
Eötvös Loránd Tudományegyetem

Informatikai Kar

Információs Rendszerek

Tanszék

# Általánosított Euler-diagramok automatikus elrendezése

*Témavezető:*

Dr. Molnár Bálint

habilitált egyetemi docens

*Szerző:*

Sarkadi-Nagy Bence

programtervező informatikus MSc

*Budapest, 2020*

Az eredeti szakdolgozati / diplomamunka témabejelentő helye.

# Tartalomjegyzék

# 1. fejezet

# Bevezetés

## 1.1. Absztrakt

A hipergráfok gyakran alkalmazott matematikai eszközök az informatika számos területén, így a szemantikus web, bioinformatika, szenzorhálózatok, adatbázisrendszerek, szociális hálózatok, gépi látás és egyéb területek számos megoldása épül rájuk. Ezen feladatok vizsgálata során különösen hasznos lehet a hipergráfok, illetve halmazrendszerek vizualizációja. Vizualizáció során egyszerre merül fel igény a mögöttes struktúra tökéletes leképezésére, a könnyű értelmezhetőségre (így esztétikai metrikákra), az általános alkalmazhatóságra, gyors futásidőre, és - ezzel összefüggésben - a megjeleníthető adathalmaz méretének maximalizálására is, továbbá gyakran felmerül a dinamikus környezet - akár a struktúra, akár Mint látható, ez egy felettébb összetett problémát eredményez, amely megoldására számos módszer született már a szakterületi irodalomban, azonban ezek gyakran szenvednek egy - vagy több - tervezési elv sérülésétől, így egyik sem terjedt el a gyakorlatban. Jelen dolgozatban különböző optimalizációs módszerek, illetve heurisztikák teljesítményét vizsgáljuk a fenti szempontok alapján, különös tekintettel az általános alkalmazhatóságra és helyes leképezésre.

## 1.2. A dolgozat felépítése

Segítendő a szövegben való tájékozódást, ebben az alfejezetben részletezem a dolgozat felépítését. A diplomamunkára vonatkozó szabályoknak megfelelően három fejezetre bomlik a szöveg. Az első (Bevezetés című) fejezet további három nagyobb részre bontható. Az első ilyenben megadom a probléma leírásához, illetve megértéséhez szükséges definíciókat és fogalmakat, majd a tágabb és szűkebb problématerület irodalmi áttekintésére kerül sor, a probléma pontos megfogalmazásával lezárva. A második elkülönülő része a bevezetésnek a megoldás során alkalmazott módszereket, illetve az azokhoz közvetlenül kapcsolódó definíciókat írja le. A bevezetés végén a különböző megoldási módszerek teljesítményét vizsgáljuk, különböző hiperparaméterek mellett.

A mérések során alkalmazott szoftver felhasználói dokumentációját a második (Felhasználói dokumentáció), míg fejlesztői dokumentációját a harmadik (Fejlesztői dokumentáció című) fejezetében találhatjuk.

**Megjegyzés.** *A dolgozatban szereplő egyes szakszavak (néha egész szakterületek) egyáltalán nem, vagy nem elég hangsúlyosan szerepelnek a magyar szakirodalomban ahhoz, hogy elterjedt fordításuk legyen. Ezekben az esetekben - további figyelmeztetés nélkül - az angol nyelvű megfelelőjüket fogom használni. Népszerű, szinonímaként használt szakkifejezéseket igyekszem per jellel elválasztva felsorolni.*

## 1.3. Probléma

### 1.3.1. Alapdefiníciók

Ahhoz, hogy megértsük a megoldandó problémát, számos fogalmat át kell tekintsünk először.

#### Gráfok

A gráf (variációi) alapvető fontosságú adattípus(ok) a modern informatikai megoldások során. Definiálásuk nem csak a dolgozat későbbi részeiben felbukkanó algoritmusok miatt fontos, hanem a - gyakran a gráfok általánosításának tekintett - hipergráfok mélyebb megértését is elősegítik.

**1. Definíció.** ***Irányítatlan gráfnak*** *nevezünk egy olyan, rendezett $G = (V, E)$ párt, ahol $V$ egy nem-üres halmaz, $E$ pedig egy olyan multihalmaz, amely a $V$ elemeiből képzett kételemű halmazokat tartalmaz. Formálisan $E \subseteq \{\{u, v\} | u, v \in V\}$. A $V$ halmazt* ***csúcshalmaznak*** *is szokás nevezni, elemeit* ***csúcsoknak****, míg $E$-t az* ***élhalmaz****, elemeit pedig* ***él*** *névvel illetjük. Az élek által tartalmazott elemeket az adott él* ***végpontjainak*** *hívjuk. Két csúcs* ***szomszédos****, ha van olyan él $G$-ben, amely őket tartalmazza, míg egy csúcs* ***izolált****, ha egyetlen élnek sem végpontja. A $G' = (V', E'), V' \subseteq V, E' \subseteq E$ a $G = (V, E)$ gráf* ***részgráfja****.*

**2. Definíció.** *Egy nem-üres halmaz, $V$, és a $V$ elmeiből képzett kételemű rendezett párokat tartalmazó $A$ multihalmazból képzett rendezett $D = (V, A)$ párt* ***irányított gráfnak*** *hívjuk. Az irányítatlan gráfok nómenklatúrája itt is érvényes, azonban az élek rendezett párjában az első elemet speciálisan* ***kiindulópontnak****, a másodikat pedig* ***végpontnak*** *is szokás nevezni. Azt mondjuk, hogy egy $(u, v) \in A$ él az $u$ csúcsnak egy* ***kimenő éle****, $v$-nek pedig egy* ***bemenő éle****.* ***Forrásnak*** *nevezzük azt a csúcsot, amelynek nincsenek bemenő élei,* ***nyelőnek*** *azt, aminek nincsenek kimenő élei.*

**3. Definíció.** *Az $e_i, e_j \in E, i \neq j$ éleket* ***párhuzamos éleknek*** *nevezzük, ha $e_i = e_j$.* ***Hurokélnek*** *egy olyan élet nevezünk, amely $\{v, v\}$ vagy $(v, v)$ formájú, azaz a két végpontja azonos.*

**4. Definíció.** *Az $v \in V$ él* ***fokszámát*** *$d$-vel jelöljük, és $d = |\{e | e \in E \wedge v \in e\}|$. A $G$ gráf* ***k-reguláris****, ha minden csúcsának fokszáma $k$. Irányított gráf esetében megkülönbeztetjük a* ***befokszámot*** *és a* ***kifokszámot****.*

**5. Definíció.** ***Egyszerű gráf*** *egy olyan irányítatlan gráf, amelyben sem párhuzamos, sem hurokélek nincsenek jelen.*

**6. Definíció.** *Irányított gráf esetén élek egy $(u_1, v_1), \ldots, (u_k, v_k)$ sorozatát* ***sétának*** *nevezzük, ha $v_i = u_{i+1}, i = 1 \ldots k - 1$. Irányított gráfok esetén analóg módon definiáljuk a fogalmat, azonban eltekintünk a csúcsok élen belüli sorrendjétől. Ha a séta semelyik két éle nem tartalmazza ugyanazt a csúcsot, akkor a sétát* ***útnak*** *mondjuk. Ha az út kezdőpontja megegyezik a végpontjával, akkor az egy* ***kör****. Két csúcs* ***távolsága*** *a köztük lévő legrövidebb útban szereplő élek száma, ha ilyen nincs, akkor végtelen. Egy $v$ csúcs* ***elérhető*** *az $s$ csúcsból, ha a távolsága nem végtelen tőle.*

***Összefüggőnek*** *mondott egy gráf (vagy részgráf), ha abban minden csúcs elérhető mindegyik másikból.*

**7. Definíció.** *$K_n$-nel jelöljük az $n$ csúcsú egyszerű gráfot, amelyben minden csúcspár között fut él, az ilyen gráfok neve* ***teljes gráf****. Mikor csak egy részgráfra igaz ez a tulajdonság, akkor azt* ***teljes részgráfnak*** *vagy* ***klikknek*** *mondjuk.*

**8. Definíció.** ***Topologikus sorrendként*** *ismert a $D = (V, A)$ irányított gráf csúcsainak egy olyan sorrendje, ahol kisebb sorszámú csúcsból csak nagyobb sorszámúba megy él. A topologikus sorrend megléte ekivivalens azzal, hogy az adott gráfban nincsen kör, így az ilyet* ***körmentes gráfnak*** *nevezzük.*

**9. Definíció.** *Egy $G = (V, E)$ irányítatlan gráf* ***line graphja*** *alatt az $L(G) = (E, \{\{e_i, e_j\} | v \in V \wedge v \in e_i \wedge v \in e_j, i \neq j\})$ gráfot értjük.*

### Halmazrendszerek és hipergráfok

Mint az absztraktban is említettem, halmazalapú adatreprezentációkkal, így hipergráfokkal és halmazrendszerekkel az informatika számos területén találkozhatunk. Az egyes problématerületek - sőt gyakran egy problématerületet vizsgáló különböző szakcikkek - azonban egymással konkuráló definíciókat alkalmaznak. Gyakran a halmazrendszerek szinonímájának tekintik a fogalmat, míg például - a szakterület egyik alapművének számító - Hypergraphs c. [1] kötet mind az általunk használt - mindjárt megismertetett - definícióhoz, mind a halmazrendszer alapúhoz képest megszorításokat vezet be.

**10. Definíció.** *A $H$ halmaz* ***hatványhalmaza*** *$\mathcal{P}(H) = \{x | x \subseteq H\}$, azaz a $H$ halmaz összes részhalmazainak halmaza.*

**11. Definíció.** *A $H$ halmaz fölötti $\mathcal{F}$* ***halmazrendszert*** *$\mathcal{F} \subseteq \mathcal{P}(H)$-ként definiáljuk.*

**12. Definíció.** *Ebben a dolgozatban* ***irányítatlan hipergráf*** *vagy egyszerűen hipergráf alatt egy olyan, rendezett $H = (V, E)$ párt értünk, ahol $V$ a* ***(hiper)csúcsok*** *nem-üres halmaza, míg $E$, a* ***hiperélek*** *halmaza, egy olyan multihalmaz, amelynek az elemei $\mathcal{P}(V) \setminus \emptyset$-ből kerülnek ki.*

**Megjegyzés.** *A hipergráf, úgy is ismertek, mint a gráfok általánosításai, ahol minden hiperél pontosan 2 elemet tartalmaz, azaz 2-reguláris. Egyrészről ez jól láthatóan*

*függ a választott gráf-, és hipergráfdefinícióktól. Az itt használt definíciók alapján hipergráfok nem tartalmazhatnak hurokéleket, viszont párhuzamos éleket igen, így nem tökéletes általánosításai az irányítatlan gráfoknak, viszont irányítatlan egyszerű gráfoknak már igen.*

**Megjegyzés.** *Egy másik felmerülő kérdés a hipergráfok kapcsán a hiperutak, más szóval a tranzitivitás fogalma a hiperélek között. A szakirodalomban erre is több, azonban jobban elkülönülő definíció létezik. Az itt bemutatott eredmények nem építenek a tranzitív relációra, így tetszőleges definíció tételezhető fel.*

### Gráf- és hipergráf-adatszerkezetek

Gráfok gépi kezelésére köztes gráfreprezentációkra, gráfadatszerkezetekre van szükség. Leggyakrabban az adjacencia mátrix, az incidencia mátrix, az éllista és a ritka reprezentációk általános osztálya használatos. Mivel a megoldások során csak az első kettőt alkalmazzuk, ezért a többit itt nem is definiáljuk.

**13. Definíció.** *A $G = (V, E)$ gráf **adjacencia mátrixa** alatt azt a $|V| \times |V|$ méretű $A_G$ mátrixot értjük, amelyben $a_i j = 1$, ha $\{v_i, v_j\} \in E$ és 0 egyébként.*

**14. Definíció.** *A $G = (V, E)$ gráf **incidencia mátrixa** alatt azt a $|V| \times |E|$ méretű $I_G$ mátrixot értjük, amelyben $i_j k = 1$, ha $v_j \in e_k$ és 0 egyébként.*

**15. Definíció.** *A $H = (V, E)$ hipergráf **incidencia mátrixát** ugyanúgy definiáljuk, ahogy a gráfokon definált megfelelőjét.*

**Megjegyzés.** *Hipergráfok esetében az adjacencia mátrix nem értelmezett, azonban a szakirodalomban gyakran használnak gráfokat köztes reprezentációként, így különösen a bipartite incidence structure, a multimodal/multilayer/multiplex/multidimensional graph/network és a line/intersection graph fogalmaknak érdemes utánanéznie az ez iránt érdeklődő olvasónak.*

**16. Definíció.** *A $H = (V, E)$ hipergráf **line/intersection graphja** az $L(H) = (E, \{\{e_i, e_j\} | v \in V \wedge e_i \cap e_j \neq \emptyset, i \neq j\})$ képlettel kapott egyszerű gráfot fedi. Jól látható, hogy ez általánosíta a gráfok esetében alkalmazott definíciónak.*

**17. Definíció.** *Egy irányítatlan $H = (V_H, E_H)$ hipergráf **(szuper)duálisa** az a $G = (V_G, E_G)$ egyszerű gráf, amelyben $V_G = \{X | X \subseteq E_H \wedge \exists v \in V_H : ((\forall e \in X :$*

$v \in e) \wedge (\forall e \notin X : v \notin e)\}$ *és* $E_G = \{\{X, Y\} | X, Y \subseteq E_H, X \neq Y \wedge \exists v \in V_H : (v \in X \wedge v \in Y)\}$

**Megjegyzés.** *Figyeljük meg, hogy a szuperduálisban szereplő csúcsok száma exponenciális az eredeti csúcshalmaz tekintetében!*

### 1.3.2. Gráfok ábrázolása

Mivel a gráfok gyakran használt, és könnyen konceptualizálható modellek, ezért a számítógépes vizsgálat mellett gyakran előnyös a felhasználó számára vizualizálni őket. Gráfok reprezentálhatók halmazokként, numerikusan (erre később látni fogunk példát), azonban ember-gép közreműködések során a leggyakrabban olyan képként szokás ábrázolni őket, melyeken a csúcsok körökként, az őket összekötő élek pedig egyenes (egyes esetekben akár görbe) vonalakként.

Gyakorlatban különösen fontos tulajdonságnak bizonyult - a könnyű értelmezhetőség szempontjából -, hogy egy adott gráf kirajzolható-e anélkül, hogy bármely két éle metszené egymást. A gráfelmélet egy ismert tétele, hogy nem minden gráf rajzolható síkba. A **Fáry-Wagner** tétel továbbá kimondja, hogy minden olyan gráf, ami görbe vonalak használatával síkbarajzolható, az egyenes vonalak esetén is síkbarajzolható marad.

### 1.3.3. Hipergráfok és halmazrendszerek ábrázolása

Gráfokhoz hasonlóan hipergráfok (és/vagy halmazrendszerek) esetén is hasznos a felhasználó által értelmezhető ábrázolásuk, azonban - a gráfoktól eltérő módon - a vizuális reprezentációjuk már egyáltalán nem egységes. A szakirodalom számos különböző módszert tart nyilván, melyeket Alsallakh és társai foglaltak össze[2]. Az említett cikk számos különböző ábra kategóriát különböztet meg, én ezek közül kiemelném, hogy léteznek régiókon, vonalakon, mátrixokon és aggregáción alapuló, illetve hibrid módszerek. Az egyes kategóriák önmagukban is több ábrázolási módszert fednek, melyeket jellemzően egyenként is szakcikkek sokasága fed, így az áttekintésükre itt nincs módunk, azonban jól mutatja a kutatási terület mélységét.

Ebben a dolgozatban egy specifikus, régióalapú ábrázolási mód, az Euler-diagram vizsgálatát tűztem ki célul. Az Euler-diagram kirajzolását célzó algoritmusok az úgy-

nevezett Euler Diagram Generation Problem (EDGP) megoldásai. Ahogy a neve is mutatja, a szóban forgó ábrázolási módot még maga Leonhard Euler vezette be a XVIII. században, azonban mindmáig előszeretettel használatos. Ahogy Baron írja[3], Euler mindössze példákon mutatta be, illetve alkalmazta módszerét, nem definálta konkrétan. Ezek alapján - a hipergráfokhoz mintájára - Euler-diagramokra is többféle definíció létezik, melyek mind megyeznek abban, hogy az egyes halmazok/hiperélek zárt görbékkel reprezentáltak, melyek metszetei közös halmazelemeket feltételeznek, míg diszjunkt esetben azok hiányát. Egyes definíciók az alaphalmaz elemeit/hipercsúcsokat is elhelyezik az ábrán, így létrejöhetnek olyan metszetek is a vizualizáció során, melyek valójából üresek, de az értelmezés során ez mégsem okoz gondot. Gyakori kérdés, hogy a zárt görbék körök, ellipszisek vagy tetszőleges formájúak lehetnek-e, hogy szükségszerűen konvexek-e, illetve az egyes halmazok több, azonos címkével ellátott görbével is reprezentálhatók-e. A következőkben definiálom, hogy ebben a dolgozatban milyen értelmezést használok, azonban - az előzőeknek megfelelően - egyéb források ettől eltérhetnek.

**18. Definíció.** *A továbbiakban a $H = (V, E)$ hipergráfot reprezentáló* ***egyszerű Euler-diagram*** *alatt egy olyan ábrát értünk, amelyben minden $e \in E$ halmaz egyetlen zárt görbére képződik le, mely azokat, és csak azokat a hipercsúcsokat tartalmazza, melyek az adott e hiperélnek is elemei. Azon hipercsúcsok, melyek egyetlen hiperélben sem szerepelnek, az összes görbén kívül kell megjelenjenek. Az egyes görbék címkével (esetünkben színekkel) rendelkeznek.*

**19. Definíció.** ***Általánosított Euler-diagramként*** *vagy csak* ***Euler-diagramként*** *fogom nevezni azt az egyszerű Euler-diagramot, mely egy hiperélt több, azonos címkével ellátott zárt görbére is leképezhet.*

### 1.3.4. Euler-diagramok vizualizációja

#### Síkbarajzolhatóság

Gráfok esetében láthattuk, hogy felmerül a síkbarajzolhatóság kérdése. Amennyiben egyszer Euler-diagramokkal, így zárt, a végpontokat (jelen esetben hipercsúcsokat) tartalmazó görbékkel reprezentáljuk az éleket, akkor rögtön láthatjuk, hogy ezeknek tartalmazniuk kell legalább egy görbét is, amely közvetlenül összekö-

ti a végpontokat. Ebből már észrevehetjük, hogy nem minden hipergráf rajzolható síkba egyszerű Euler-diagramokkal.

Verroust és Viaud bebizonyította[4], hogy az általuk használt Euler-diagram definíció 8 halmazig megőrzi a síkbarajzolhatósági tulajdonságot. Simonetto és Auber egy másik struktúrát vizsgált[5], ami megfelel az itt általánosított Euler-diagramként definiált fogalomnak (ők Euler representationnek nevezik), melyről levezették, hogy alkalmas tetszőleges hipergráf síkbarajzolására. A szuperduális felhasználásával pontos leírása is adható annak, hogy mikor nem rajzolható ki egy hipergráf egyszerű Euler-diagramok használatával[4, 5]. (Megjegyzendő, hogy a Sunibetto cikk intersection graphnak nevezi a szuperduálist, miközben az egy másik struktúrát jelöl, de a leírásból, illetve a példákból levezethető, hogy valójából felcserélték a kettőt.)

### Esztétikai mértékek

Természetesen egy adott diagram síkbarajzolása nem feltétlenül szükséges ahhoz, hogy az ábra helyesen képezze le a mögöttes struktúrát (halmazrendszert vagy hipergráfot), azonban minenképpen megkönnyíti az emberi értelmezést. Egy adott ábra ilyen tulajdonságait esztétikai tulajdonságoknak nevezzük, ha pedig számszerűsíthetők (és adott rajtuk egy teljes rendezés), akkor esztétikai metrikáknak hívjuk. Esztétikai metrikák definiálhatók magukon a görbéken, a görbék metszetein, a csúcsok eloszlásán, az ábra színezésén, továbbá - gyakorlatilag - az ábrán megjelenő bármely aspektus fölött [6, 7, 8].

Az EDGP során felmerülő két leggyakorabban vizsgált[9, 6, 7, 10] esztétikai tulajdonság, az úgynevezett well-formed és well-matched tulajdonságok.

**20. Definíció.** *Egy adott Euler-diagram esetén* ***kontúr/contour*** *névvel illetjük az egy címkéhez (vagy hiperélhez/halmazhoz) tartozó különböző zárt görbék összességét.* ***Minimális régiónak/minimal regionnek*** *hívjuk a görbék egymás által létrehozott legkisebb síkpartícióit, míg* ***alaprégió/basic region*** *alatt azon minimális régiók halmazát értjük, melyek ugyanazon görbék részei. Egy* ***zóna/zone*** *az alaprégiók egy olyan halmaza, amelyek azonos címkével rendelkeznek.*

A well-formed tulajdonság hat kritériumból tevődik össze, bár néha csak az első ötöt használják:

1. Minden görbék egyszerű, azaz nem metszi önmagát - WFC1
2. Nincs két görbe, amelynek közös határolószakasza van. (Az elfajuló, pontbeli találkozást nem vesszük hozzá - WFC2
3. Nincs olyan pont, ahol három görbe érintkezik - WFC3
4. Ha két görbe érintkezik, akkor metszik egymást - WFC4
5. Minden zóna összefüggő, azaz egyetlen minimális régióból áll - WFC5
6. Nem rendelkezik két görbe azonos címkével - WFC6

A well-matched tulajdonság az alábbi négy tulajdonság meglétét fedi:

1. Egy Euler-diagram well-matched a zónák szintjén, ha nem tartalmaz üres zónákat. - WMP1
2. Egy Euler-diagram well-matched a görbék szintjén, ha a halmazok közti részhalmaz, metszet és diszjunkt relációk megfelelnek az adott halmazokat reprezentáló görbék tartalmazás, átfedés és diszjunkt tulajdonságának. - WMP2
3. Egy Euler-diagram well-matched a minimális régiók szintjén, ha well-matched a zónák szintjén, és csak összefüggő zónákat tartalmaz. - WMP3
4. Egy Euler-diagram well-matched a kontúrok szintjén, ha a halmazok közti részhalmaz, metszet és diszjunkt relációk megfelelnek az adott halmazokat reprezentáló kontúrok tartalmazás, átfedés és diszjunkt tulajdonságának. - WMP4

Szintúgy széles körben vizsgált tulajdonság az area-proportionality, avagy méretarányosság[11, 12, 13, 14], amely azt mondja ki, hogy minden régió (bizonyos definíciók szerint az univerzumot reprezentáló kivételével) mérete úgy aránylik az ilyenek összegéhez, mint az $\omega$ súlyfüggvényük azok összegéhez. Leggyakrabban a zárt görbe által tartalmazott elemek számát alkalmazzuk súlyfüggvényként. Hibamértékek segítségével könnyen metrika is előállítható a tulajdonságból.

Annak ellenére, hogy több esztétikai metrika és tulajdonság is széles körben alkalmazott, nagyon kevés empirikus tapasztalatunk van arról, hogy ezek ténylegesen befolyásolják-e egy ábra értelmezhetőségét. Fish és társai kis mintán vizsgálták[15] a well-formed tulajdonságnak az ábrák megértésére vonatkozó hatását. Az ő eredményeik alapján a WFC2 megsértése akár segítheti is egy ábra értelmezését, míg a

WFC1 és WFC4 egyidejű, illetve a WFC3 vagy WFC4 önálló megsértése is rontja azt. Blake és társai[10] arra az eredményre jutottak, hogy az egyes görbék orientációja nincs hatással az emberi percepciójukra. Blake-ék egy másik cikkükben[16] a szimmetrikus alakzatokat azonosították a legkönnyebben megérthetőként, így különösen a kör alakú reprezentációt javasolják.

### Euler-diagramok generálása

Még úgy is, hogy az Euler-diagramok mindössze részterületét képezik a halmazábrázolási módszereknek, a szakirodalomban rengeteg különböző eljárás található, melyek gyakran nem is tekinthetők ugyanannak a szűken vett probléma megoldásának. Az alkalmazott Euler-diagram definíció, a vizsgált probléma mérete, az alkalmazott esztétikai tulajdonságok és metrikák, illetve ezek erős vagy gyenge megkövetelése mind-mind új variánsait hozzák létre a - összefoglaló néven EDGP-nek nevezett - problémának.

A halmazábrázolási terület legátfogóbb összegzését Alsallakh és társai adták[2], melyben számos EDGP megoldás összehasonlítását is adták (lásd az első táblázatot a cikkükben). Itt számos szempont alapján kategorizálják az egyes módszereket. Elsősorban megkülönböztetik a tetszőleges relációk ábrázolására alkalmas, illetve az ebben a tekintetben korlátozott megoldásokat. Ettől nem függetlenül megadják, hogy milyen alakzatokkal reprezentál egy halmazt az adott módszer (kör, poligon vagy ellipszis), a cikkből azonban sajnos kimaradt, hogy ismert Bézier-görbe alapú megoldás is[8]. Másik szempontként hozzák fel a teljesített esztétikai tulajdonságokat, mint a well-formed, well-matched, area-proportional, szimmetrikus görbe tulajdonságokat és vizsgálják, hogy létrejönnek-e üres minimális régiók (az univerzumon kívül). Az általuk adott táblázat alapján továbbá azt tételezhetjük fel, hogy az egyes módszerek vagy három, vagy tetszőleges számú halmazra alkalmazhatók. Megfigyelhető, hogy ezutóbbi az alapján válik el, hogy az Euler-diagramok egy speciális esetét, a Venn-diagramokat vizsgálja-e egy adott cikk vagy az általános problémát. Amiről ezek alapján nem kapunk képet, hogy egyes módszerek csak 8 halmazig alkalmazhatók[4], mivel ezek fölött már ismertek olyan példák[5, 17, 4], amelyek nem síkbarajzolhatók egyes Euler-diagram definíciók szerint. (A cikkben nem említett, de hasznos kiemelni, hogy egyes algoritmusok csak már meglévő diagramok esztétikai

javítását szolgálják[6] vagy emberi beavatkozást igényelnek[18].)

A dolgozat későbbi részeiben legfontosabbnak Flower, Rodgers és Mutton munkájára[8] fog bizonyulni, akik sztochasztikus optimalizációs módszereket, illetve metaheurisztikákat alkalmaztak Bézier-görbékkel reprezentált Euler-diagramokra, és akikkel részben hasonló megközelítést választottunk. Érdemes még megemlíteni Stapleton és társainak munkáját[17], amelyben induktív módon generálnak well-formed euler diagramokat, mikor ez lehetséges, a többi esetben pedig a well-formed kritériumok megsértésével, a görbék önmetszésével érik el, hogy továbbra is szemantikailag helyes ábrát generáljanak. Különösen érdemes megfigyelni, hogy az általánosság ilyen szintű eléréséhez a duális gráf egy módosított verzióját használják, ami exponenciális futásidőt eredményez.

### 1.3.5. Problémaleírás

Láthattuk, hogy az EDGP egy összetett probléma, amely magában foglalja a konkrét ábrázolási mód (Euler-diagram definíció), a vizsgált esztétikai metrikák, az elfogadható futásidő és a megoldható problémaméret meghatározását is. Különösen fontos kitérni arra is, hogy egy adott probléma vizsgálata során nem feltétlenül egyféle szempont szeretnénk ábrázolási módot választani. Elképzelhető, hogy ugyanúgy szeretnénk az előforduló klasztereket vizsgálni, mint a leghosszabb utakat, melyek másféle elrendezést tételeznek fel.

Az általam kitűzött célt az egyetemen folyó egyik kutatás igényei szerint tűztem ki, ahol adatbázisrendszerek redundanciáját csökkentjük hipergráfmodellek segítségével. Az itt folyó napi munka során egy olyan eszközre támadt szükség, amely - a jelenleg elérhető programokkal szemben - képes több tucat éllel és akár több száz csúcsal bíró hipergráfot többféle esztétikai metrika szerint is kirajzolni, akár hosszabb futási idő (órák) és/vagy egyszeri, kifejezetten hosszú (napok, hetek) tanulási idő után. Különösképp megnehezíti a feladatot, hogy Alsallakh és társai - a halmazábrázolási módszereket áttekintő cikkükben[2] - az Euler-diagramokat mindössze 10-20 halmazig tartják alkalmazhatónak. A tématerület bonyolultságának és mélységének megfelelően a dolgozat különböző módszerek vizsgálatáról szól, a végső eszközt még nem hivatott létrehozni.

Mint láthattuk, az általános megoldhatóság érdekében emberi beavatkozás, korlát nélküli paramétertér (például Bézier görbék kontrollpontjai[8]) vagy exponenciális futásidő[17] lehet szükséges. Mivel tetszőleges esztétikai metrika fölött az optimalizáció, így a legjobb Euler-diagram megtalálása is NP-nehéz, ezért ez egyáltalán nem meglepő. A megoldásom alapjaként választott Flower cikkel[8] szemben, az általam használt, bonyolultabb problématér (mind a hipergráf méretében, egy adott halmazt reprezentáló zárt görbék számában és ebből kifolyólag a költségfüggvényként alkalmazott heurisztikák nem-folytonos jellegében) felveti a modell egyszerűsítésének igényét, illetve újabb heurisztikák kidolgozásának szükségességét is.

## 1.4. Megoldás

### 1.4.1. Optimalizáció

A matematikai optimalizáció célja egy adott $f : X \to \mathbb{R}$ valós értékű függvény globális minimum- vagy maximumhelyének megtalálása, ahol X tetszőleges halmaz lehet. Mivel az $f$ függvény negáltja segítségével maximalizációs problémák minimalizáliós problémákká vezethetők vissza (vagy fordítva), ezért a kettő feladatot azonosnak tekintjük. A továbbiakban - mikor nem jelzem az ellenkezőjét - minimalizálási problémát tételezek fel.

**21. Definíció.** *Az $f$ függvényt számos módon nevezik; minimalizálási problémák esetén* ***költség-/veszteségfüggvényként*** *vagy* ***objektívfüggvényként****, míg maximalizálási problémák esetében* ***utility*** *vagy* ***fitness functionként*** *ismerjük. Egyes szakterületeken (mint fizikában) egyéb nevek is ismertek. Ha az $f$ függvény a $g_i, i \in \mathbb{N}_0$ függvények átlagaként áll elő, akkor költségfüggvénynek hívjuk, a $g_i$ függvényeket pedig veszteségfüggvényeknek.*

**22. Definíció.** *Egy optimalizálási algoritmus paramétereit* ***hiperparamétereknek*** *nevezzük.*

Általánosságban beszélve az optimalizálási probléma NP-nehéz, azonban a költségfüggvényre tett megszorításokkal ez feloldható. A kidolgozott algoritmusok ezért különösen fontos, hogy milyen megszorítások mellett operálnak.

### Gradiensalapú optimalizáció

Gradiens-, illetve deriváltalapú algoritmusok vagy a deriváltfüggvényt (Hesse-mátrixot) vagy a pontbeli (parciális) deriváltat alkalmazzák. Bár szigorúan véve az első- és másodikderivált-próba is ide tartozik, a gyakorlatban a vizsgált függvény pontos képlete jellemzően nem ismert, illetve a paraméterek és a lokális szélsőértékek nagy száma is ellehetetleníti ezt a féle megoldást. Ezzel szemben iteratív mószereket szokás használni, melyek egy megadott - általában véletlenszerű - kiindulópontból egy lokális minimumponthoz konvergálnak. Amennyiben a függvény konvex vagy a kiindulópont kellőképpen közel volt a globális minimumhoz, akkor a kapott lokális minimumhely globálisan is az lesz, azonban ez általában nem garantált. Az előzőeknek megfelelően sokszor elvárt tulajdonság a folytonos deriválhatóság is.

A gyakorlatban használt legtöbb iteratív algoritmus a gradiens leszálláson alapszik. Gradiens leszállás során egy tetszőleges $\theta_0$ kiindulópontot választunk, majd a $\theta_{i+1} = \theta_i - \alpha * \nabla f(\theta_i)$ szabály alapján kiválasztjuk a következő vizsgálandó pontot, ahol $\alpha$ egy hiperparaméter. Az iteráció addig folytatódik, míg a megállási feltétel (például felső korlát az iterációk számára, a lépésköz nagyságára vagy a gradiensre egy normájára) nem teljesül. Fontos tulajdonsága ennek az algoritmusnak, hogy túl nagynak megválasztva $\alpha$-t átugorhatunk minimumhelyeket, míg túl kicsire állításával a keresési időt növeljük meg. Számos módszer született ennek a hiányosságnak az áthidalására, így egyes variációk az iterációk számának növekedésének függvényében csökkentik $\alpha$-t vagy eltérő kiindulópontokból indítják újra a keresést. Bizonyos feltételek mellett az algoritmus garantáltan egy lokális minimumhelyhez konvergál[19].

A gradiensalapú módszerekkel rokon jegyeket mutat a - nem feltétlenül differenciálható, de konvex függvények esetében a alkalmazható - szubgradiensmódszer, illetve az általánosabban használható, lokális keresésen alapuló hegymászó algoritmus (hill climbing).

### Sztochasztikus algoritmusok

Összefoglaló néven **sztochasztikus algoritmus** névvel illetjük azokat a módszereket, amelyek futásuk során valószínűségi változókat alkalmaznak. Ez a megközelítés gyakran használt, mikor a költségfüggvény nem alkalmas direkt optimalizációra, viszont megelégszünk közelítő megoldásokkal is. Szintúgy előnyös, mikor a keresési

térben számos, globális minimumértékhez közel álló pont létezik. A gradiens leszállás során látott véletlen kezdőpont kiválasztását inputnak tekintjük, így az algoritmus alapverzióját nem tekintjük sztochasztikusnak, viszont - mint számos egyéb determinisztikus (nem sztochasztikus) algoritmusnak - vannak sztochasztikus variánsai. Ennek megfelelően fontos kiemelni, hogy az egyes optimalizációs módszereknek általában több variánsa is létezhet, melyek esetenként összemossák a kategóriákat. Ennek a szekciónak a további részében át fogjuk tekinteni azokat az algoritmusokat, amelyeknek szükséges az ismerete a továbbiakban, azonban néhol pont az itt bemutatott alapverziótól való eltérés lesz különösen érdekes.

A legismertebb ilyen variáns az úgynevezett **sztochasztikus gradiens leszállás** (SGD). Mikor az $f$ költségfüggvény több $f_i$ veszteségfüggvény átlagaként jön létre, azaz $f(x) = 1/n * \sum_{i=1}^{n} f_i(x)$ (például gépi tanulás során több mintaelemen alkalmazva ugyanazt a veszteségfüggvényt), akkor előfordul, hogy a gradiens leszállás során a mai számítógépek kapacitásához képest túl sok parciális deriváltat kéne kiértékelni. Az SGD ez úgy kerüli el, hogy a veszteségfüggvényeket egyesével értékeli ki, mindegyik kiértékelés után modósítva a paramétereket. Láthatóan ez nem ekvivalens a teljes költségfüggvény gradiensének használatával (csak közelíti azt), továbbá nagyban függ a kiértékelés sorrendjétől is, ezért a veszteségfüggvények sorrendje a kiértékelés során randomizált. Néha szintúgy SGD-nek nevezik a mini-batch gradient descent módszert, ami annyiban tér el az SGD-től, hogy a paraméterek módosítása során egyszerre $k$ darab veszteségfüggvény átlagát használjuk, ahol $k$ egy hiperparaméter.

Természetben lejátszódó folyamatok számos esetben optimalizációs módszerként is tekinthetők. Az ilyen adaptált optimalizációs technikákat összefoglaló néven **biologiailag inspirált** algoritmusoknak nevezzük. A természeti rendszerek komplexitásából fakadó modellezési bizonytalanság feloldását sokszor a sztochaszticitás bevezetésével érjük el, így - gyakorlatban - a legtöbb biológiailag inspirált optimalizációs algoritmus egyben sztochasztikus algoritmus is.

Egy ilyen, természet ihlette algoritmus, a természetes szelekción alapuló **genetikus algoritmus** (GA)[20, 21]. A módszer mögötti megfontolás, hogy egy populáció életciklusa a új egyedek születéséből/halálából, az egyedek szaporodásából (így genetikai keveredéséből), illetve mutációkból tevődik össze, ahol a párosodás a gének

előnyös jellegének $f$ függvénye. Az algoritmus célja, hogy ezt az $f$ fitness függvényt *maximalizálja*, az életképes egyedek kombinációjával, illetve kis, véletlenszerű módosításokkal. Az algoritmus pszeudokódja alább látható, azonban nem térünk ki minden részletére.

**1. Algoritmus** Genetikus algoritmus

**Function** GA($populationSize, parentNumber, mutationRate$)

1: $t = 0$
2: $population_t$ = generateFeasibleSolutions($populationSize$)
3: $fitnessValues$ = evaluateFitness($population_t$)
4: $bestFitness, bestPosition = updateBest(fitnessValues, \infty, population_t, bestPosition)$

5: **while** megállási feltétel nem teljesült **do**
6: $parents$ = selectFitnessProportionally($population_t, fitnessValues, parentNumber$)

7: $children$ = recombine($parents$)
8: $population_t$ = deleteLast($population_t, fitnessValues, |children|$)
9: $population_{t+1} = population_t \cup children$
10: $population_{t+1}$ = mutate($population_{t+1}, mutationRate$)
11: $fitnessValues$ = evaluateFitness($population_{t+1}$)
12: $bestFitness = updateBest(fitnessValues, bestFitness, population_{t+1}, bestPosition)$

13: $t = t + 1$
14: **end while**
15: **return** $bestFitness, bestPosition$

Ahol $generateFeasibleSolutions(n)$ $n$ darab megengedett megoldást generál, $evaluateFitness(population)$ kiértékeli az egyes egyedek fitness értékeit, $updateBest(newValues, oldBest, newPositions, oldPosition)$ az új értékek és pozíciók, illetve a korábbi legjobbak segítségével kiválasztja a legjobbat (maximumot és maximumhelyet), $selectFitnessProportionally(population, fitnessValues, num)$ a fitness értékek által implikált valószínűségi eloszlás szerint kiválaszt $num$ da-

rab egyedet, $recombine(population)$ két elemenként egy új egyedet hoz létre, amely a szülei génállományán alapszik, $deleteLast(population, fitnessValues, num)$ törli a $num$ darab legrosszabb fitness értékkel bíró egyedet a populációból, $mutate(population, mutationRate)$ pedig a $mutationRate$ arányában megváltoztatja az egyedek génállományát. A megállási feltétel gyakran iterációs felső korlát vagy alsó korlát a legjobb fitness értékre.

A **particle swarm optimization** (PSO) egy másik biológiailag inspirált algoritmus, amely analóg módon működik egyes rajként együtt dolgozó állat- és rovarfajokkal[21, 22, 23]. A módszer alapelve, hogy különböző, részecskéknek nevezett, entitások pozícióiban értékeljük ki az $f$ költségfüggvényt. A részecskék haladási iránnyal és sebességgel rendelkeznek, amelyet minden iterációban úgy frissítünk, hogy - véletlenszerű mértékben - figyelembe vesszük magát az irányt/sebességet és a részecske által, illetve a globálisan talált eddigi legjobb pozíciót/értéket. Ahogy látható algoritmus hiperparméterek segítségével adja meg a részecskék számát, illetve a módosítási szabály egyes tagjainak súlyát.

**2. Algoritmus** Particle swarm initialization

**Function** InitializePSO($S, lowerBounds, upperBounds$)

1: $bestPosition_{global} = \infty$
2: **for** $i = 1 \dots S$ **do**
3: $\quad particlePosition_i \sim U(lowerBounds, upperBounds)$
4: $\quad bestPosition_i = particlePosition_i$
5: $\quad velocityRange = |upperBounds - lowerBounds|$
6: $\quad velocity_i \sim U(-velocityRange, velocityRange)$
7: $\quad$ **if** $f(bestPosition_i) < f(bestPosition_{global})$ **then**
8: $\quad\quad bestPosition_{global} = bestPosition_i$
9: $\quad$ **end if**
10: **end for**
11: **return** $bestPosition_{global}, particlePosition_{1\dots S}, bestPosition_{1\dots S}, velocity_{1\dots S}$

**3. Algoritmus** Particle swarm optimization

**Function** PSO$(S, w, c_1, c_2, lowerBounds, upperBounds)$

1: $bestPosition_{global}, particlePosition_{1...S}, bestPosition_{1...S}, velocity_{1...S} = InitializePSO(S, lowerBounds, upperBounds)$
2: **while** megállási feltétel nem teljesült **do**
3:  **for** $i = 1 \dots S$ **do**
4:  $random_1, random_2 \sim U([0,1]^{dim(lowerBounds)})$
5:  $velocity_i = w * velocity_i + c_1 * random_1 * (bestPosition_i - particlePosition_i) + c_2 * random_2 * (bestPosition_{global} - particlePosition_i)$
6:  $particlePosition_i = particlePosition_i + velocity_i$
7:  **if** $f(particlePosition_i) < f(bestPosition_i)$ **then**
8:  $bestPosition_i = particlePosition_i$
9:  **if** $f(bestPosition_i) < f(bestPosition_{global})$ **then**
10:  $bestPosition_{global} = bestPosition_i$
11:  **end if**
12:  **end if**
13:  **end for**
14: **end while**
15: **return** $f(bestPosition_{global}), bestPosition_{global}$

#### Egyéb optimalizációs módszerek

Természetesen az áttekintett kategóriákat nem merítettük ki, számos további algoritmus és variáns áttekintésére a dolgozat keretei között nincsen lehetőségem, illetve számos további megszorítás és függvényosztály is ismert, amelyekre létezik optimalizációs algoritmus, mint például lineáris egyenlőtlenségrendszerek vagy diszkrét halmazok esetén.

### 1.4.2. Függvényapproximáció

A numerikus analízis - így például az optimalizáció - számos területén merül fel a (jellemzően valós értékű) matematikai függvények használatának az igénye. Mikor a vizsgált függvény túl komplex, nem mindenhol kiértékelhető, esetleg nem rendelke-

zik az elvárt tulajdonságokkal, akkor függvények egy meghatározott részhalmazából kiválaszthatunk egy rá legjobban illeszkedőt. Ezt az eljárást - mint az ezt vizsgáló szakterületet - **függvényapproximációnak** nevezzük. Mikor meghatározott pontokban várjuk csak el a legjobb illeszkedést, akkor görbék illesztéséről beszélünk.

Legyen $f = (f_1 \dots f_n)^\top, w \in \mathcal{R}^n$, ahol $f_i, i = 1 \dots n$ tetszőleges függvény lehet. Ekkor az $f(x, w)$ **linear function approximatornek** nevezzük, ha az lineáris súlyok $w$ vektorában (bár nem feltétlenül az az $x$ inputban), azaz $f(x, w) = w_1 * f_1(x) + \ldots + w_n * f_n(x)$. Mikor az $f$ függvény nem teljesíti a linearitási feltételt, akkor **nonlinear function approximatorről** beszélünk.

### 1.4.3. Mesterséges neurális hálók

Hasonlóan a biológiailag inspirált algoritmusokhoz, a **mesterséges neurális hálók** (artificial neural network - ANN) olyan számítási rendszerek, amelyek biológiai folyamatokat, speciálisan az emberi - illetve állati - agyban működő neuronokat, és azok működését modellezik. Egy ANN legkönnyebben irányított gráfként képzelhető el, amelyben **(mesterséges) neuronok** alkotják a valós számokkal címkézett csúcsokat, míg a köztük létező, súllyal rendelkező kapcsolatok (szinapszisok) az éleket. A csúcsok címkéjét **biasnek** hívjuk, az élekét **weightnek**. A weight és a bias értékek közösen adják ki a neurális hálózat paramétereit. Az így kapott gráf számítási rendszerként fogható fel, amennyiben az egyes csúcsok műveleteket reprezentálnak. A forráscsúcsoknak közvetlenül beadhatók a rendszer bemenetei, míg az összes többi csúcs egy újabb értéket számít ki, melyek a nyelő csúcsokban értelmezhetők végeredményként. A források kivételével az egyes csúcsok az $x_v = \sigma(w_n \cdot x_n + b_v)$ képlet alapján számítják ki az értéküket, ahol $x_v$ a $v$ csúcs új értéke (nem címkéje), $w_n$ a bemenő élek súlyainak (weight) vektora, $x_n$ ezen élek kiindulópontjainak értéke, $b_v$ a $v$ csúcs címkéje/biase, a $\sigma$ függvény pedig egy - jellemzően nem-lineáris - aktiválási függvény.

#### Architektúra

A számítási gráf csúcsait általában a forrásoktól való távolságuk alapján particionálni szokás, ahol strukturálisan az egyes rétegek általában minden csúcsukban azonos módon viselkednek. Az egyes partíciókat **rétegeknek** nevezzük, és számos

előnyös tulajdonsággal bírnak. Egyrészt a különböző rétegek szemantikailag egyre magasabb szintű absztrakciókat reprezentálnak, másrészt párhuzamosan is kiszámíthatók. Mivel a rétegek minden csúcsukban hasonlóan épülnek fel, ezért gyakran megkülönböztetünk speciális célú rétegeket. Ilyen az időkomponenssel bíró adatok kezelésére kifejlesztett rekurrens/visszacsatolt réteg, a lokális információkat összegző konvolúciós réteg vagy az általánosabb, teljesen összekötött réteg. A rétegekből (vagy anélkül) kialakított számítási gráfot a neurális hálózat architektúrájának is szokás nevezni.

### Tanulás

Neurális hálózatok architektúrája leggyakrabban emberi munkával készül el. Van példa arra is, hogy jól működő (a célnak megfelelő pontosságú eredményt adó) neurális hálózatok paramétereit is emberi erővel vagy egyszerű konstruktív szabályokkal állítanák be, azonban ezek a megoldások szélsőséges esetekben, kis méretű hálózatokon szoktak működni. A paraméterek automatikus konfigurálását szokás a hálózat tanításának nevezni, míg az erre alkalmazott algoritmus paramétereit hiperparaméternek. Számos ilyen algoritmus létezik, azonban manapság a sztochasztikus gradiens leszállás variánsait szokás használni. Az SGD-alapú tanulás során véletlenszerű módon inicializáljuk a paramétereket, ami után a tényleges tanulás a kimenetre alkalmazott költségfüggvény segítségével történik, melynek függvényében módosítjuk a hibás (rossz eredményre jutó) paramétereket. Különösen nehéz értékelni, hogy egy rossz kimenethez egy adott paraméter mennyire járult hozzá, azonban itt nem térünk ki az erre alkalmazott módszerekre. A tanításnak lehetnek másodlagos követelményei is, mint például a kapott paraméterek komplexitásának (értékskálájának) csökkentése. Fontos kiemelni, hogy nem mentesek a gradiensalapú megoldások a problémáktól (kezdve a paraméter/kimenet hozzárendelési problémától, a szükségszerűen differenciálható költségfüggvényen át, a deriválhatósági szempontból megfelelő inicializálási módszerekig bezárólag), és egyáltalán nem csak ilyen tanulóalgoritmusok léteznek. Evolúciós algoritmusok hatásfokát a neurális hálózatok tanításának kontextusában már számos szerző vizsgálta, illetve javította[24, 25]. Bár ígéretes eredményeket érhetők el genetikus algoritmusokkal, és nem is szükséges hozzá, hogy deriválható költségfüggvényt alkalmazzunk, de - egyelőre - ez a módszer nem terjedt el szé-

les körben, azonban - neuroevolúció néven - virágzó szakterületté nőtte ki magát a problémakör.

### 1.4.4. Gráf konvolúciós neurális hálózatok

#### Laplacian

#### Hipergráf Laplacian

#### Hipergráf konvolúciós neurális hálózatok

### 1.4.5. Alkalmazott Euler-diagram reprezentáció

### 1.4.6. Vizsgált optimalizációs módszerek

### 1.4.7. Vizsgált inicializációs módszerek

### 1.4.8. Vizsgált heurisztikák

#### Szemantikus heurisztikák

#### Esztétikai heurisztikák

#### Futásidő analízis

## 1.5. Mérések és következtetések

# 2. fejezet

# Felhasználói dokumentáció

# 3. fejezet

# Fejlesztői dokumentáció

# 4. fejezet

# Összegzés

# Irodalomjegyzék


[1] C. Berge. *Hypergraphs*. North Holland Publishing Company, 1989. ISBN: 9780080880235.

[2] Bilal Alsallakh és tsai. "The State-of-the-Art of Set Visualization". *Comput. Graph. Forum* 35.1 (2016. febr.), 234–260. ISSN: 0167-7055. DOI: 10.1111/cgf.12722. URL: https://doi.org/10.1111/cgf.12722.

[3] Margaret E. Baron. "A Note on the Historical Development of Logic Diagrams: Leibniz, Euler and Venn". *The Mathematical Gazette* 53.384 (1969), 113–125. old. ISSN: 00255572. URL: http://www.jstor.org/stable/3614533.

[4] Anne Verroust és Marie-Luce Viaud. "Ensuring the Drawability of Extended Euler Diagrams for up to 8 Sets". *Diagrammatic Representation and Inference*. Szerk. Alan F. Blackwell, Kim Marriott és Atsushi Shimojima. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, 128–141. old. ISBN: 978-3-540-25931-2.

[5] P. Simonetto és D. Auber. "Visualise Undrawable Euler Diagrams". *2008 12th International Conference Information Visualisation*. 2008, 594–599. old. DOI: 10.1109/IV.2008.78.

[6] Luana Micallef és Peter Rodgers. "eulerForce: Force-directed layout for Euler diagrams". *Journal of Visual Languages & Computing* 25.6 (2014). Distributed Multimedia Systems DMS2014 Part I, 924 –934. old. ISSN: 1045-926X. DOI: https://doi.org/10.1016/j.jvlc.2014.09.002. URL: http://www.sciencedirect.com/science/article/pii/S1045926X14000810.

[7] Peter Rodgers, Leishi Zhang és Helen Purchase. "Wellformedness Properties in Euler Diagrams: Which Should Be Used?": *IEEE transactions on visualization*

*and computer graphics* 18 (2012. júl.), 1089–100. old. DOI: 10.1109/TVCG.2011.143.

[8] J. Flower, P. Rodgers és P. Mutton. "Layout metrics for Euler diagrams". *Proceedings on Seventh International Conference on Information Visualization, 2003. IV 2003.* 2003, 272–280. old. DOI: 10.1109/IV.2003.1217990.

[9] Mithileysh Sathiyanarayanan és John Howse. "Well-matchedness in Euler Diagrams". 2014. júl. DOI: 10.13140/2.1.2861.9524.

[10] A. Blake és tsai. "Does the orientation of an Euler diagram affect user comprehension?": *Proceedings: DMS 2012 - 18th International Conference on Distributed Multimedia Systems* (2012. jan.), 185–190. old.

[11] Gem Stapleton és tsai. "Automatically drawing Euler diagrams with circles". *Journal of Visual Languages & Computing* 23.3 (2012), 163 –193. old. ISSN: 1045-926X. DOI: https://doi.org/10.1016/j.jvlc.2012.02.001. URL: http://www.sciencedirect.com/science/article/pii/S1045926X12000134.

[12] S. Chow. "Generating and Drawing Area-Proportional Euler and Venn Diagrams". Dissz. University of Victoria, 2007. URL: http://hdl.handle.net/1828/128.

[13] Stirling Chow és Frank Ruskey. "Drawing Area-Proportional Venn and Euler Diagrams". *Graph Drawing*. Szerk. Giuseppe Liotta. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, 466–477. old. ISBN: 978-3-540-24595-7.

[14] Stirling Chow és Frank Ruskey. "Towards a General Solution to Drawing Area-Proportional Euler Diagrams". *Electronic Notes in Theoretical Computer Science* 134 (2005). Proceedings of the First International Workshop on Euler Diagrams (Euler 2004), 3 –18. old. ISSN: 1571-0661. DOI: https://doi.org/10.1016/j.entcs.2005.02.017. URL: http://www.sciencedirect.com/science/article/pii/S1571066105050395.

[15] Andrew Fish, Babak Khazaei és Chris Roast. "User-comprehension of Euler diagrams". *Journal of Visual Languages & Computing* 22.5 (2011), 340 –354. old. ISSN: 1045-926X. DOI: https://doi.org/10.1016/j.jvlc.2011.

01.002. URL: http://www.sciencedirect.com/science/article/pii/S1045926X11000036.

[16] Andrew Blake és tsai. "The Impact of Shape on the Perception of Euler Diagrams". *Diagrammatic Representation and Inference*. Szerk. Tim Dwyer, Helen Purchase és Aidan Delaney. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, 123–137. old. ISBN: 978-3-662-44043-8.

[17] G. Stapleton és tsai. "Inductively Generating Euler Diagrams". *IEEE Transactions on Visualization and Computer Graphics* 17.1 (2011), 88–100. old. DOI: 10.1109/TVCG.2010.28.

[18] M. Wang és tsai. "SketchSet: Creating Euler diagrams using pen or mouse". *2011 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC)*. 2011, 75–82. old. DOI: 10.1109/VLHCC.2011.6070382.

[19] Roger Fletcher. "On the Barzilai-Borwein Method". *Optimization and Control with Applications*. Szerk. Liqun Qi, Koklay Teo és Xiaoqi Yang. Boston, MA: Springer US, 2005, 235–256. old. ISBN: 978-0-387-24255-2.

[20] John H. Holland. *Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence*. Cambridge, MA, USA: MIT Press, 1992. ISBN: 0262082136.

[21] Warren Hare, Julie Nutini és Solomon Tesfamariam. "A survey of non-gradient optimization methods in structural engineering". *Advances in Engineering Software* 59 (2013), 19 –28. old. ISSN: 0965-9978. DOI: https://doi.org/10.1016/j.advengsoft.2013.03.001. URL: http://www.sciencedirect.com/science/article/pii/S0965997813000288.

[22] J. Kennedy és R. Eberhart. "Particle swarm optimization". *Proceedings of ICNN'95 - International Conference on Neural Networks*. 4. köt. 1995, 1942–1948 vol.4. DOI: 10.1109/ICNN.1995.488968.

[23] Yuhui Shi és B.Gireesha Obaiahnahatti. "A Modified Particle Swarm Optimizer". 6. köt. 1998. jún., 69 –73. old. ISBN: 0-7803-4869-9. DOI: 10.1109/ICEC.1998.699146.

[24] F. Ahmad és tsai. "Performance comparison of gradient descent and Genetic Algorithm based Artificial Neural Networks training". *2010 10th International Conference on Intelligent Systems Design and Applications*. 2010, 604–609. old. DOI: 10.1109/ISDA.2010.5687199.

[25] Edmund Ronald és Marc Schoenauer. "Genetic Lander: An Experiment in Accurate Neuro-Genetic Control". *Proc. 3rd Conf. Parallel Problem Solving from Nature*. Springer-Verlag, 1994, 452–461. old.