő időpontban. Az élekhez tartozó információk:

- *Kiindulás*: Hivatkozás a kiinduló gráfpontra az azonosítószámával
- *Cél*: Hivatkozás a cél gráfpontra az azonosítószámával
- *Típus*: Az adott él busszal történő utazást, vagy az várakozást jelöl
- Járat *azonosító*: Ha busszal való utazást jelöl az él, akkor mely járathoz tartozik
- *Idő*: Időbeli különbség az él két végpontja között
- *Súly*: Az él súlya, mely az útvonaltervezés során játszik szerepet.



7. ábra – Az élek jelképezik a járatot

Példa: adott két pont, és egy él.

9. TÁBLÁZAT

	pont0	pont1
azonosító	0	1
ido	480 (8:00)	482 (8:02)
allomasid	1	2

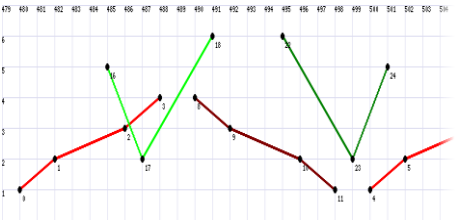
Él adatai:

- Típus: utazás
- Indulási pont: 0
- Érkezési pont: 1
- Járat azonosító: 1
- Idő: 2
- Súly: 2

Az adatok alapján leolvasható, hogy az 1-es azonosítójú járat (10-es oda irány) 8:00-kor érinti az 1. azonosítójú (“A”) állomást, majd arra felszállva 2 perc múlva a 2. azonosító számú állomásra (“B”) lehet eljutni.

A gráf felépítése a következő módon történik:

1. Minden indulási időpont lekérdezése, majd ezek egyesével történő feldolgozása
2. Az adott indulási időponthoz tartozó járat útvonalának lekérdezése, és feldolgozása
3. Az útvonal adatai alapján ismert az állomás azonosító, valamint az indulástól számított perc érték, tehát időpont. Ha 8 órakor indul a járat, és az adott megállóba 5 perc alatt ér, akkor 8:05 a gráf pontjához tartozó érték
4. Ha még nem létezik ilyen állomás azonosítóval és perc értékkel rendelkező csúcs, akkor az létrehozásra kerül, különben a meglévő kerül felhasználásra
5. Új él létrehozása az előző érintett pont, és a mostani pont között az adott járat azonosítójával.



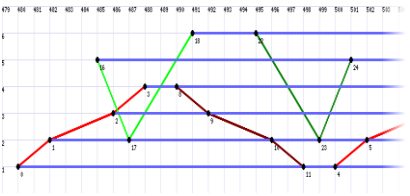
8. ábra – Az összes járat felvitele utáni gráf

Ezzel már az összes, a buszok által érintett útvonal szerepel, viszont átszállással történő utazás tervezésére nem alkalmas, mivel nem lehetséges az adott állomáson történő várakozás.

A gráf felépítésének a második lépése a várakozási élek létrehozása. Ennek folyamata a következő:

A program a gráf pontjai közül sorban kigyűjti a különböző állomásazonosító számúakat, ha ezekből 2 vagy több van, akkor az időbeli tulajdonságuk alapján növekvő sorrendben rendezi, majd új éleket hoz létre, melyek típusa várakozás.

Az ábrán ez vízszintes vonalak formájában jelenik meg, mivel időbeli változás történik, de térbeli nem.



9. ábra – A várakozási élek hozzáadása után, a teljes gráf

A gráf élei irányítottak, mivel időben csak az egyik irányban lehet haladni. Körjáratok esetén a járat induló és végpontja ugyan abban a sorban helyezkedik el.

A gráf csúcsaihoz legalább egy, leggyakrabban 3-4 él kapcsolódik, így elmondható, hogy az élek száma hozzávetőleg a csúcsok számának

másfél- kétszerese. A példában bemutatott gráf 28 csúcsot és 42 élet tartalmaz, melyből 20 él kapcsolódik járatokhoz, és 22 várakozás típusú.

Így a legmegfelelőbb módszer a gráf tárolására a listás (vektros) tárolás. Az egyik lista a gráf csúcsait tárolja, a másik pedig az éleket. A két lista között a kapcsolat az élekben van tárolva, mivel annak kezdő és vég változója a csúcsokat tartalmazó vektor elemére mutat, a vektorban elfoglalt helye alapján.

3 Összegzés

Az előbbiek alapján elérkeztünk oda, hogy az adatbázisban tárolt menetrendből felépíthető egy kezelhető adatstruktúra. Ezen gráf kereső algoritmusok alapján már lehet útvonalat tervezni. A megfelelő algoritmus kiválasztásához tesztelések szükségesek. Az Androidos szoftver jelenleg fejlesztés, tesztelés alatt áll van. A megoldáshoz a Google Maps [5] szolgáltatása kerül felhasználásra. Székesfehérvár buszközlekedése éppen átszervezés alatt van. Remélhetőleg az új menetrenddel egyszerre megjelenhet a elkészült szoftver.

Irodalom

[1] Julie C. Meloni, Tanuljuk meg a MySQL használatát 24 óra alatt Kiskapu Kft., 2003

[2] Bert Bates, Kathy Sierra, Agyhullám: Java Kiskapu Kft., 2011

[3] Timár Lajos,Vígh Kriszta,Tátrai József, Szigeti Judit, Vathy Ágnes, Építsünk könnyen és lassan adatmodellt! Veszprém, 1997

[4] Szabó Zsolt: a BKV FUTÁR-Forgalomirányítási és UTastÁjékoztató Rendszer, OEKVK-HI szakdolgozat, 2013

[5] Google Developers - Google Maps Geocoding API (https://developers.google.com/maps/documentation/geocoding/intro#Geocoding, 2016.04.10)