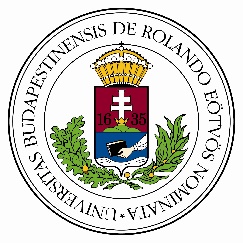
**Eötvös Loránd Tudományegyetem  
Informatikai Kar  
Adattudományi és Adattechnológiai Tanszék**

**Keretrendszer nagy erőforrás igényű számítások elosztott végrehajtására**

**Kiss Péter Polozgai Máté**

Doktorandusz, programtervező informatikus (Msc) nappali tagozat

programtervező informatikus

Budapest, 2018

**Köszönetnyilvánítás**

Ezúton szeretném megköszönni a témavezetőmnek, Kiss Péternek, hogy a szakdolgozat ötletemet felkarolta, valamint szakértelmével a rendszeres konzultációk alkalmával tanácsaival segítséget nyújtott a dolgozat elkészítéséhez.

Köszönettel tartozom szüleimnek, valamint az egész családomnak, hogy a türelemmel, és megértéssel támogatták tanulmányaimat, valamint minden helyzetben mellettem álltak.

Valamint szeretném hálásan megköszönni az Eötvös Lóránd Tudományegyetem Informatikai Kar valamennyi oktatójának, dolgozójának azt a kitartó munkáját, amivel hozzájárulnak ahhoz, hogy a hallgatók magas színvonalú, és versenyképes tudáshoz jussanak.

**Köszönöm!**

Tartalom

# 1. Bevezetés

## 1.1 A szakdolgozat ötlete

Napjainkban az informatika tudományban nagy szerepet tölt be, hogy a komplex, nagy számításigényű feladatokra, amik meghaladják a lokálisan igénybe vehető erőforrásainkat, olyan elosztott keretrendszert tervezzünk, amivel külső erőforrást igénybe véve, az adott feladat végrehajtása optimálisabbá váljon, futási ideje pedig jelentősen lerövidüljön.

A témához igazodva egy olyan az elosztott végrehajtást támogató rendszer megvalósítása volt a cél, ami egy speciális gráf algoritmust valósít meg. Az algoritmussal irányítás nélküli gráfokban lehet megkeresni a legrövidebb utakat egy adott csúcsból kiindulva. Előfeltételként teljesülnie kell, hogy a gráfnak összefüggőnek kell lennie, valamint nem szabad hurokélt tartalmaznia.

A gráfok és a velük kapcsolatos algoritmusok alapvető szerepet játszanak az informatika tudományban. Ezek közül talán a legjelentősebb, és a legtöbb felhasználási területtel rendelkező a legrövidebb utak problémája, aminek segítségével például olyan hétköznapi problémára kaphatunk egyszerű és kézzel fogható megoldást, hogy hogyan jutunk el A városból B városba.

Bár ezeknek a megoldására már számos algoritmus született, a szakdolgozat legfőbb célja, hogy rámutasson azokra az elosztott számítási módszertanokra, valamint bemutassa az ezekhez kapcsolódó technológiákat, mely során az algoritmus az adatmozgatás minimalizálására törekszik. Az eredmény pedig egy weboldalon kerül vizualizálásra.

Elsőként az alapvető gráfelméleti fogalmakról esik szó, amelyek elengedhetetlenek a szakdolgozat megértéséhez, utána fog következni a felhasználói és a fejlesztői dokumentáció.

# 2. Gráfelmélet

## 2.1 Fogalmak jelölések

Ebben a fejezetben lesznek megemlítve, azok a jelölések, és fogalmak, amelyekre szükségünk van a téma kifejtéséhez, illetve amelyek a szakdolgozat megértését szolgálják.

**Definíció (irányítatlan gráf):** Egy G irányítatlan gráfot a G=(V,E) rendezett párral jellemezzük, ahol:

* V a csúcsok halmaza, mely egy véges nem üres halmaz
* E az irányítatlan éleknek megfelelő csúcsok rendezetlen párjainak halmaza.

Az e={u,v} élről azt mondjuk, hogy az u és v között fut, összeköti u-t és v-t.

Ebben az esetben, ahogy a bevezetésben már utaltunk rá, a csúcsokra objektumokként is tekinthetünk, így a köztük lévő kapcsolat jelentheti út vagy kommunikáció lehetőségét. Ilyenkor rendszerint költségek vagy súlyok tartoznak az élekhez. Ezt a kapcsolatot egy valós értékű függvénnyel fogjuk leírni, melynek értelmezési tartománya a gráf él halmaza, az érték készlete pedig a valós számok halmaza és ce-vel jelöljük, tehát ce : E R. A súlyokat az élekre szokás írni. ???

# 2. Felhasználói dokumentáció

A következőkben ismertetésre kerül a program telepítése, összes funkciója, azoknak pontos használata. Továbbá részletezésre kerül a felhasználó számára szükséges lépések a program használatához.

## 2.1 Minimális rendszerkövetelmények

A felhasználónak az alábbiakra van szüksége ha futtatni szeretné a programot:

* Ajánlott operációs rendszer: Windows 10
* Java SE Runtime Environment 8
* Ajánlott webböngésző: Google Chrome

## 2.2 A program telepítése

A program telepítésére két lehetőségünk van. Ha csak futtatni szeretnénk a programot a szakdolgozathoz mellékelt CD-n megtaláljuk a ??? nevű ".jar" kiterjesztésű fájl, amire kétszer kattintva a program automatikusan elindul, és "http://localhost:4567" című URL-t kell megnyitnunk egy általunk választott böngészőben. A projekt fejlesztése során a Google Chrome nevű böngészőt használtam. Az így elénk tároló grafikus felhasználói felület használatáról a későbbiekben esik szó.

Amennyiben a teljes projektet szeretnénk megnyitni, ez megtaláljuk a szakdolgozathoz tartozó CD mellékleten, vagy a <https://github.com/polozgai/szakdolgozat> weboldalról is letölthetjük a „Clone or download” gombra kattintva azon belül pedig a „Download ZIP”-re. Miután letöltődött csomagoljuk ki a projektet, majd nyissuk meg egy fejlesztői környezetben.

???

2.1 ábra. A teljes ptojekt letöltése GitHubról

## 2.3 A program megnyitása Intellij IDEA

Miután a projekt fejlesztése közben az Intellij IDEA fejlesztői környezetet használtam, ezért ez az ajánlott fejlesztői környezet, de mivel a projekt a Maven-t használja a build folyamatok automatizálására, ezért más fejlesztői környezetben, például Eclipse, Netbeans is könnyen tudjuk importálni.

Intellij-ben a "File" >> "Open..." menüpontok segítségével válasszuk ki azt az elérési útvonalat, ahova a projektet kicsomagoltuk, majd az "OK" gombra kattintva várjuk meg, amíg a projektünk betöltődik.

## 2.3 A program funkciói

Ebben a fejezetben bemutatjuk a felhasználó számára, hogy mely funkciókat hogyan érhet el és azokat hogyan használhatja.

### 2.3.1 A program indulása

A program indulásakor a felhasználó a választott böngészőjében a "http://localhost:4567" URL címen érheti el a grafikus felhasználói felületet. A megjelent képernyő felső részében láthatóvá válik a gráf összes csúcsa, a "Start node" és "End node" feliratok, a mellettük található szövegdobozok, és a három gomb: a "Basic", az "Animation", és a "Submit". Ezeknek a gomboknak a segítségével tudja inicializálni az algoritmust, és az algoritmus működését kiválasztani. Ezek alatt található meg az a szürke hátterű panel, melyben a gráfot, és a két választott csúcs közötti legrövidebb út kerül szemléltetésre.

## 2.4 Felhasználói esetek

### 2.4.1 A gráf beolvasása

??? A gráf megadása a graph.txt-ben történik, méghozzá a gráf éleinek a megadásával: Először meg kell adni a gráf éleinek kezdő és végpontját szóközzel elválasztva, majd szóközzel elválasztva az adott élhez tartozó él súlyt. Ügyeljünk arra, hogy minden ilyen élt új sorban kell megadni, valamint, hogy a gráfunk összefüggő legyen és ne tartalmazzon hurokélt.

### 2.4.2 Az algoritmus inicializálása

A böngésző ablak bal felső részében található "All nodes" felirat mellett találhatja a felhasználó a gráf összes csúcsát, melyek közül kettőt kell beírnia a "Start node", és "End node" feliratok melletti szövegdobozba, majd vagy az Enterrel, vagy a szövegdobozok mellett található "Submit" gombra kattintva adja meg a kezdőcsúcs és a végcsúcs cimkéjét. Amennyiben nem a felsorolt csúcsok közül választ kezdő, illetve végpontot, vagy amennyiben valamelyik csúcs értékét üresen hagyja, a program egy figyelmeztető dialógust dob fel a böngésző ablakában.

### 2.4.3 A "Basic" gomb funkciója

Amennyiben a felhasználó hibaüzenet nélkül megadta a kezdő, illetve a végpontot, és amennyiben a "Basic" gombra kattint, akkor a képernyő szürke hátterű panel részén az algoritmus ki fogja rajzolni az egész gráfot, és a korábban megadott két csúcs között a legrövidebb utat tartalmazó éleket pirossal rajzolja ki.

### 2.4.4 Az "Animation" gomb funkciója

Amennyiben a felhasználó hibaüzenet nélkül megadta a kezdő, illetve a végpontot, és amennyiben az "Animation" gombra kattint, akkor a képernyő szürke hátterű panel részén az algoritmus működése lépésről-lépésre be lesz mutatva, mégpedig a következő módon:

1. A grafikus felületünkön a kezdő ponttól kiindulva csúcsról-csúcsra a gráfban található utolsó csúcsig, függetlenül attól, hogy melyik csúcsot választottuk végpontnak megjelennek a gráf csúcsai zöld háttérrel.
2. Az előbb említett sorrend fordítottjából haladva kezdi meg az algoritmus a futásának második szakaszát. A zöld gráf csúcsok pirosra váltanak, vagyis arról a csúcspontról az algoritmus a működéséhez szükséges minden információval rendelkezik. A szükséges módosításokat a csúcspont attribútumain végrehajtotta, ezért ismét csúcsról-csúcsra lépeget vissza, egészen a kezdő csúcsig.
3. A gráf csúcsai közötti lépegetés során amennyiben a kezdő csúcs így talál egy esetleges legrövidebb utat, ennek az útnak az éleit kiszínezi piros színnel. Ha az algoritmus következő lépéseiben talál egy az előbb említett útnál kevesebb összköltséggel bírót, akkor az előbb említett utat törli, vagyis az éleit visszaállítja kék színre, és ennek az új útnak az éleit színezi ki piros színnel.
4. Az algoritmus futása akkor ér véget, ha a gráf összes csúcsa piros színű lesz, és az így látható piros színű élek alkotta út lesz a legrövidebb út a korábban választott két csúcs között.
5. Megjegyzés: Ha az "Animation" funkciót választjuk, meg kell várnunk az algoritmus befejezését, ezáltal a funkció gombok letiltásra kerülnek, hogy a felhasználó még véletlenül se tudja hibás működésre késztetni a programot.

# 3. Fejlesztői dokumentáció

A következő fejezetek tartalmazni fogják a program részletes felépítését, a felhasznált technológiák leírását, valamint ezek felhasználását a projektben. Ismertetésre kerül az algoritmus részletes működése, szerver oldal, valamint az algoritmus működésére szolgáló weblap részletes leírása. Leírásra kerül az algoritmus tesztelési folyamata is. A programnak az algoritmus logikája, valamint a szerver forráskódja Java nyelven íródott. A weblap HTML melyhez a CSS stílusleírót használtam, ezen a weblapon történő szemléltetésre a Javascript programozási nyelvet használtam. A projekt fejlesztése során ügyeltem arra, hogy a forráskódban szereplő egymástól jól elkülöníthető

## 3.1 Minimális rendszerkövetelmények a fejlesztéshez

A felhasználónak az alábbiakra van szüksége ha megszeretné nyitni a projektet:

* Windows operációs rendszer (Ajánlott: Windows 10)
* Java SE Development Kit 8
* Apache Maven
* Ajánlott fejlesztői környezet: Intellij IDEA
* Ajánlott web böngésző: Google Chrome

## 3.2 Forráskód letöltése

Ez a fejezet első része a szoftver fejlesztése során felhasznált verziókezelő megismertetésére, valamint szoftver letöltési lehetőségeire szolgál.

### 3.2.1 Verziókezelés

A program verziókezeléséhez a Git-et használtam, amely az egyik legismertebb ilyen szoftver. A lényege, hogy a forráskódunkat nem csak a saját gépünkön tároljuk, hanem egy felhőben lévő tárhelyen is, egy saját "**repository**"-ban. Innen bárki más is letöltheti, figyelemmel kísérheti a projekt fejlesztését, illetve adott esetben másokkal is együtt tudunk dolgozni az adott projekten. A tárhelyen megtalálhatók a projekt biztonsági mentései is a projektről, így nem kell aggódnunk, hogy elveszik a munkánk.

Tárhely szolgáltatásnak a GitHub-ot választottam, mely részben ingyenes, könnyen kezelhető grafikus felületet nyújt, és akár fiókkal nem rendelkező felhasználók számára is elérhető a forráskód.

A Git használatához telepítenünk kell magát a verziókezelő programot, amely a Git hivatalos oldaláról: <https://git-scm.com/downloads> letölthető a megfelelő operációs rendszer kiválasztása után.

### 3.2.2 Letöltési lehetőségek

Github: A szakdolgozat projektjét tartalmazó repository megtalálható a következő weblapon: <https://github.com/polozgai/szakdolgozat> , ahol a zöld „Clone or download” gombra kattintva felnyíló ablakon két lehetőségünk van a letöltésre:

1. Letöltés tömörített mappaként, ha „Download ZIP” gombra kattintunk, majd egy tetszőlegesen választott mappába kitömörítjük a letöltött állományt.
2. Tárhely klónozás, vagyis a Git segítségével a projekt repository-ban lévő könyvtárat egy az egyben letölti az általunk kijelölt mappába. Ehhez a művelethez le kell tölteni az előbb említett Git verziókezelőt, valamint rendelkezni kell GitHub fiókkal is. Elsőként nyissunk egy parancssort abban a könyvtárban, ahova szeretnénk a forráskódot letölteni. Ezután gépeljük be a következőt: git clone https://github.com/polozgai/szakdolgozat.git , majd üssünk entert. A program a Github felhasználó nevünket és jelszavunkat fogja kérni, majd automatikusan végrehajtja a kiadott parancsot.

A szakdolgozathoz mellékelt CD-n is megtalálható a forráskód.

## 3.3 Projekt megnyitása

A szakdolgozat forráskódja egy Maven projektként lett létrehozva, így könnyen megtudjuk nyitni bármely az ilyen projektek megnyitását tartalmazó fejlesztői környezetben. A forráskód az Intellij IDEA nevű fejlesztői környezetben íródott, így ez az ajánlott is.

Miután megnyitottuk az Intellij IDEA-t kattintsunk a bal felső sarokban lévő „File” menüpontra, majd az itt legördülő ablakon válasszuk az „Open...” opciót. Az itt elénk táruló ablakban keressük meg a projektet, majd az „OK” gombra kattintva automatikusan betöltődik a projekt.

## 3.4. Felhasznált technológiák

A projekt részletes bemutatása előtt a következőkben a projekt létrehozásához, és működéséhez elengedhetetlen alkalmazásprogramozási interfészt, és az őt megvalósító technológiáról, valamint projekthez tartozó egyéb függőségekről lesz szó.

### 3.4.1 Maven

A Maven egy a Java programozási nyelvhez kifejlesztett csomagoló keretrendszer. Működése során feladatokat automatizál. Bevezeti a POM, azaz a Projekt Objektum Modell fogalmát. A gyakorlatban ezt a könyvtárszerkezetben található pom.xml-ben tudjuk megadni a buildelendő projekt leírását, valamint annak a függőségeit, amelyeket be is szerez. Lefordítja a forráskódot, amit így a Java virtuális gépe értelmezni tud, illetve összeállítja a .jar, .war fájlokat.

A Maven teljes körű használatához telepíteni kell azt, amihez a következő oldalon található az útmutató: <https://maven.apache.org/install.html>

### 3.4.2 Java Message Service

A Java Message Service (röviden JMS) egy Java API, amellyel üzeneteket lehet küldeni két vagy több kliens között. Az elosztott rendszerekben az üzenetküldés módszere aszinkron kommunikációt biztosít a kliensek számára, hogy egy üzenetsoron keresztül kommunikáljanak egymással, anélkül hogy a küldő oldali kliens vagy a fogadó oldali kliens bármilyen információval rendelkezne a másikról.

A JMS API kétféle modellt támogat:

* point-to-point modell
* publish-subscribe modell

A szakdolgozatban a point-to-point modell lett megvalósítva, ezért csak ez kerül bemutatásra. Erről a modellről általánosságban el lehet mondani, hogy a kommunikáció a "producer" kliens, és a "consumer" kliens közt zajlik, így ebben az esetben a "producer" ismeri a fogadót, és közvetlenül az ő üzenetsorában publikálja az üzenetét, viszont ezt az üzenetet kizárólag egy "consumer" kliens kaphatja meg. Általánosságban elmondható az is, hogy mivel az üzenetküldés egy külső komponensen zajlik, ezért nem kell a "producer"-nek futnia, amikor a "consumer" megkapja az üzenetet, illetve a "consumer"-nek sem kell futnia, amikor a "producer" publikálja az üzenetet.

### 3.4.3 Apache ActiveMQ

A Java Message Service használatához szükség van egy őt implementáló szolgálatóra. A szakdolgozat az Apache ActiveMQ nevű nyílt forráskódú üzenetbrókerét használja, melynek a függősége természetesen megtalálható a Maven által létrehozott pom.xml fájlban. Az ehhez tartozó bróker URL-je a következő: "tcp://localhost:61616" Amennyiben nem csak a forráskódban szeretnénk használni a szoftver adta szolgáltatást, de monitorozni is szeretnénk a klienseket, illetve létrehozni vagy szerkeszteni is akarjuk az üzenetsorokat, akkor lehetőségünk van letölteni az ActiveMQ legfrissebb stabil verzióját.

Telepítés lépései:

1. Keresse fel ezt a weboldalt: <http://activemq.apache.org/activemq-5158-release.html>. Itt válassza ki a számunkra megfelelő operációs rendszert, majd töltsük le a zip állományt
2. A zip állományt csomagoljuk ki egy tetszőleges mappába, majd lépjünk bele az így kapott könyvtár "bin" nevű mappájába, ahol nyissunk egy parancssort.
3. A parancssorba gépeljük be a következőt: activemq start
4. Ennek a parancsnak a hatására a kiírt információk alsó felében láthatjuk, hogy a web konzol melyik porton indult el. Nyissuk meg ezt a portot a localhost-on és a /admin eléréssel. Pl (http://localhost:8161/admin).
5. Itt be kell jelentkezni az "admin"/"admin" párossal és már lehet is használni az üzenetbróker által nyújtott szolgáltatásokat.

### 3.4.4 Spark Java

A Spark Java egy keretrendszer webes alkalmazások létrehozásához a Java 8-hoz. A szakdolgozat projektje ezt webes alkalmazást használja, hogy összekösse az algoritmus modell logikáját a grafikus felülettel, vagyis szerverként szolgál, ahol az információt továbbíthatjuk, és feldolgozhatjuk a bejövő kéréseket. A használatához természetesen szintén meg kell adni a Maven által generált pom.xml-ben. A keretrendszerhez tartozó dokumentáció, valamint oktató kód részletek megtalálhatók a Spark Java hivatalos honlapján: <http://sparkjava.com/>

### 3.4.5 JSON-Simple

A Java Message Service-ben az üzenetek megkonstruálására a TextMessage osztályt használtam melyben csak String típusú üzeneteket lehet küldeni az üzenetsorra. Az algoritmus működése során összetett üzenetek kerülnek összeállításra és értelmezésre, ezért ezeknek a könnyebb átláthatósága, az adatstruktúrák, valamint a tömbök reprezentálására is jobban alkalmasabb JSON formátumot használtam. Természetesen ezt csak úgy lehetett megoldani, hogy a kész JSON objektumunkat a küldő oldalon Stirng-é alakítjuk és így küldjük el, majd ezt a fogadó oldalán pedig átparszoljuk. A JSON-t a későbbiekben is fel tudtam használni a web oldal és az algoritmus modell rétege közti információ cserére is. A projekthez tartozó pom.xml-ben a JSON-Simple nevű függőség van megadva, ami egy JSON könyvtárat biztosít a Java nyelvhez.

### 3.4.6 Vis.js

A Vis.js egy Javascript könyvtár, melynek segítségével a weboldalon lehetőség van különböző vizualizációkat dinamikusan végrehajtani. Ezzel a könyvtárral lehetőségünk van arra, hogy a szerver felől érkező akár nagy mennyiségű adatokat megjelenítsük a weboldalunkon, illetve a későbbiekben ezt módosítsuk. Erről a könyvtárról, valamint a használatáról segítséget kaphatunk a könyvtár hivatalos oldaláról: <http://visjs.org/>

## 3.5 A program szerkezete

A Maven már a projekt megalkotásánál rendelkezésünkre bocsát egy sablon mappa szerkezetet, ezzel teszi könnyebbé az eligazodást a mappaszerkezet, valamint a bennük található csomagszerkezetek között. Az ezekben a külön mappákban található csomagszerkezetek megfelelnek az általánosan elfogadott Java fejlesztési konvencióknak, azok jól elkülöníthető részekből állnak, amely megkönnyíti az eligazodást az osztályok között.

??? ábra

### 3.5.1