Eötvös Loránd Tudományegyetem
Természettudományi Kar

Tóth Szilvia Ágnes

Alkalmazott matematikus MSc

Gráfok síkbarajzolása

Témavezető: Bérczi Kristóf

Budapest, 2014

Tartalomjegyzék

Bevezetés

A gráfelmélet egyik régóta kutatott kérdése a gráfok síkbarajzolása. Először arra keresték a választ, hogy egy tetszőleges gráf mikor rajzolható le a síkban úgy, hogy az élek ne keresztezzék egymást. Észrevették, hogy bizonyos gráfokat nem lehet így lerajzolni. Amelyekre viszont létezik ilyen lerajzolás, azokat a gráfokat nevezzük síkbarajzolható gráfoknak. Kuratowski 1930-ban belátta, hogy a síkbarajzolhatóság karakterizálható tulajdonság [24].

Sok érdekes kérdéssel foglalkoztak azóta a témában, például, hogy hogyan lehet lerajzolni egy nem síkbarajzolható gráfot a síkban úgy, hogy az éleknek minél kevesebb kereszteződése legyen. Másik aktuálisan kutatott kérdés, ha adott n pont a síkban és adott egy n csúcsú gráf, akkor van-e a gráfnak olyan síkbarajzolása, melyben az élek egyenes vonalak, és a gráf csúcsai a megadott pontokra esnek.

Motiváció

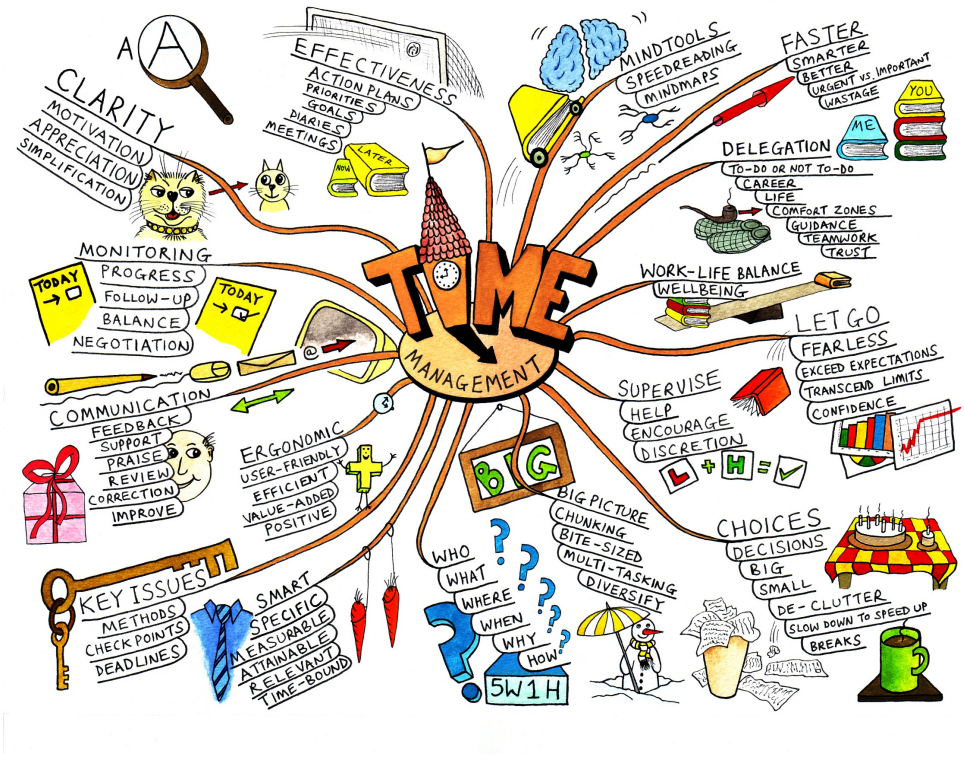
Érdekes és aktuális téma, melyet rengeteg alkalmazás motivál, hogy hogyan lehet egy gráfot szépen lerajzolni. Az információ megjelenítés ma egy külön tudományágnak számít, melynek célja az, hogy találjon olyan módszereket, melyekkel absztrakt adatok vizuálisan ábrázolhatók, és ezáltal könnyebben érthetővé tehetők mások számára. Adatokat és összefüggéseket kézenfekvő módon lehet gráfokkal ábrázolni: a csúcsok reprezentálják az egyes adatokat, az élek pedig a közöttük fennálló kapcsolatokat. Ahhoz, hogy a gráf valóban megkönnyítse a mögöttes adatstruktúra megértését, a gráfot meg kell jeleníteni, méghozzá az emberi megértés szempontjából fontos tulajdonságokat figyelembe véve. A feladat tehát az, hogy pontosan megadjuk egy gráf pontjainak a koordinátáit úgy, hogy bizonyos "esztétikai" feltételeket teljesítsen az így elkészült rajz.

Abban az esetben, ha minden csúcshoz tartozó információ egyformán fontos, akkor nem lehet túl nagy egy rajz területe ahhoz képest, hogy mekkora benne két csúcs között a legkisebb távolság. Ellenkező esetben egy fix méretű képernyőn megjelenítve a rajzot, két közel lévő csúcsra vonatkozó információ elvész, mert nem tudjuk kiolvasni a ráírt szöveget. Hasonló okokból a rajz vízszintes és függőleges méretének is nagyjából azonosnak kell lennie - például egy aránytalanul széles kép esetében - , vagy nem látjuk egészben a gráfot és így a struktúra egészét, vagy pedig bezsúfoljuk a rajzot egy képernyőre és akkor ismét nem láthatjuk az egymáshoz közel eső csúcsokra írt szövegeket. Ezeket a szempontokat hívjuk a rajz esztétikáinak, melyeket a következő részben formálisan is ismertetünk.

A dolgozatban fák lerajzolására mutatunk be algoritmusokat. Miért éppen fák lerajzolásával foglalkozunk? - vetődik fel a jogos kérdés. Rengeteg alkalmazásban hordoz lényeges információt az adatok hierarchikus ábrázolása, ezért ez központi kérdés a már említett információ megjelenítés témakörében. Hierarchikus adatok és összefüggések ábrázolására remek megoldást nyújt egy fagráf. Ennek alkalmazási

területe igen sokrétű. Az informatika területén belül az objektumorientált osztályhierarchia (UML class diagramok), adatbázisok relációinak vagy függvényhívások diagramjának megjelenítésére, illetve weboldalak struktúrájának tervezésekor és fájlrendszerek ábrázolására is használhatunk fákat. A biológia területén az evolúciós fák, a kémia területén molekulák rajzaihoz, illetve tudományágaktól függetlenül döntési fák, folyamatábrák, családfák és organigramok megjelenítésekor is alkalmazhatunk fákat lerajzoló algoritmusokat. Mindig az alapján választjuk ki, hogy melyik lerajzolási módszert használjuk, hogy melyik nyújtja a legtöbb információt a mögöttes adatokról. Egy jó lerajzolás segíti az adatok megértését, viszont egy nem megfelelő lerajzolás összezavarhatja a felhasználókat.

Ami miatt én elkezdtem foglalkozni a témával, az a gondolattérképek, vagy elmetérképek (mindmap) automatikus lerajzolásának ötlete volt. Ez a módszer arra való, hogy képileg kiemeljünk információkat. Egy adott szöveg megértésére, lexikális anyag megtanulásának elősegítésére, vagy bármilyen rendszer struktúrájának áttekintésére használható. Nagyon jól segíti egy lexikális anyag megtanulását. Általában kézzel érdemes készíteni, de rengeteg elmetérkép rajzoló program készült már, melyek szintén jól használhatók. Alapja, hogy nem lineárisan készítünk jegyzetet a témakörről, hanem a kulcsfogalmakat egy gráfszerű ábrán helyezzük el. Ha kézzel készítjük, akkor a témakör központi fogalmát helyezzük a lap közepére, majd a hozzá tartozó fogalmakat sugár irányban helyezzük középről kiindulva, összekötve az előző szint elemeivel. Fontos még, hogy minél kevesebb szöveget használjunk, inkább helyettesítsünk a fogalmat képekkel, ábrákkal. Használhatósága abban rejlik, hogy az emberek nagy része vizuális beállítottságú, így a képek által megjelenített gondolatokat könnyebben befogadja és megjegyzi, mint a puszta szöveget. Ezt segíti elő a fogalmak kapcsolatának képi megjelenítése illetve a fogalmak képekkel való ábrázolása.



1. ábra. Példa elmetérképre

A dolgozatról

A dolgozat célja, hogy betekintést adjon a fák lerajzolásának témakörébe. A legismertebb és leggyakrabban használt lerajzolási módszereket mutatjuk be, ismertetve a kapcsolódó hivatkozásokat, melyekben az egyes algoritmusok különböző változatait találhatjuk meg. Ezen kívül minden fejezet végén felsoroljuk az adott módszer felhasználási területeit.

A legtöbb alkalmazáshoz gyökeres fák lerajzolására van szükség, így mi is ilyen lerajzolási eljárásokat mutatunk be. Az itt tárgyalt algoritmusok mindegyike egyenes vonalakkal rajzolja le a fa éleit, és csak megemlítjük azokat a változatokat, amikor töröttvonalakkal, körívekkel, vagy más görbékkel ábrázoljuk az éleket. Több esetben először bináris fákra mutatunk lerajzolási módszereket, és csak utána általánosítunk tetszőleges gyökeres fákra. A harmadik fejezetben bemutatott sugárirányú lerajzolást nem számítva mindegyik algoritmusban közös, hogy az "oszd meg és uralkodj" elven alapul. Eszerint szétvágjuk a fát néhány részre, rekurzívan elkészítjük az egyes részek rajzát, majd a részeket összeillesztjük.

Az első fejezetben tisztázzuk a gráfrajzoláshoz szükséges alapfogalmakat, majd a második fejezettől mutatjuk be a különböző farajzolási algoritmusokat. Az egyik legrégebben publikált lerajzolási módot mutatjuk be a második fejezetben, a szintezett lerajzolást. Alapja az a kézenfekvő gondolat, hogy azokat a csúcsokat, melyek fa azonos szintjén vannak, a gyökértől függőlegesen azonos távolságban helyezzük el. Ennek egy általánosítása, a sugárirányú lerajzolás szerepel a harmadik fejezetben. Itt a gyökértől azonos távolságban helyezzük el az egy szinten lévő csúcsokat, azaz minden csúcs egy-egy körön helyezkedik el, melynek a középpontjában a gyökércsúcs van. A negyedik és ötödik fejezetben olyan lerajzolásokat mutatunk be, melyekben bármely két szomszédos él vagy egyazon egyenesre esik, vagy pedig merőlegesek egymásra. A negyedik fejezetben ismertetett HV lerajzolás ezen kívül teljesíti, hogy minden csúcs gyökeres részfáját tartalmazó befoglaló téglalap tartalmazza minden gyermeke gyökeres részfáját tartalmazó befoglaló téglalapot, és ezek nem metszik egymást. Az ötödik fejezet tekercselt lerajzolásának általunk bemutatott változatára nem igaz az előbbi tulajdonság, viszont a HV lerajzolásának aránytalanul nagy a szélessége a mélységéhez képest, amit a tekercselt lerajzolás lényegesen javít. Ez utóbbinál a magasság és a szélesség aránya $O(1)$. Az hatodik és hetedik fejezet a buborék lerajzolás két változatát mutatja be. Ennek a lerajzolásnak az alapgondolata az, hogy minden gyermek a szülőjétől azonos távolságban helyezkedik el, így minden gyökeres részfa egy-egy diszjunkt körlap belsejébe kerül. A hatodik fejezetben a buborék lerajzolás egyik változatát ismertetjük, melyet több szempont szerint javítunk. A hetedik fejezetben egy kevésbé alkalmazott, de elméleti szempontból jelentős algoritmus szerepel, a nehéz utas lerajzolás. Különlegessége abban áll, hogy minden csúcs körül a szomszédos élek közötti szögek azonos nagyságúak.

1. fejezet

Alapfogalmak

A dolgozatban bemutatott algoritmusok szinte mindegyike **gyökeres fákat** (fenyőket) rajzol le. Gyökeres fának hívunk egy $T = (V, E)$ irányított fát, ha adva van egy kitüntetett $r \in V$ csúcsa, melyből csak kifelé vezetnek élek és r-en kívül minden más $v \in V$ csúcsba pedig pontosan egy él lép be. Az r csúcsot a fa gyökerének hívjuk. Legyen $v \in T$ tetszőleges csúcs. A v-ből induló élek végpontjait v gyermekeinek nevezzük, és ezek halmazát $child(v)$-vel jelöljük. A v-be érkező él kezdőpontját a v szülőjének nevezzük, és $p(v)$-vel jelöljük. A v **csúcs fokszámának** nevezzük a rá illeszkedő élek számát, és $d(v)$-vel jelöljük. A v-ből induló, illetve a v-be érkező élek számát **ki- és befokszámnak** nevezzük, és $d_{out}(v)$-vel illetve $d_{in}(v)$-vel jelöljük. A v-ből irányított úton elérhető csúcsok részfáját, azaz a v gyökerű legnagyobb részfát T-ben v **részfájának** nevezzük és T_v-vel jelöljük. Azt mondjuk, hogy v **csúcs szintje** k, ha az r gyökércsúcstól a v-be menő egyértelműen meghatározott út pontosan k élből áll. A v szintjét $level(v)$-vel jelöljük. Nyilván $level(r) = 0$. A T **magasságának** nevezzük és $h(T)$-vel jelöljük a fa legnagyobb szintű csúcsának szintjét. T_v magasságát $h(v)$-vel jelöljük. T leveleinek számát $l(T)$-vel, T_v leveleinek számát pedig $l(v)$-vel jelöljük.

Bináris fa alatt olyan gyökeres fát értünk, melyben minden pont kifoka legfeljebb kettő. Egy tetszőleges G gráfot **rendezettnek** nevezünk, ha minden csúcshoz adva van a rá illeszkedő éleknek egy ciklikus sorrendje. Ha a gráf egy bináris fa, akkor a rendezettség azt jelenti, hogy minden v csúcs két gyermekének van egy meghatározott sorrendje, azaz megmondjuk, hogy melyik a **bal és jobb gyereke**, melyeket v_{bal}-al és v_{jobb}-al jelölünk. A rendezett bináris fába azonban azt is beleértjük, hogy amelyik csúcsnak egy gyereke van, ott is megmondjuk, hogy az bal vagy jobb gyerek. v bal és jobb oldali gyermekeinek a részfáit **bal és jobb részfának** hívjuk és $T_{bal}(v)$-vel és $T_{jobb}(v)$-vel jelöljük. A rendezettség fogalma egyébként azért fontos, mert van olyan lerajzolás, amelynél követelmény, hogy minden csúcsnál az élek körüljárási sorrendje az előre megadott sorrenddel egyezzen meg. Ekkor a lerajzolásról azt mondjuk, hogy **megőrzi az élek rendezettségét**.

A dolgozat során többször fogunk indexeket használni, például egy v csúcs gyermekeit általában $v_1, v_2, ..., v_k$-el jelöljük. Amikor egy $(1, 2, ..., n)$ halmaz elemein kivonást vagy összeadást végzünk, azt mindig ciklikusan értjük.

1.1. Hogyan kell gráfokat lerajzolni?

Rengeteg szempont szerint tudunk gráfokat szépen lerajzolni. Ez főleg a mögöttes alkalmazástól függ. A [35] könyv részletesen beszámol a különböző gráfrajzolási alapfogalmakról, melyeket itt ismertetünk, de előbb definiáljuk mit értünk egy gráf lerajzolásán.

1.1.1. Definíció. *Egy $G = (V, E)$* ***gráf lerajzolásán****, vagy* ***rajzán*** *egy Γ függvényt értünk, mely minden csúcsot különböző $\Gamma(v)$ pontra képez le a síkban, és minden uv élet egy $\Gamma(uv)$ egyszerű nyitott Jordan görbére képez, melynek végpontjai $\Gamma(u)$ és $\Gamma(v)$.*

A következőkben bemutatjuk azokat a szempontokat és tulajdonságokat, amiket figyelembe veszünk egy lerajzolási algoritmus kidolgozásánál. A szakirodalomban általában mindegyik tulajdonságot esztétikának hívnak, mert végeredményben minden feltétel azért van, hogy "széppé", azaz jobban érthetővé tegye a gráf lerajzolását. A [35] könyv viszont több kategóriába osztja ezeket a tulajdonságokat, melyeket mi is megkülönböztetünk egymástól. Egyrészt vannak olyan fix előírások, amiket a rajznak mindenképpen teljesítenie kell. Ilyen előírás lehet például, hogy az élek egyenes vonalak legyenek. Ezeket a megkötéseket hívjuk konvencióknak. Vannak olyan szempontok, amiket csak megközelíteni tudunk. Célunk, hogy az elkészült rajz az adott szempontot minél nagyobb mértékben teljesítse. Ilyen például a korábban már említett példa, hogy egy rajz szélessége és magassága legyen közel azonos nagyságú.

1.1.1. Gráfrajzolási konvenciók

A gráfrajzolási konvenciók olyan alaptulajdonságok, amelyeket egy adott alkalmazás előír. Például az objektumorientált programozásban az osztályok hierarchiáját modellező UML diagramok esetében a csúcsok téglalapokkal vannak ábrázolva, míg az élek olyan töröttvonalakkal, melyek vízszintes és függőleges szakaszokból állnak. Az ilyen rajzot nevezzük ortogonális rajznak. Bemutatunk pár gyakran használt konvenciót:

1.1.2. Definíció. *Egy $G = (V, E)$ gráf egy lerajzolását*

- ***Töröttvonalas rajznak*** *nevezzünk, ha minden él képe töröttvonal. Ekkor az él képén a töröttvonal töréspontjait* ***éltörésnek*** *hívjuk.*
- ***Egyenesvonalas rajznak*** *nevezzük, ha minden él egy egyenes szakasz.*
- ***Ortogonális rajznak*** *nevezzük, ha minden él olyan töröttvonal, mely vízszintes és függőleges szakaszokból áll.*
- ***Rácsrajznak*** *nevezzük, ha az ábra egy négyzetrácsra van lerajzolva és minden csúcs, kereszteződés és éltörés egy-egy rácspont.*
- ***Síkrajznak*** *vagy* ***síkbarajzolásnak*** *nevezzük, ha az élek nem keresztezik egymást.*

Ha a gráf egy aciklikus digráf, akkor egy lerajzolását ***ereszkedő (emelkedő) rajznak*** *nevezzük, ha minden él egy töröttvonal úgy, hogy minden uv irányított élre az u csúcs y koordinátája nem nagyobb (nem kisebb) mint a v csúcs y koordinátája.* ***Szigorúan ereszkedő (emelkedő)*** *rajzról beszélünk, ha az u és v megfelelő koordinátái között szigorú egyenlőtlenség áll fenn.*

1.1.2. Esztétika

A gráfjrajzolás esztétikáinak, hívjuk azokat a tulajdonságokat, melyek alapján szépnek nevezzük egy lerajzolást. Ezeket a tulajdonságokat, amennyire csak lehet, teljesíteni szeretnénk, így a lerajzolás egyfajta célfüggvényének is tekinthetjük őket. A leggyakrabban használt esztétikák a következők:

- **Keresztezések száma:** Minimalizáljuk a keresztezések számát. Ennek speciális esete a síkrajz.
- **Terület:** Minimalizálni szeretnénk a rajz területét, melynek definíciója változó. Az egyik ilyen lehetséges definíció, hogy tekintsük egységnek a két legközelebbi csúcs távolságát, és ennek függvényében a rajz területe legyen a legkisebb sokszög területe, amely tartalmazza a rajzot. A második, harmadik és negyedik fejezetben rácsrajzokat mutatunk be. Itt egy rajz területe alatt a rácstávolságot egységnek tekintve a rajz **befoglaló téglalapjának** területét értjük, azaz annak a legkisebb rácstéglalapnak a területét, mely tartalmazza a rajzot.
- **Összélhossz:** Minimalizáljuk élek hosszainak összegét.
- **Képfelbontás** (aspect ratio) [35]: A rajz **befoglaló** téglalapjának hívjuk azt a legkisebb a téglalapot, amely tartalmazza a rajzot. A képfelbontás a befoglaló téglalap kisebb és a nagyobb oldal hosszának a hányadosa. Célunk ennek maximalizálása.
- **Szögarány** (angular resolution) [26]: A legnagyobb és legkisebb szög hányadosa, melyeket a gráf egy csúcsából kiinduló két szomszédos él bezár. (A legkisebb és a legnagyobb szöghöz tartozó csúcs nem feltétlenül azonos!) Célunk minimalizálni ezt az értéket.
- **Szögfelbontás** (aspect ratio) [26]: Egy v **csúcs szögfelbontása** a legkisebb belőle kiinduló szomszédos élek közötti szög. Egy G **gráf szögfelbontása** az összes $v \in G$ csúcs szögfelbontásának a minimuma, azaz a legkisebb egy csúcsból kiinduló két szomszédos él közötti szög. Az egyik optimalizálási lehetőség, ha maximalizáljuk a gráf szögfelbontását (lásd: 6 fejezet). Egy lerajzolásának **tökéletes a szögfelbontása**, ha minden v csúcs minden szomszédos élpárja által bezárt szög pontosan $\frac{2\pi}{d(v)}$.

2. fejezet

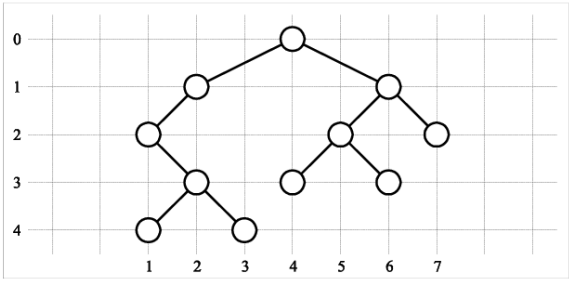
Szintezett lerajzolás

Ha gyökeres fát szeretnénk átláthatóan lerajzolni, akkor egy hasznos információ lehet, hogy mely csúcsok vannak ugyanakkora távolságra a gyökércsúcstól. Például, ha egy családfát szeretnénk megjeleníteni, akkor jogos az igény, hogy egy generáció tagjait könnyedén le tudjuk olvasni az ábráról. Erre a problémára egy jó megoldás a szintezett lerajzolás, melynek alapja az a gondolat, hogy azok a csúcsok, melyek a fa azonos szintjén vannak, kerüljenek egy egyenesre, és ezek az egyenesek legyenek párhuzamosak egymással. Ezen az elgondoláson alapul az egyik legrégebben publikált lerajzolási mód. Több változata is létezik, és a mai napig adnak új és új változatokat. Ebben a fejezetben azt az algoritmust mutatjuk be melyet Tilford és Riengold írt le a 1981-ben [30].

A továbbiakban egy lerajzolást **szintezettnek** nevezünk, ha minden csúcs y koordinátája megegyezik a mélységének ellentettjével.

2.1. Bináris fák szintezett lerajzolása

A szintezett lerajzolás algoritmusát először bináris fákra ismertetjük, azaz olyan gyökeres fákra, melyben minden pont kifoka legfeljebb kettő. Ezt fogjuk később általánosítani tetszőleges fokszámú gyökeres fákra.



2.1. ábra. Példa szintezett rajzra [1].

A lerajzolás pontos tulajdonságainak meghatározásához szükség van két fogalom bevezetésére.

2.1.1. Definíció. *Legyen T és S két bináris fa. Azt mondjuk, hogy S és T* ***egyszerűen izomorf****, ha mindkettő egy csúcsból áll, vagy ha T bal részfája egyszerűen izomorf S bal részfájával, T jobb részfája egyszerűen izomorf S jobb részfájával.* ***Tengelyesen izomorfnak*** *nevezzük őket, ha T-ben minden csúcshoz tartozó jobb és bal részfát felcserélve az így kapott gráf egyszerűen izomorf S-sel.*

2.1.2. Állítás. *Legyen T egy rendezett bináris fa n csúccsal. A szintezett lerajzolás algoritmusa egy olyan rajzot ad, amely:*

1. *szintezett,*
2. *keresztezésmentes,*
3. *egyenesvonalas rajz,*
4. *szigorúan ereszkedő,*
5. *megőrzi az élek rendezettségét,*
6. *bármely két csúcs vízszintes legalább 2 vagy függőleges távolsága legalább 1,*
7. *egy v csúcs x koordinátája a gyermekei x koordinátájának az átlaga (kivéve, hogyha egy gyermeke van v-nek),*
8. *függetlenül attól, hogy hol helyezkednek el a fában, két egyszerűen izomorf részfa rajza egybevágó, két tengelyesen izomorf részfa rajza egymás tükörképe.*

2.1.1. Algoritmus

Célunk, hogy mutassunk egy lineáris idejű algoritmust, mely a 2.1.2. állítás tulajdonságait teljesíti.

Az eljárás két fabejárással megtehető. Az első, postorder bejárással megadjuk minden csúcsnak az ő szülőjéhez vett relatív vízszintes távolságát. A második preorder bejárással pedig minden csúcsnak megmondjuk a végső x koordinátáját. Az algoritmus előfeltétele, hogy tudjuk minden csúcsra az ő részfájának magasságát, melyet egy postorder bejárással megkaphatunk $O(n)$ időben.

A relatív koordináták meghatározását úgy képzelhetjük el, mintha egy v gyökerű részfa két részfájának rajzát már lerajzolnánk egy egy papírra, majd a kontúrjuknál kivágnánk őket. Ezután a két rajzot a gyökerüknél egymásra helyezzük, majd addig "mozgatjuk" el őket, míg a két legközelebbi csúcs távolsága a két rajzon egy előre meghatározott minimális távolságot elér. Ezután v-t a két részfa gyökerei koordinátáinak átlagában helyezzük el egy egységgel függőlegesen felettük. Ha csak egy gyereke van v-nek, például v_{jobb}, akkor v-t vízszintesen a minimális távolság felének távolságában helyezzük el úgy, hogy tőle jobbra essen v_{jobb}.

Az algoritmus tisztázása előtt szükségünk van a következő fogalomra.

2.1.3. Definíció. *Egy $h(T)$ magasságú T fa* ***bal kontúrja*** *$v_1, v_2, ...v_h$, ha v_i a fa i. szintjének legbaloldalibb eleme. A* ***jobb kontúr*** *hasonlóan definiálható.*

Amikor egy v csúcsot dolgozunk fel, akkor végighaladunk a jobb részfájának a bal kontúrján illetve a bal részfájának a jobb kontúrján, és kiszámítjuk, hogy mennyivel kell széthúzni őket, hogy a minimális távolság meglegyen az adott szinten.

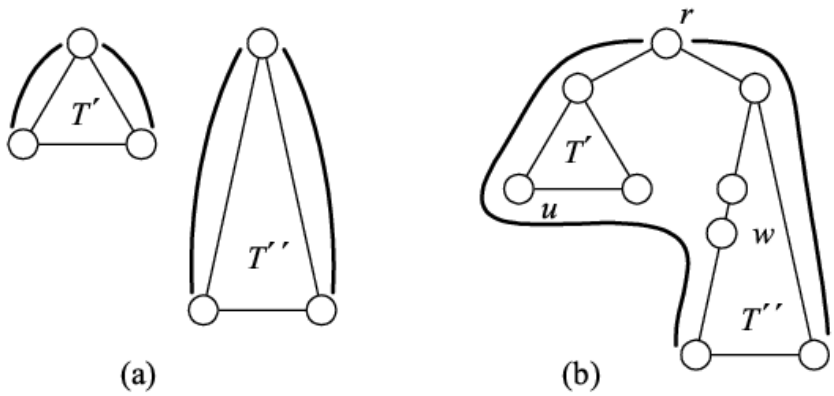
2.1.4. Algoritmus (Kontúrok kiszámítása)**.**

Ha T_v egyetlen csúcsból áll, akkor ő a bal és jobb oldali kontúr. Ha T_v-nek legalább két csúcsa van, akkor a következő két eset lehetséges:

Első eset: *$T_{bal}(v)$ és $T_{jobb}(v)$ magassága ugyanannyi. Ekkor T_v bal kontúrja nem más, mint a $T_{bal}(v)$ részfa bal kontúrjának és a v csúcsnak az egyesítése, a jobb kontúr pedig $T_{jobb}(v)$ részfa a jobb kontúrja és v egyesítése.*

Második eset: *$T_{bal}(v)$ és $T_{jobb}(v)$ magassága nem egyezik. Legyen T_v bal részfájának magassága kisebb, mint a jobb részfájának a magassága, és jelöljük a bal részfa magasságát h-val. Ekkor a T_v bal*

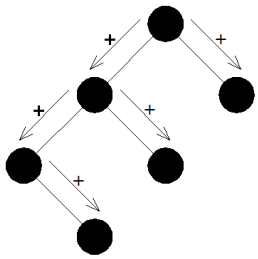
kontúrja a v csúcsot, a T_{bal} balkontúrját és a T_{jobb} részfa $h+1$. szintjétől a legalsó szintig vett bal kontúrjának csúcsait tartalmazza. Mivel a T_v jobb részfája a magasabb, ezért T_v jobb kontúrja T_{jobb} részfa jobb kontúrjának és a v csúcsnak az egyesítéséből jön létre, hasonlóan az első esethez (lásd 2.2. ábra).



2.2. ábra. Kontúrok kiszámítása [1].

A postorder bejárásnál tehát minden lépésnél frissíthetjük a bal és jobb kontúrokat. Most az a feladatunk, hogy minden lépésnél meghatározzuk a kontúrok segítségével, hogy mennyire kell széthúzni a két részfát ahhoz, hogy minden szinten a minimális távolság elég nagy legyen a két részfa között.

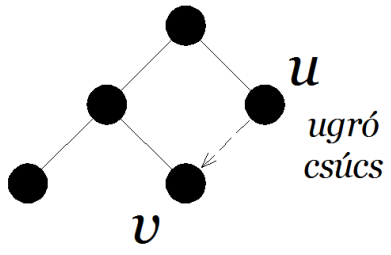
Minden csúcsnál csak a szülőjéhez vett relatív távolságot tároljuk el, így az abszolút koordinátáit nem tudjuk frissíteni, különben nem lenne lineáris az algoritmus futásideje. A részfák gyökérétől a gyermekek mentén folyamatosan összegezhetjük a kontúrok mentén ezeket a relatív távolságokat, így az abszolút távolságot ki tudjuk számolni a két részfa között az adott szinten. Viszont ez csak addig tehető meg, amíg a kontúrban csak szülőből gyerekbe váltunk, azaz a kontúr egy út a gyökérből a legalsó szint egy csúcsáig. De ha a kontúron van egy olyan csúcs, melynek nincs gyereke, akkor a kontúron az ezután következő csúcs relatív koordinátája a szülőjéhez képest semmitmondó. Bár összegezhetnénk a gyökértől a legrövidebb úton eddig a csúcsig a relatív távolságokat, de ekkor nem lenne lineáris futásidejű az algoritmus, mivel csúcsonként akár minden lépésben az egész részfát be kellene járnunk (lásd a 2.3. ábra).



2.3. ábra. A kontúr relatív koordinátáinak meghatározásánál legrosszabb esetben szintenként végig kell menni a részfa gyökeréig minden szinten.

Az ilyen eseteket **ugrásoknak** hívjuk, és ezek csak az 2.1.4. algoritmus 2. esetében fordulnak elő. Az ugrásokra egy virtuális **ugró élt** rakunk (lásd 2.4. ábra). A kontúr felsőbb szintjén lévő csúcsát -

ahonnan ugrunk- **ugró csúcsnak** nevezzük, és beállítjuk az ő bal vagy jobb oldali elemének a kontúr következő elemét, attól függően, hogy az ugró él végpontja milyen irányban helyezkedik el az ugró csúcshoz képest. Majd ezen két csúcs közti távolságot adjuk meg az ugró csúcs relatív koordinátájának. Így azt érjük el, ha egy kontúron végigmegyünk a részfa gyökerétől a legalsó szintig, akkor mindig csak szülőből gyerekbe váltunk, és így a kontúr minden csúcsára meghatározhatjuk a részfa gyökerétől vett abszolút távolságot. Természetesen ugró csúcsokat megjelöljük egy címkével azért, hogy a 2. preorder bejárásnál, amikor az abszolút koordinátákat számítjuk ki, már ne használjuk az ugró éleket.



2.4. ábra. Ugró élre és ugró csúcsra példa.

Ahhoz, hogy az ugró csúcs és a virtuális gyermeke közötti a távolságot kiszámítsuk, bevezetjük az **extrém csúcs** fogalmát. Ezek az ugró csúcsok ugyanis mindig extrém csúcsok lesznek abban az értelemben, hogy egy részfa legalsó szintjének legszélső elemei lesznek a bal vagy jobb oldalon. Ilyenkor ugyanis, ha hozzáillesztjük az ő aktuálisan vizsgált részfája mellé a testvér részfáját, mely nagyobb mint ez a fa, akkor mindenképpen létre kell hozunk egy ugró élt. Ezekhez az extrém csúcsokhoz mindig eltároljuk az aktuális részfa gyökerétől vett távolságot. Így az ugró él relatív koordinátáit könnyen kiszámíthatjuk, hiszen az ugró él végpontjának már tudjuk a relatív távolságát a saját részfájának gyökerétől (mert az előző rekurzív lépésben már meghatároztuk azt), és az aktuálisan vizsgált jobb és bal oldali részfa gyökerei közötti távolságot is tudjuk.

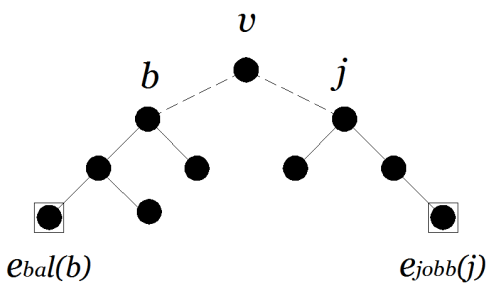
Az extrém csúcsok kiszámítása hasonlóan történik, mint az 2.1.4. algoritmus. A T_v-hez tartozó extrém csúcsokat jelöljük $e_{bal}(v)$-vel és $e_{jobb}(v)$-vel, az ezek abszolút távolságát a v gyökértől pedig rendre $abs_{bal}(v)$-vel és $abs_{jobb}(v)$-vel. A v távolságát a gyermekeitől, mivel szimmetrikus, ezért jelölhetjük $o(v)$-vel. Az egyszerűség kedvéért v gyermekeit jelöljük b-vel és j-vel.

2.1.5. Algoritmus (Extrém csúcsok kiszámítása). *Ha T_v egyetlen csúcsból áll, $e_{bal}(v) = e_{jobb}(v) = v$ és $abs_{bal}(v) = abs_{bal}(v) = 0$*

Első eset: *$T_{bal}(v)$ és $T_{jobb}(v)$ magassága ugyanannyi. Ekkor $e_{bal}(v) = e_{bal}(b)$ és $e_{jobb}(v) = e_{jobb}(j)$, $abs_{jobb}(v) = abs_{jobb}(j) + o(v)$, hasonlóan $abs_{bal}(v) = abs_{bal}(b) + o(v)$. (lásd 2.5. ábra)*

Második eset: *$T_{bal}(v)$ és $T_{jobb}(v)$ magassága nem egyezik. Legyen $T_{bal}(v)$ magassága kisebb mint $T_{jobb}(v)$ magassága. Ekkor $e_{bal}(v) = e_{bal}(j)$ és $e_{jobb}(v) = e_{jobb}(j)$, $abs_{jobb}(v) = abs_{jobb}(j) + o(v)$, $abs_{bal}(v) = abs_{bal}(j) + o(v)$*

Az algoritmus tehát a következő lépéseket teszi egy v csúcs vizsgálatakor: végig megy a v bal részfájának a jobb kontúrján és a jobb részfájának bal kontúrján. Minden iterációs lépés végén csak szülőből gyerekre kell átváltania, legyen az a gyermek egy ugró csúcsból érkező gyermek, vagy valódi gyermek. Ezek alapján kiszámolja az abszolút távolságokat a részfák gyökerétől, valamint a két részfa aktuális



2.5. ábra. Extrém csúcs kiszámítása különböző magasságú részfák esetén.

szintjén a bal és jobb részfa közötti távolságot, majd ha túl kevés ez a távolság, akkor növeli a széthúzandó távolságot. A ciklus végeztével beállítja $o(v)$-t, amely a széthúzandó távolság fele lesz. Majd, ha az egyik részfa legalsó szintjéig eljutottunk, akkor ha kell, beállítja az ugró élt. (Ez akkor van, ha a másik fában még van ez alatt a szint alatt csúcs.) Végül frissíti a T_v-hez tartozó extrém csúcsokat.

2.1.2. Elemzés

2.1.6. Állítás. *A szintezett lerajzolás algoritmusa $O(n)$ időben legfeljebb n^2 területű rajzot ad.*

Bizonyítás. A rajz sorában van legalább egy csúcs, tehát a magassága sem lehet nagyobb n-nél. A rajz szélessége $n-1$. amelyet a részfákra alkalmazott indukcióval beláthatjuk. Tehát a terület legfeljebb n^2.

A lépésszám az algoritmus három szakaszának lépésszámából tevődik össze. Először a relatív koordinátákat határozzuk meg, erről mindjárt belátjuk, hogy lineáris idejű. Ennek előfeltétele volt, hogy ismerjük minden gyökeres részfa magasságát, mely egy postorder bejárással szintén lineáris időben megtehető. Végül az abszolút koordináták kiszámítása a relatív koordinátákból könnyen elkészíthető a fa egy preorder bejárásával lineáris időben. Tehát csak a relatív távolságok kiszámításáról kell belátni, hogy lineáris időben megtehető és készen vagyunk. A relatív távolságok kiszámításakor egy v csúcs vizsgálatánál elegendő csak addig vizsgálni a távolságkülönbségeket, ameddig a két gyermek részfája közül az alacsonyabbik részfa legalsó szintjéig eljutunk. Tehát a v csúcs feldolgozási ideje arányos a bal és jobb részfájának magasságainak minimumával.

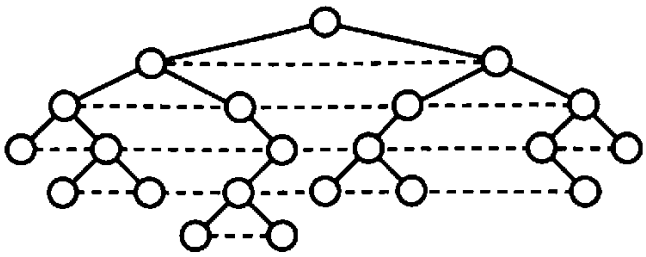
Emlékeztetünk, hogy a $T_{bal}(v)$ magasságát $h(v_{bal})$-al, a $T_{jobb}(v)$ magasságát $h(v_{jobb})$-al jelöljük. Az előzőek alapján igaz a következő becslés a postorder bejárás lépésszámára:

$$\sum_{v\in T}\Big(1+\min\big\{h(v_{bal}),h(v_{jobb})\big\}\Big)=n+\sum_{v\in T}\min\big\{h(v_{bal}),h(v_{jobb})\big\}$$

A $\sum_{v\in T}\min\{h(v_{bal}),h(v_{jobb})\}$ szumma kiszámításához húzzunk be új éleket minden azonos szinten lévő szomszédos két csúcs közé (lásd 2.6. ábra). Ezeknek az éleknek a száma éppen a szumma, hiszen ha egy v csúcsnál szétvágjuk a gráfot a $T_{bal}(v)$ és $T_{jobb}(v)$ részfára, akkor a $\min\{h(v_{bal}),h(v_{jobb})\}$ érték éppen a két részfa között futó új, vízszintes éleknek a száma. Az új élek pedig legfeljebb csúcsszámnyian vannak. □

2.2. Gyökeres fák szintezett lerajzolása

Az algoritmust könnyedén átvihetjük bináris fákról tetszőleges gyökeres fákra is.



2.6. ábra. A vízszintes élek mind pontosan egyszer szerepelnek az algoritmus egy összeillesztési lépésében [35].

2.2.1. Algoritmus (Szintezett lerajzolás algoritmus). ***Input:*** *T, r gyökerű rendezett fa.*

Output: *T Szintezett lerajzolása.*

1. *Ha a gráf egy csúcsból áll, akkor annak a rajza triviális.*
2. *Tegyük fel, hogy a T-t a gyökerénél szétvágva T_1, T_2, ..., T_k részfára esik szét. Rekurzívan alkalmazzuk az algoritmust ezekre a részfákra.*
3. *$i = 1, ..., k$-ig tegyük T_i-től vízszintesen jobbra 2 távolságra T_{i+1}-et úgy, hogy a gyökerüknek ugyanaz legyen az y koordinátája. Végül tegyük a részfák szülőjét a T_1 és T_k gyökereinek x koordinátáinak átlagába, egy egységgel magasabbra tőlük.*

A kontúrok és az extrém csúcsok kiszámítása hasonlóan tehető meg, mint a bináris fák esetében. Egy k fokú csúcs esetében $k-1$-szer végezzük el a csúcs részfáinak széthúzásának kiszámítását, hiszen bármely két szomszédos részfát össze kell hasonlítani egymással.

Az előzőek alapján belátható, hogy:

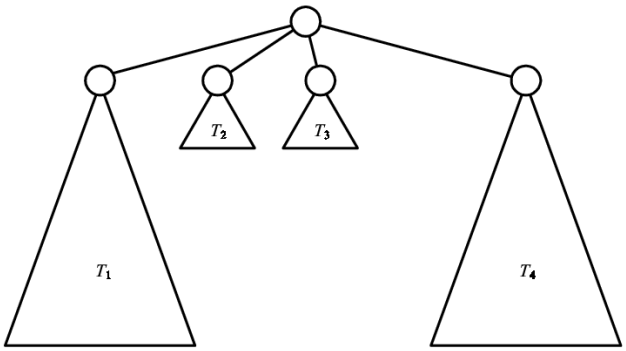
2.2.2. Tétel. *Legyen T egy r gyökerű rendezett fa n csúccsal. A szintezett lerajzolás algoritmusa $O(n)$ időben egy olyan lerajzolást ad, mely:*

1. *szintezett,*
2. *keresztezés mentes,*
3. *megőrzi az élek rendezettségét,*
4. *bármely két csúcs vízszintes és függőleges távolsága legalább egy,*
5. *területe $O(n^2)$,*
6. *Egy v csúcs x koordinátája a gyermekeinek x koordinátájának az átlaga.*

2.2.3. Megjegyzés. *A probléma ezzel a lerajzolással az, hogy nem egyenletesen osztja el a gyermekek részfáit, azaz lesznek részfák, melyek sokkal közelebb vannak az egyik szomszédjukhoz, mint a másikhoz (lásd 2.7. ábra). Ez általában nem mondható szép lerajzolásnak, mert szimmetrikus fák lerajzolásai nem lesznek egymás tükörképei.*

2.3. További eredmények

Az itt ismertetett algoritmusokat bináris és m-áris fákra Reingold és Tilford dolgozta ki [30]. (**m-áris fa** alatt egy olyan gyökeres fát értünk, melyben minden pont kifoka legfeljebb m.) Az algoritmusok alapja a Wetherell és Shannon módszere bináris fák lerajzolására [7]. Wetherellék algoritmusa teljesíti



2.7. ábra. Kiegyenlítelen ábrát ad a szintezet lerajzolási algoritmus [1].

2.1.2. állítás (1-6) tulajdonságait, viszont (7)-et nem. Azaz egybevágó részfáknak nem feltétlenül lesz egybevágó rajza, és szimmetrikus fák rajzai sem lesznek mindig szimmetrikusak, pedig ezt sok alkalmazás elvárja. Reingold és Tilford fenti algoritmusa nem csak ebben tekinthető szebb lerajzolásnak, de sokszor sokkal keskenyebb ábrát ad, mint a Wetherell-Shannon-féle.

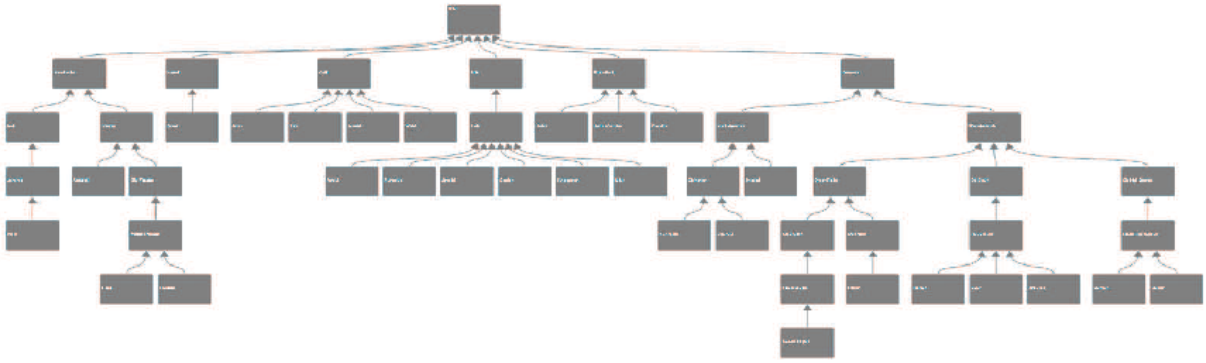
Gyökeres fák esetében a 2.2.3. megjegyzésben felvetett kiegyensúlyozottsági problémára többen is találtak megoldást. Walker az m-áris fákra olyan algoritmust adott, mely megőrzi a 2.2.2. tétel tulajdonságait és még azt is, hogy izomorf részgráfokra izomorf rajzot ad, és ha egy részfának a "tükörképét" vesszük, akkor azok rajzai egymás tükörképei lesznek [36]. Ez az algoritmus azonban négyzetes futásidejű. Buchheim és Liepert [6]-ben egy lineáris futásidejű algoritmust ad, mely minden gráfra pontosan ugyanazt a lerajzolást adja mint Walker [36]-ben publikált módszere ad. Bloesh szintén a kiegyensúlyottsági a problémára keresett megoldást és két algoritmust is kidolgozott, melyek futásideje $O(hn)$, ahol h a fa magassága [5].

Mariott és Sbarsky 2007-es cikkjében a korábbiakhoz képest már elveti a 2.2.2. tétel (6)-os tulajdonságát, annak érdekében, hogy keskenyebb rajzot kapjon [27]. Kvadratikus programozást használva minimalizálják a szülő és a gyermek csúcsok vízszintes távolságának az összegét. Összehasonlításképpen lásd a 2.8. ábrát Walker algoritmusával és 2.9. ábrát Mariott és Sbarsky algoritmusával lerajzolva ugyanazt a gráfot.

Végül egy érdekes eredményt bizonyított Mariott és Stuckey: bináris, nem rendezett fák esetében annak megállapítása, hogy létezik-e legfeljebb W szélességű lerajzolás, mely a 2.1.2. (1-4) és (6-7) tulajdonságait teljesíti egy NP teljes feladat [28].

2.4. Alkalmazások

Talán az egyik legelterjedtebb lerajzolási módszer a szintezett megközelítés. Családfák, evolúciós diagramok, organigramok, döntési fák, ciklus nélküli folyamatábrák és függvény hívások diagramjának lerajzolására használják, valamint gráfok szélességi keresési fáját is így érdemes átláthatóan lerajzolni.



2.8. ábra. Példa Walker lerajzolására [27].

2.9. ábra. Példa Maroitt és Sbarsky algoritmusára [27].

3. fejezet

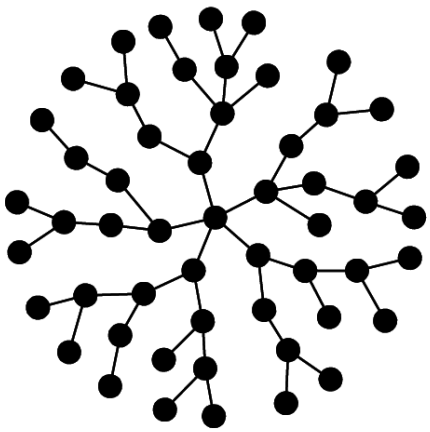
Sugárirányú lerajzolás

A sugárirányú lerajzolás egy bizonyos értelemben egy általánosítása a szintezett lerajzolásnak. Most is az a célunk, hogy könnyen meg tudjuk állapítani az ábráról, hogy mely csúcsok vannak a gyökértől azonos távolságban. Itt gyökér az origóba kerül, az azonos szintű csúcsok pedig koncentrikus körökön helyezkednek el körülötte. Több ilyen tulajdonságú lerajzolás létezik, ebben a fejezetben a [35] könyvben ismertetett változatot mutatjuk be.

3.1. Tulajdonságok

3.1.1. Állítás. *Legyen T egy r gyökerű gyökeres fa. A sugárirányú lerajzolásra igazak a következők:*

1. *keresztezés mentes,*
2. *egyenesvonalas rajz,*
3. *a csúcsok a gyökérpont körül egy-egy koncentrikus körön helyezkednek el úgy, hogy az i szintű csúcsok az i. körre kerülnek.*

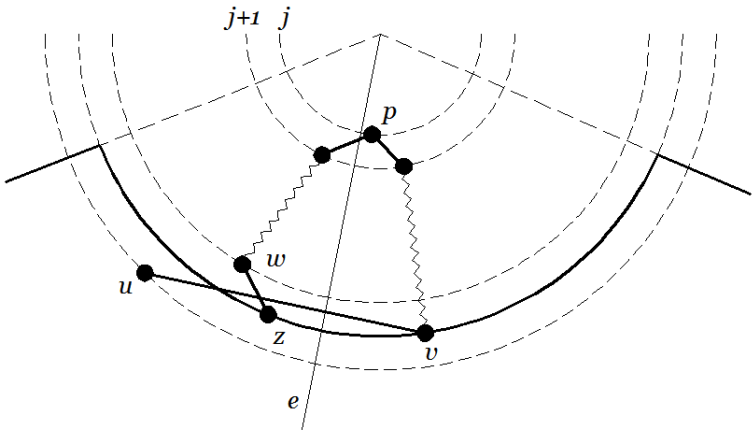


3.1. ábra. Példa sugárirányú rajzra [1].

3.2. Algoritmus

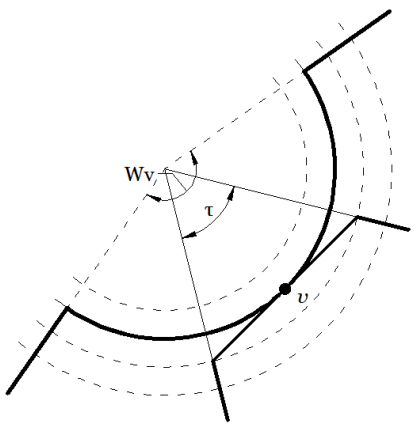
Legyenek $C_1, C_2, ..., C_{h(T)}$ origó körüli koncentrikus körök úgy, hogy az i. kör sugara $r_i = i \cdot r_1$ és minden i mélységű csúcs C_i-re fog kerülni. Minden v csúcsra v részfája egy W_v körgyűrűszeleten belül

fog elhelyezkedni, melynek határoló körei C_i és $C_{h(T)}$. v a W_v középponti szögének szögfelezője és a C_i kör metszéspontjába kerül. Első gondolatunk, hogy W_v-hez tartozó középponti szöget a részfa $l(v)$ leveleinek számával arányosan határozzuk meg. Meg kell vizsgálni, hogy ekkor lehet-e kereszteződés a rajzban. Tekintsük az 3.2. ábrát, ahol a körgyűrűszeletből kilépő élet vu élet keresztezi egy wz él. Ha w nem őse v-nek akkor legyen a legközelebb lévő közös ősük, p a C_i körön. Ennek lesz két leszármazottja C_{i+1}-en, amiket külön szögtartományba helyez el az algoritmus, melyek között van egy e határoló egyenes és az ő leszármazottjaik szögtartománya is diszjunkt lesz emiatt. Ezért az u csúcs nem lehet ugyanabban a szögtartományban mint a v csúcs. Akkor azonban, ha a keresztező él felsőbb végpontja szülője a v-nek, már lehet mutatni példát kereszteződésre.



3.2. ábra. Kiderül, hogy ebben az esetben nem lehetséges élkeresztezés.

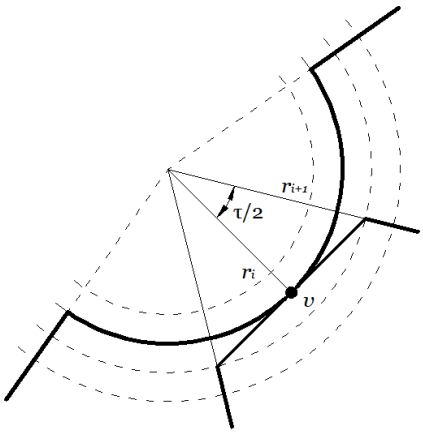
Ha a keresztező él kezdőcsúcsa őse a W_v-ből kilépő él kezdőcsúcsának, akkor még lehet kereszteződés a gráfban, ezért még egy megszorítást tesznek. Húzzunk v-ben egy érintőt C_i körhöz, és tekintsük a C_{i+1}-el vett metszéspontjait, a-t és b-t. Legyen az origóból a-n és b-n keresztül húzott egyenesek és az ab szakasz által határolt konvex terület F_v (lásd 3.3. ábra). Ha v minden gyereke ezen a területen belül helyezkedik el, akkor nem lesznek kereszteződések a rajzban.



3.3. ábra. F_v-hez tartozó szög.

Mivel a W_v körgyűrűszeletet lehet, hogy szűkebb halmaz, mint F_v, ezért valójában azt a körgyűrűszeletet kell a v részfájához rendelni, amelyiknek a középponti szöge az origóból az a és b pontokba húzott egyenesek által bezárt τ_v szögnek és W_v középponti szögnek a minimuma (lásd 3.3 ábra). Ezt a

szögtartományt kell felosztani v gyerekei között. Ehhez már csak a τ szöget kell kiszámítani, amelyről könnyen látható, hogy $\cos(\frac{\tau}{2}) = \frac{r_i}{r_{i+1}}$ (3.4 ábra).



3.4. ábra. F_v-hez tartozó szög kiszámítása.

Az algoritmus tehát a következő. Először egy postorder bejárással meghatározzuk a fa minden v csúcsára, hogy mekkora az ő részfájának leveleinek $l(v)$ száma. Majd az origóba letesszük a gyökeret és szintenként meghatározzuk a csúcsok poláris koordinátáit. Jelöljük β_v-vel a W_v szögtartomány középponti szögét, és γ_v-vel az origóból a v-be húzott egyenes és x tengely által bezárt szöget. Minden csúcshoz eltároljuk a β_v, γ_v és τ_v értékeket, melyeket egy preorder bejárás során határozunk meg. Ha egy u csúcs gyermeke v-nek, akkor az ő szögei az alábbi egyenletek alapján határozzuk meg.

$$\alpha_v := \min(\tau_v, \beta_v)$$

$$\beta_u = \frac{l(u)}{l(v)} \cdot \alpha_v$$

γ_u-t pedig úgy határozhatjuk meg, ha lerögzítjük a v gyermekeinek egy sorrendjét, azaz legyenek v gyermekei $u_1, u_2, ..., u_k$, és legyen $u = u_i$. Ekkor:

$$\gamma_u = \gamma_v - \frac{\alpha_v}{2} + \sum_{j=1}^{i-1} \beta_{u_j} + \frac{\beta_u}{2}$$

Ebből már meghatározható minden csúcsnak a poláris koordinátája: $\rho(v) = (r_i, \gamma_v)$, ahol a v szintje éppen i.

3.2.1. Elemzés

3.2.1. Állítás. *Egy T r gyökerű gyökeres fára a sugárirányú lerajzolás algoritmusa $O(n)$ idejű.*

Bizonyítás. Az eljárás egy preorder bejárással megtehető, mivel minden lépésben meghatároztuk a vizsgált csúcs poláris koordinátáit, így a futásidő lineáris □

A terület becslésére pontos bizonyítást ehhez az algoritmushoz nem találtunk. A rajz területe alatt a két legközelebbi csúcs távolságának és a rajz befoglaló körének arányát értjük. Jelen esetben a befoglaló kör éppen a legutolsó szinthez tartozó $C_{h(T)}$ kör.

A [35] könyv állítása szerint a sugárirányú lerajzolás területe $O(h(T)^2 \cdot \Delta^2)$, ahol Δ a legnagyobb kifokú csúcs fokszáma, és h a fa magassága. Ez azonban nem igaz, könnyen mutathatunk rá ellenpéldát. Legyen T egy teljes bináris fa, azaz olyan gyökeres fa, melyben minden a levelek kivételével csúcsnak pontosan két gyermeke van. Ekkor $\Delta = 2$, így [35] szerint a rajz területe $O(2 \cdot h(T)^2)$ Viszont a lerajzolásra az igaz, hogy az i. szinthez tartozó körön 2^i csúcs van, melyek legalább egy távolságra vannak egymástól. Így a C_i kör k_i kerületét alulról becsülhetjük 2^i-vel, és mivel $4r_i < k_i$, ezért $r_i < 2^{i-2}$. A $C_{h(T)}$ kör területe tehát $O(2^{h(T)})$, ami ellentmondás.

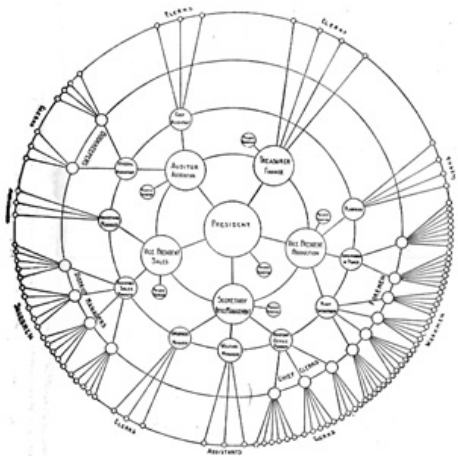
Az nyilván látszik, hogy fa magasságától függ a rajz területe, így irányítatlan esetben, fákra javíthatunk a területen, ha úgy választjuk ki a gyökérpontot, hogy minimalizáljuk a fa magasságát. Ezt $O(n)$ időben megtehetjük a következő algoritmussal. Ha a fának legfeljebb két csúcsa van, akkor végeztünk, ha nem, akkor távolítsuk el a fa leveleit. Végül vagy egy pont marad, vagy egy él. Ez utóbbi esetben úgy módosíthatjuk az algoritmust, hogy egy 1 hosszú egyenessel összekötjük a két csúcsot és ennek a közepét helyezzük az origóra.

3.3. További eredmények

A fent bemutatott algoritmust Eades publikálta [15], de korábban Bernard is foglalkozott ezzel a témával [4]. Sok sugáriányú lerajzolási módot lehet még találni fákra, melyek a gyökér megválasztásában, a körök sugarában és a körgyűrűszelet méretében különböznek egymástól. Book és Keshary [2] cikkében például az egyes csúcsokhoz tartozó szögtartományokat a csúcs gyermekeinek számával arányosan határozzák meg. Ez általában nagyobb területű rajzot ad, mint az ebben a fejezetben ismertetett algoritmus. Bernard és Mohammed olyan algoritmust dolgozott ki, mely a csúcsokat címkézettnek tekinti, azaz minden csúcshoz tartozik egy téglalap, és az így felépített gráfot kell sugárirányú lerajzolással, síkban lerajzolni [3]. Csak olyan nem bináris fákra ad lerajzolást, melyekben minden levél a fa azonos szintjén szerepel. Az algoritmus nem a gyökértől indulva határozza meg a csúcsok koordinátáit, hanem először a levelek szintjétől, majd minden csúcsot a "legbaloldaliabb" és "legjobboldalibb" gyermekébe mutató sugarak szögfelezőjében helyez el.

3.4. Alkalmazások

A sugárirányú lerajzolást már nagyon régóta használják intuitív módon (lásd 3.5. ábra). Ma a módszert elsősorban az elmetérképek lerajzolásához használják, és számos elmetérkép rajzoló program alkalmazza, mint például Mindmanager, MindMapper vagy E-Draw. Másik fő irány a web fejlődésének vizulalizálása, ahol a elsősorban nem fákra, hanem tetszőleges gráfokra használnak sugárirányú lerajzolást ([11], [10]).

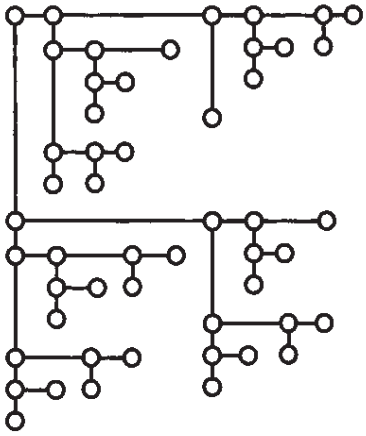


3.5. ábra. Organigram 1924-ből [33].

4. fejezet

HV Lerajzolás

A következő két fejezetben két olyan rácsrajzot ismertetünk, melyekben minden élet vagy vízszintes, vagy függőleges egyenes vonallal ábrázolunk. Sok esetben ez nem ad jól átlátható ábrát, mert nem érzékelteti olyan jól rajta a fa szerkezetét, mint a szintezett vagy a sugárirányú lerajzolások. Ha viszont nem az átláthatóság az elsődleges célunk, hanem az, hogy a rajz területe minél kisebb legyen, akkor ezek a lerajzolások nagyon jól alkalmazhatóak. Ugyanis ezek a módszerek $O(n \log n)$-es területű ábrát adnak, amely előző két fejezetben ismertetett polinomiális területű lerajzolásoknál sokkal jobb. Ebben a fejezetben a *HV* lerajzolást mutatjuk be a [35]-ben leírtak alapján.



4.1. ábra. Példa HV lerajzolásra [35].

4.1. Tulajdonságok

4.1.1. Állítás. *Legyen T egy rendezett bináris fenyő n csúccsal. A HV lerajzolás algoritmusa egy olyan rajzot ad, amely:*

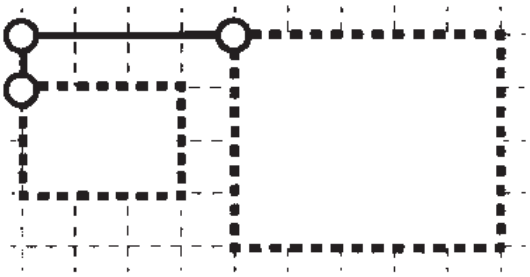
1. *rácsrajz,*
2. *keresztezés mentes,*
3. *ortogonális,*
4. *ereszkedő,*
5. *minden csúcs vízszintesen, vagy függőlegesen egy vonalban van a szülőjével,*

6. *ha két csúcs részfája diszjunkt, akkor a részfáikat tartalmazó* ***befoglaló téglapok*** *(legkisebb téglalap ami tartalmazza részfa rajzát) nem metszik egymást.*

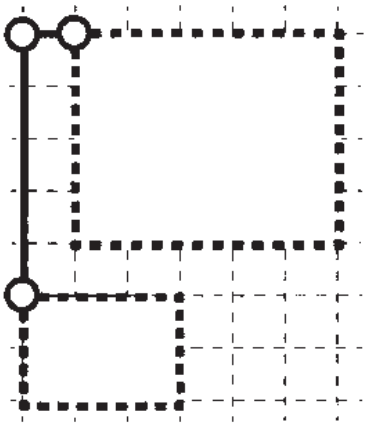
Emlékeztetünk, hogy rácsrajz esetében, ezért egy rajz területe alatt a rács méretét egységnek tekintve a rajz befoglaló téglalapjának területét értjük.

4.2. Algoritmus

A most következő algoritmus is az "oszd meg és uralkodj" elven alapszik, hasonlóan mint a szintezett lerajzolás. Ha két részfát már megrajzoltunk, akkor azokat vízszintes vagy függőleges elrendezésnek megfelelően egymáshoz illesztjük az 4.2. és a 4.3. ábráknak megfelelően.



4.2. ábra. Viszszintes elrendezés [35].



4.3. ábra. Függőleges elrendezés [35].

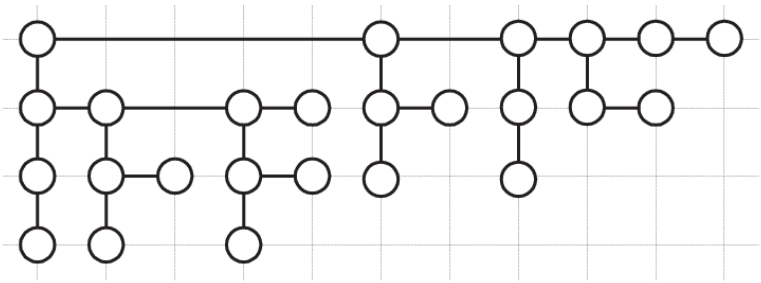
4.2.1. Megjegyzés. *A HV lerajzolásban minden sorban és oszlopban van egy csúcs, így mind a rajz magassága, mind a szélessége nem lehet nagyobb, mint* $n-1$.

4.2.2. Algoritmus (Jobb-nehéz HV lerajzolás algoritmus bináris fákra)**.** ***Input:*** *T, r gyökerű bináris fa.*

Otuput: *T HV lerajzolása.*

1. *Ha a gráf egy csúcsból áll, akkor annak a rajza triviális.*
2. *"Oszd meg:" Rekurzívan alkalmazzuk az algoritmust a T jobb és bal oldali részfáira.*

3. *"Uralkodj:" Helyezzük a vízszintes elrendezés szerint a több csúcsból álló részfát a másik jobb oldalára. Ha T-nek csak egy gyermek részfája van, akkor azt helyezzük egy egységgel jobbra a T gyökerétől.*



4.4. ábra. Példa jobb-nehéz HV lerajzolásra [1].

4.2.1. Implementáció

Most precízen is megadjuk, hogy mit jelent a vízszintes elrendezése a csúcsoknak.

Mint a szintezett lerajzolásnál, itt is először egy postorder bejárással a csúcsok szülőhöz vett relatív koordinátáit adjuk meg, majd egy preorder bejárással megadjuk az abszolút koordinátákat.

A postorder bejárásnál egy csúcsról eltároljuk az ő részfájának befoglaló téglalapjának magasságát és szélességét, $H(v)$-t és $W(v)$-t, valamint a relatív koordinátáit, $relX(v)$-t és $relY(v)$-t. A jobb és bal gyerekeket v_{jobb}-el és v_{bal}-el jelöljük. Egy v csúcs részfájának a méretét pedig $N(v)$-vel jelöljük, melyet szintén a postorder bejárás során számítunk ki. Ha v levél, akkor $H(v) = 0$, $W(v) = 0$, $relX(v) = 0$, $relY(v) = 0$ és $N(v) = 1$

Amikor egy olyan v csúcsot vizsgálunk, melynek két gyermeke van, akkor ezeket az értékeket a következőképpen határozzuk meg:

---

if $N(v_{jobb}) \leq N(v_{bal})$ **then**
- $relX(v_{bal}) := W(v_{jobb} + 1)$
- $relY(v_{bal}) := 0$
- $relX(v_{jobb}) := 0$
- $relY(v_{jobb}) := -1$
- $H(v) := H(v_{jobb})$
- $W(v) := W(v_{bal}) + W(v_{jobb}) + 1$
- $N(v) := N(v_{bal}) + N(v_{jobb})$

end

4.2.2. Elemzés

4.2.3. Állítás. *Legyen T egy bináris fenyő. A HV lerajzolás rajzának magassága $O(\log n)$.*

Bizonyítás: A teljes rajz magassága legfeljebb n. Legyen a rajzban a legmélyebb csúcs w. Mivel csak vízszintes elrendezést alkalmaztunk, ezért minden függőleges él egység hosszú. Így elég megmondani,

hogy hány él van a a w-től a legfelső szintig menő úton. Legyen uv ezen úton. Amikor az u alatti két részfát összeillesztettük, akkor a nagyobb csúcsszámút helyeztük a kisebb, v gyökerű részfa mellé. Így a u részfájában legalább a kétszer annyi pont van, mint a v részfájában. Mivel minden éllel kétszereződik a részrajz csúcsszáma, ezért nem lehet több él, mint $\log n$.

4.2.4. Állítás. *A HV lerajzolás bináris fenyőkre egy $O(n\log(n))$ méretű rajzot ad $O(n)$ időben.*

Bizonyítás: Az előző állítás és a 4.2.1. megjegyzés alapján igaz a terület becslés. A lépésszám is igaz, hiszen minden csúcsnál az összeillesztésnél csak a két gyermek relatív koordinátáit adjuk meg, azaz konstans sok lépést végzünk. A második preorder bejárás is n nagyságrendű, így összesen $O(n)$ lépést végzünk.

4.3. Kiterjesztés gyökeres fákra

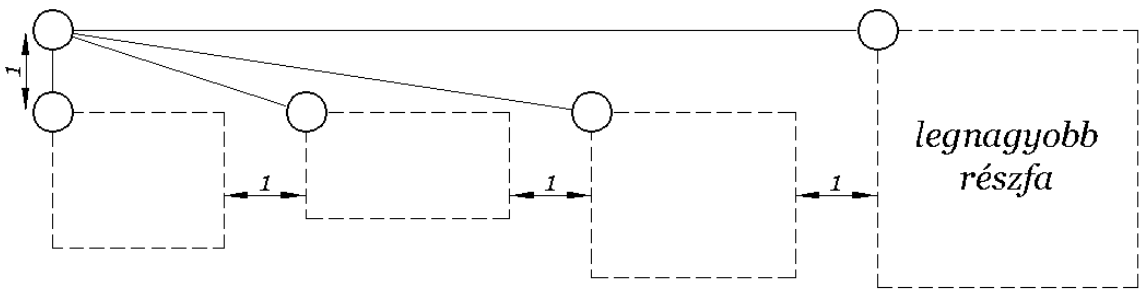
A HV lerajzolás algoritmusa tetszőleges gyökeres fákra is általánosítható.

4.3.1. Algoritmus (Jobb nehéz HV lerajzolás gyökeres fákra). ***Input:*** *T, r gyökerű fa.*
Otuput: *T HV lerajzolása.*

1. *Ha a gráf egy csúcsból áll, akkor annak a rajza triviális.*
2. *"Oszd meg": Tegyük fel, hogy a T gyökerénél szétvágva T_1, T_2, ..., T_k részfára esik szét. Rekurzíven alkalmazzuk az algoritmust ezekre a részfákra.*
3. *"Uralkodj": Legyen T_k a legnagyobb csúcsszámú részfa. Tegyük $i = 1, ..., k$-ig jobbról balra T_i mellé T_{i+1}-et vízszintesen 1 távolságra, az 4.5. ábra alapján.*

A pontos koordináták kiszámításához legyen az i. fa szélessége $W(i)$, és legyen a T_i gyökere v_i. Ekkor a v_i relatív koordinátáit a szülőhöz képest a következőképpen határozhatjuk meg:

$$\begin{aligned} relX(v_i) &= \sum_{j=1}^{i} w(j) + j - 1, \qquad 1 \le j \le k-1 \\ relY(v_i) &= -1, \qquad 1 \le j \le k-1 \\ relX(v_k) &= \sum_{j=1}^{i} w(k) + k - 1, \\ relY(v_k) &= 0 \end{aligned}$$



4.5. ábra. A jobb-nehéz HV lerajzolás általánosításának egy összeillesztési lépése.

4.3.2. Állítás. *A HV lerajzolás $O(n)$ időben tetszőleges r gyökerű fenyőre olyan ábrát ad, mely:*

1. *egyenesvonalasrajz,*
2. *rácsrajz,*
3. *ereszkedő,*
4. *keresztezésmentes,*
5. *a rajz magassága legfeljebb* $O(\log n)$*, szélessége legfeljebb* $n-1$
6. *területe* $O(n \log n)$*.*

4.4. További eredmények

A HV lerajzolást először 1992-ben Crescenzi, Di Battista és Piperno dolgozta ki [12]. Publikációjukban a fent bemutatott jobb nehéz HV lerajzolás algoritmusához hasonlóan $O(n \log n)$-es területű rajzot készítettek lineáris időben bináris fákra. Ezen kívül egy olyan algoritmust is bemutattak, mely teljes bináris fákra $O(n)$ területű rajzot ad. Később Crescenzi és Penna igazolta, hogy a HV lerajzolás minimális mérete h magasságú bináris fa esetén $2.5n - 4.5\frac{\sqrt{n+1}}{2} + 3.5$, ha h páros és $2.5n + \sqrt{n+1} + 3.5$, ha h páratlan, továbbá egy lineáris idejű algoritmust is adtak a lerajzolás megtalálásához [13]. Kim egy másik fontos célfüggvény, az összélhossz szerint is vizsgálta a HV lerajzolást, és bemutatott egy $n \log \log n$ összélhosszú rajzot adó algoritmust [21]. Crescenzi és Penna lerajzolásának az összélhossza $n \log n$, így Kim algoritmusa jelentős javítás, a területre viszont ez az algoritmus rosszabb, $O(n \log n \log \log n)$.

4.5. Alkalmazások

Bár a HV lerajzolás nem a legmegfelelőbb az olyan alkalmazáshoz, melyeknél a struktúra megértése a cél, mert kevéssé látható át a fa szerkezete, viszont a rajz területe jobb a korábban bemutatott algoritmusokhoz képest. Mivel az élek hosszainak összege ennél a lerajzolásnál az egyik legkisebb, ezért felosztható kapcsolási rajzok tervek elkészítéséhez használják, ahol a huzalok összhossza kardinális kérdés. Még egy érdekes alkalmazásáról olvashatunk [20]-ben a Lisp nevű programozási nyelven írt programok vizualizálásához.

5. fejezet

Tekercselt lerajzolás

A tekercselt lerajzolás bináris fákra készít ortogonális rácsrajzot, nagyságrendileg ugyanazzal a területtel mint az előző fejezetben tárgyalt HV lerajzolás. Ennek az algoritmusnak az a célja, hogy konstans képfelbontású rajzot kapjunk. Korábban említettünk, hogy ez a tulajdonság szintén fontos elvárás a legtöbb alkalmazás esetében, így a tekercselt lerajzolás a területe és a képfelbontása miatt is kiemelkedik a többi rácsrajzot adó algoritmus közül. Az algoritmust a [35] könyvben leírtak alapján ismertetjük.

5.1. Tulajdonságok

5.1.1. Állítás. *Legyen T egy rendezett bináris fenyő n csúccsal. A tekercselt lerajzolás algoritmusa egy olyan rajzot ad, amely:*

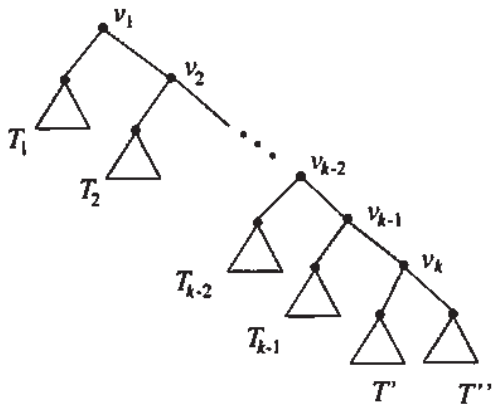
1. *rácsrajz*
2. *keresztezés mentes,*
3. *ortogonális,*
4. *ereszkedő,*
5. *minden csúcs vízszintesen, vagy függőlegesen egy vonalban van a szülőjével,*
6. *konstans képfelbontású.*

5.2. Algoritmus

Jelöljük T leveleinek számát l-el. Az általánosság megszorítása nélkül feltehető, hogy a fa teljes bináris fa, így $n = 2l - 1$. Ez a feltevés a későbbi terület becslések miatt kell, hiszen ekkor $O(n) = O(l)$, így minden becslést a levelek számának függvényében adhatunk meg. A rajzból végül kitörölhetjük a felesleges csúcsokat $O(n)$ időben, a terület ettől legfeljebb csak csökkenhet.

Emlékeztetünk, hogy T_v leveleinek számát $l(v)$-vel jelöljük. Rendezzük a fát úgy, hogy minden szinten a nagyobb levélszámú részfa mindig jobb oldalra kerüljön, azaz $l(v_{bal}) \leq l(v_{jobb})$. Ez lineáris időben megtehető.

Legyen $H(l)$ a rajz magassága és legyen $W(l)$ a szélessége, $t(l)$ pedig az algoritmus futásideje. Legyen $A > 1$ egy fix paraméter, melynek az értékét később határozzuk meg. Ha $l \leq A$, akkor a fát rajzoljuk meg a Jobb-nehéz HV lerajzolás 4.2. algoritmusa alapján. A HV lerajzolás becslése alapján $H(l) \leq \log_2 l$, $W(l) \leq A$ és $t(l) = O(A)$.



5.1. ábra. A nagyobb levélszámú csúcsokat jobbra rendezzük [35].

Tegyük fel, hogy $A < l$. Ekkor definiáljuk a $\{v_1, v_2, ..., v_{h(T)}\}$ csúcsúk sorozatát úgy, hogy v_1 legyen a gyökér, és $v_{i+1} = v_{i_{jobb}}$. Ekkor v_{i+1} jobb részfája része v_i jobb részfájának, így az $l(v_i)$ levelek száma szigorúan csökken. Legyen k az az index, amelyre $l(v_k) > l - A$, de $lv_{k+1} \leq l - A$. Ez a levelek számának szigorú csökkenése miatt $O(k)$ időben megtalálható. Legyen a v_i bal részfája T_i és legyen a T_i leveleinek száma l_i minden $i = 1, ..., k-1$-re. A v_k bal és jobb részfája pedig legyen rendre T' és T'', a leveleik száma pedig l' és l'' legyen (lásd:5.1. ábra). Ekkor a k választása miatt igazak az alábbiak:

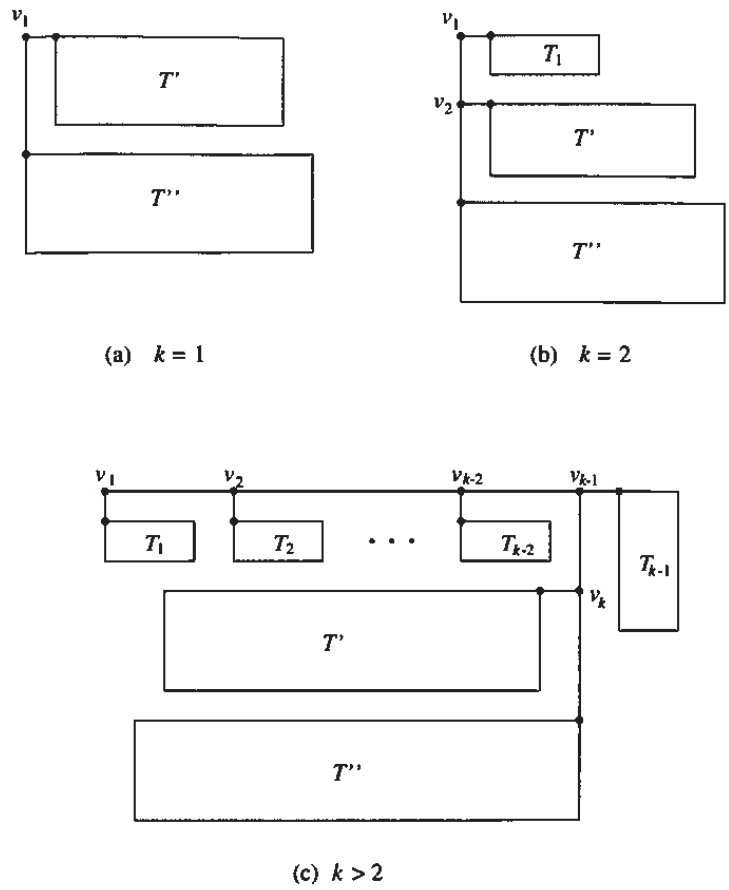
$$l_1 + ... + l_{k-1} = l - l(v_k) < A \tag{5.1}$$

$$\max\{l', l''\} = l(v_{k+1}) \leq l - A \tag{5.2}$$

Megjegyezzük, hogy $l' < l''$, mivel jobbra rendeztük a nagyobb levélszámú részfákat.

Most megadjuk az algoritmust (lásd 5.2 ábra).

1. Ha $k = 1$, akkor a v_1 alá T'-t és T''-t függőleges elrendezés szerint helyezzük el, miután rekurzíven lerajzoltuk őket. Mivel a rekurzív hívásnál a levelek száma csökkent, ezért van értelme újból összehasonlítani a levelek számát és A értékét.
2. Ha $k = 2$, T_1-et rajzoljuk meg a 4.2. algoritmus alapján, T'-t és T''-t rekurzíven rajzoljuk le. Ezután v_2 alá függőleges elrendezéssel helyezzük el T'-t és T''-t, majd ezt a rajzot és T_1-et szintén függőleges elrendezéssel v_1 alá.
3. Ha $k > 2$, akkor T_1, ..., T_{k-2}, T_{k-1}-t 4.2. algoritmus szerint rajzoljuk meg. Ezután v_1 mellé helyezzük sorban balról jobbra a T_i fák rajzait $i = 1, ..., k-2$-ig úgy, hogy a T_i gyökere v_i-vel függőlegesen egy vonalban legyen, és a v_i-k vízszintesen egy legyenek vonalban úgy, hogy a T_i ábrái között pontosan 1 legyen távolság. A T_{k-2} rajzát tükrözzük a rajz bal oldali határoló egyenesére, így a gyökér a rajz jobb felső sarkába kerül. Ezután forgassuk a rajzot el a gyökér körül $\frac{\pi}{2}$ szöggel pozitív irányba. Végül helyezzük el az így elkészült rajzot vízszintesen egyvonalban v_{k+1} mellé, v_{k-1}-től jobbra egy távolságnyira. A T'-t és a T''-t rekurzíven rajzoljuk meg, majd tükrözzük őket a rajzuk bal oldali határoló egyenesére, hogy a gyökerük az rajzuk jobb felső sarkába kerüljön. Helyezzük a rajzukat egymás alá egy távolságban úgy, hogy a gyökereik a v_{k-1}-el függőlegesen egyvonalban legyenek és a rajzuk a T_i fák rajzaitól függőlegesen legalább távolságra legyenek. (A legnagyobb magasságú T_i fa rajzától pontosan egy távolságban.)



5.2. ábra. Az összeillesztés esetei [35].

5.2.1. Elemzés

5.2.1. Tétel. *Legyen T bináris fa n csúccsal. Ekkor a tekercselt lerajzolás algoritmusa $O(n)$ időben olyan rácsrajzát adja T-nek, melynek területe $O(n \log n)$. A rajz magassága és szélessége egyaránt $O(\sqrt{n \log n})$, így a képfelbontás $O(1)$.*

Bizonyítás. A magasságokra, szélességekre és a futásidőre mindig igazak az alábbiak:

$$H(l) \leq \max\{H(l') + H(l'') + \log_2 A + 3, \log_2 l_{k-1} - 1\}$$

$$W(l) \leq \max\{W(l') + 1, W(l''), l_1 + l_2 + ... + l_{k-2}\} + \log_2 l_{k-1} - 1$$

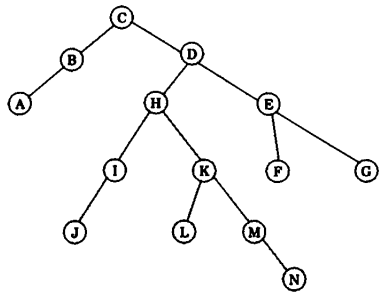
$$t(l) \leq \max\{t(l') + t(l'') + O(l_1 + l_2 + ... + l_{k-1} + 1)\}$$

A (5.1) egyenlőtlenség alapján:

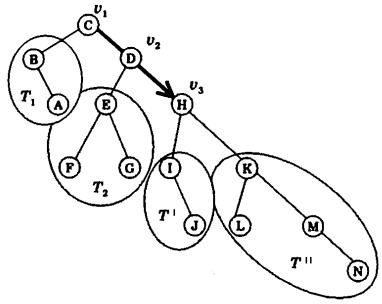
$$H(l) \leq \max\{H(l') + H(l'') + O(\log A), A\}$$

$$W(l) \leq \max\{W(l') + 1, W(l''), A\} + \log A$$

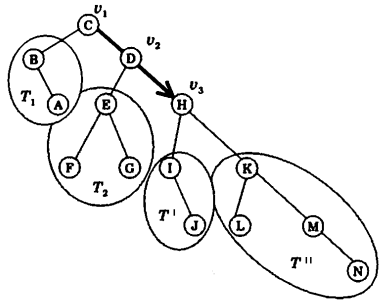
$$t(l) \leq \max\{t(l') + t(l'') + O(A)\}$$



(a) Input gráf.



(b) Felosztás az algoritmus szerint.



(c) Kész tekercselt lerajzolás.

5.3. ábra. Példa tekercselt lerajzolásra [35].

A (5.2) egyenlőtlenség alapján pedig látható, hogy $W(l) = O(\lceil \frac{l}{A} \rceil \log A + A)$. A továbbiakhoz belátjuk a következő lemmát.

5.2.2. Lemma. *Legyen $A > 1$ és f egy olyan függvény melyre:*

- *ha $l \leq A$, akkor $f(l) \leq 1$,*
- *ha $l > A$, akkor $f(l) \leq f(l') + f(l'') + 1$ bármilyen $l' \leq l'' \leq l - A$, melyre $l' + l'' \leq l$.*

Ekkor $f(l) < 4\frac{l}{A} - 1$ minden $l > A$-ra.

Bizonyítás. Indukció l-re. Tegyük fel, hogy l'-re és l''-re már igaz az állítás. Ha $l', l'' < A$, akkor $f(l) \leq 3 < 4\frac{l}{A} - 1$, hiszen $l > A$. Ha $l' \leq A$ és $l'' > A$, akkor $f(l) \leq 2 + f(l'') < 4\frac{l''}{A} + 1 \leq 4\frac{l-A}{A} + 1 < 4\frac{l}{A} - 1$. Ha pedig $l', l'' < A$, akkor $f(l) \leq f(l') + f(l'') + 1 \leq 4\frac{l'}{A} + 4\frac{l''}{A} - 1 \leq 4\frac{l'+l''}{A} - 1 = 4\frac{l}{A} - 1$. □

Mivel $l' + l'' \leq l$, ezért alkalmazhatjuk a lemmát $\frac{t(l)}{A}$-ra és azt kapjuk, hogy $t(l) = O(\lceil \frac{l}{A} \rceil A) = O(n)$. Ha $\frac{H(l)}{\log A}$-ra alkalmazzuk a lemmát, $H(l) = O(\lceil \frac{l}{A} \rceil \log A + A)$-t kapunk. Így $\frac{H(l)}{W(l)} = 1$, tehát a rajz

konstans képfelbontású. Végül, ha A-t $\sqrt{l \log l}$-nek választjuk, akkor megkapjuk, hogy a rajz területe $O(n \log n)$. $\square$

5.3. További eredmények

A fent leírt algoritmus fejlesztette tovább Chan, Goodrich Kosarajub és Tamassia [9]. A rajz területe és képfelbontása azonos az itt bemutatott lerajzolásával, ráadásul diszjunkt részfák befoglaló téglalapjai nem metszik egymást. Ezen kívül a tekercselt lerajzolás ötletével bebizonyítják, hogy bináris fákra létezik $O(n \log \log n)$-es területű nem ereszkedő, ortogonális rácsrajza. Ezt az eredményt tőlük függetlenül Shin és Kim is belátta, egy nagyon hasonló ötleten alapuló algoritmussal [32].

6. fejezet

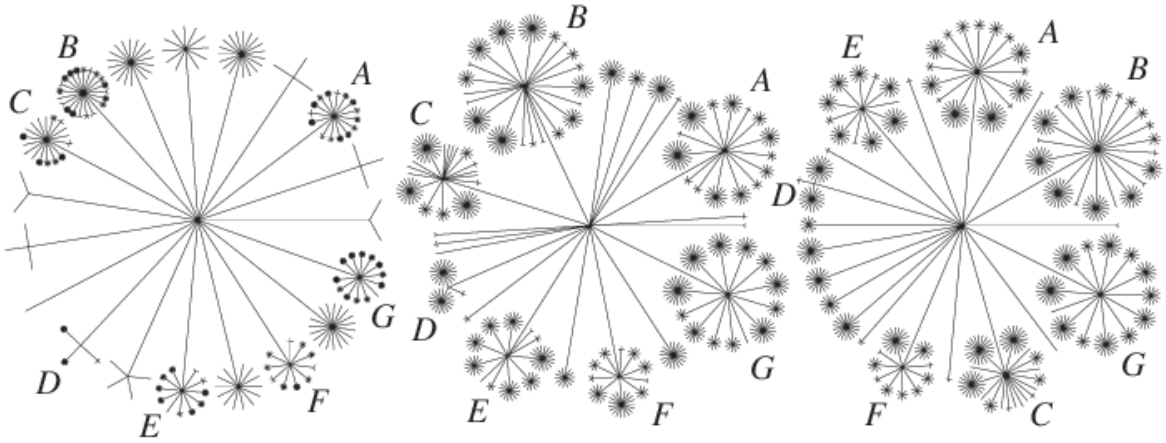
Buborék lerajzolás

A buborék lerajzolás olyan egyenesvonalas síkrajz, melynek alapgondolata, hogy egy csúcs gyermekei mind egy kör kerületén helyezkedjenek el, melynek a középpontjában a szülőjük áll. Minden részfa egy körlap belsejébe kerül, melynek a középpontjában a részfa gyökere áll. Ráadásul ezek a körlapok nem metszik egymást, ha a két részfa nem metszi egymást. Ez a két tulajdonság nagyon előnyös a fa struktúrájának megértése szempontjából, hiszen egyrészt jól látszik, hogy egy szülőnek mely csúcsok a gyermekei, másrészt a gyökeres részfák rajzai is jól elkülöníthetőek. Bár a buborék lerajzolás területe exponenciális lesz, mivel a gyökértől kiindulva egy úton minden él egyre rövidebb, az előző tulajdonságok miatt mégis széleskörű alkalmazási területe van, melyekről a fejezet végén esik szó.

A fentieken kívül még sok megszorítással élhetünk a buborék lerazolásra vonatkozólag. Definiáljuk egy v csúcs részfájához tartozó **befoglaló kört**, mely az a legkisebb kör, amely az ő gyermek részfáinak befoglaló köreit tartalmazza. Ennek a középpontjába kerül majd v. Egy v csúcs befoglaló köre v teljes részfájának rajzát tartalmazza. A levelek befoglaló köre egy pont, azaz nulla sugarú kör. A levelek szülőinek befoglaló körének sugarát egység hosszúnak definiáljuk, és a buborék lerajzolás területe alatt a befoglaló kör területét értjük. Élhetünk azzal a megszorítással, hogy az egy szinten lévő csúcsok befoglaló köreinek sugara azonos legyen, ekkor **fraktál modellről** beszélünk. Ez azonban gyakran ahhoz vezet, hogy a gyökértől távolodva nagyon kicsik lesznek az élek, ami nem ad jól átlátható ábrát. Ha ez a megkötéstől eltekintünk, azaz minden v csúcs gyermekéhez tartozó befoglaló kör különböző méretű lehet, akkor **SNS modellről** beszélünk (Subtrees with Non-uniform Size [26]). Ebben a fejezetben az SNS modellt mutatjuk be részletesen.

A buborék lerajzolás rekurzívan rajzol a levelektől a gyökérig. Csakúgy mint a korábbi algoritmusokban, először egy postorder bejárással a csúcsok relatív koordinátáit adjuk meg a szülő csúcshoz képest, majd egy preorder bejárással kiszámítjuk az abszolút koordinátákat. Minden v csúcs részfájának rajza egy W_v szögtartományba kerül, melynek a középpontjában a csúcs $p(v)$ szülője található. Ezt a W_i szögtartományt a **v-hez tartozó szögtartománynak** hívjuk. A $p(v)$-ből v-be vezető félegyenes két szögtartományra osztja W_v-t, melyeknek nem feltétlenül kell megegyezniük. Ha megköveteljük, hogy a lerajzolásban minden csúcsnál az utóbbi szögtartományok egyenlők legyenek, akkor **egyenlő szögű** (6.3. ábra), különben **nem egyenlő szögű buborék lerajzolásról** (6.11. ábra) beszélünk.

A W_v és v viszonyára kétféle megszorítást is tehetünk. Az egyik, hogy a v csúcs befoglaló körét teljes egészében tartalmazza a W_v szögtartomány, vagy elég csak a v gyermekeinek befoglaló köreit tartalmaznia. Az előbbiről a 6.1. alfejezetben, az utóbbira 6.4. alfejezetben lesz szó.



6.1. ábra. Buborék lerajzolások változata 1000 csúcsú fán (legnagyobb fokszám: 20, magasság 4). Jobboldalt a fraktál modell rajza, középen a buborék alap algoritmusával készült lerajzolás,(6.1. fejezet) baloldalt a szögfelbontás 6.2. fejezetben ismertetett javításával [26].

A fejezetben ismertetjük a buborék lerajzolás egyik változatának algoritmusát az SNS modellben [8] alapján. Majd három lépésben tesszük szebbé a lerajzolást a [26] cikkben leírtak szerint. Először az egyenlő szögű, nem rendezett fák esetében optimalizáljuk a szögfelbontást és a szögarányt egy kitüntetett műveletre nézve. Második lépésként jobbá tesszük az így elkészített rajz területét, de így nem egyenlő szögűvé tesszük a rajzot, valamint a szögarányt és szögfelbontást is rontjuk. Ezután javítjuk lerajzolás szögfelbontását és szögarányát. Végül abban az esetben adunk egy optimalizálási feladatot a rajz területére, amikor a buborék lerajzolás (2)-es tulajdonságát nem várjuk el, azaz a szülőtől a gyermek csúcsoknak nem feltelenül kell egyenlő távolságban elhelyezkedni a szülőjüktől.

6.1. Buborék lerajzolás egyenlő szögű esetben

6.1.1. Állítás. *Legyen T egy tetszőleges r gyökerű fa. A buborék lerajzolás algoritmusa egy olyan lerajzolást ad T-ről, melyre a következők igazak:*

1. *síkrajz,*
2. *egyenesvonalas rajz,*
3. *minden v csúcsra igaz, hogy a v gyermekei egy körön helyezkednek el melynek a középpontjában a szülő van,*
4. *ha két csúcs részfája diszjunkt, akkor a részfáikat tartalmazó befoglaló körök nem metszik egymást.*

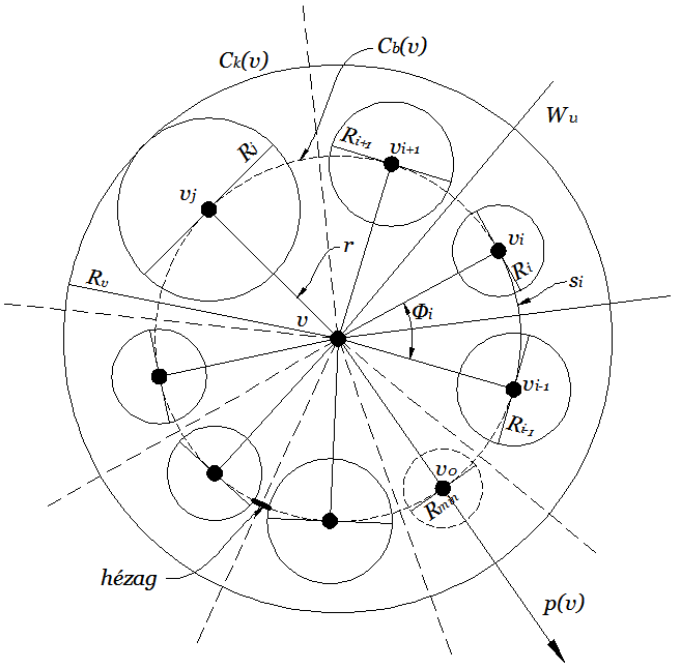
6.1.1. Algoritmus

A buborék lerajzolás rekurzívan rajzol a levelektől a gyökérig. Csakúgy mint a korábbi algoritmusokban, először egy postorder bejárással a csúcsok relatív koordinátáit adjuk meg a szülő csúcshoz képest, majd egy preorder bejárással kiszámítjuk az abszolút koordinátákat.

Legyen w egy tetszőleges csúcs. Jelöljük $C_b(w)$-vel a w **belső körét**, melyre a w gyermekei kerülnek, és jelöljük $C_k(w)$-vel a w befoglaló körét - azaz külső körét.

Tegyük fel, hogy egy v csúcs minden gyermekének már ismerjük a relatív koordinátáit a szülőhöz képest, és a hozzájuk tartozó befoglaló köröket is. Legyenek a v gyermekei $v_1, v_2, ..., v_n$. Jelöljük v_i csúcs-

hoz tartozó befoglaló $C_k(v_i)$ kör sugarát R_i-vel, és a v_i-hez tartozó szögtartományt pedig az egyszerűség kedvéért jelöljük W_i-vel. Annak érdekében, hogy v $p(v)$ szülőjét is el tudjuk helyezni, felveszünk egy v_0 csúcsot, R_{min} minimális sugárral. R_{min} lehet például a rajzban addig előforduló legkisebb befoglaló kör sugara. Így $p(v)$ majd a v-ből v_0-ba menő egyenesen fog elhelyezkedni a következő iterációban. A $C_b(v)$ belső kör sugara legyen r, kerülete C, a $C_k(v)$ befoglaló kör sugara pedig legyen R_v. Legyen ϕ_i az a középponti szög, melyet v_i és v_{i-1} határoz meg, és legyen az ehhez tartozó körív hossza $C_b(v)$-n s_i(lásd 6.8. ábra) Feladatunk, hogy r-et és ϕ_i-t meghatározzuk. A belső kör kerületét becsülhetjük a gyermek körök átmérőivel, $C \cong 2\sum_i R_i$, és így sugara $r = \frac{C}{2\pi}$ lesz. Az s_i körív $R_i + R_{i-1}$-el becsülhető, és így a középponti szögeket $\phi_i \cong \frac{s_i}{r}$-nek választhatjuk. Végül a v befoglaló körének, R_v-nek a sugara pontosan $r + R_i$ lesz, ahol R_i a legnagyobb sugár a gyerekek befoglaló körei közül.



6.2. ábra. Buborék lerajzolás algoritmusa.

Ezek a becslések azonban még nem pontosak, hiszen nem biztos, hogy így minden gyermek befoglaló köre valóban elfér a C_v körön. Minél kevesebb gyermek van, azaz minél kevesebb kör átmérőjével becsülünk, annál rosszabb ez a becslés. Így fel kell szorozni a C_v kör kerületét a v gyermekeinek számával fordítottan arányosan. Erről az átméretezésről azonban nem találni a szakirodalomban pontos leírást.

6.1.2. Megjegyzés. *A C kerület kiszámításakor a felszorzás után általában a $C_k(v_i)$ körök nem érintik W_i két szögszárát (lásd 6.8. ábra). A 6.3. alfejezetben látni fogjuk, hogy ezeket a "hézagokat" le lehet csökkenteni azért, hogy a rajz területe kisebb legyen.*

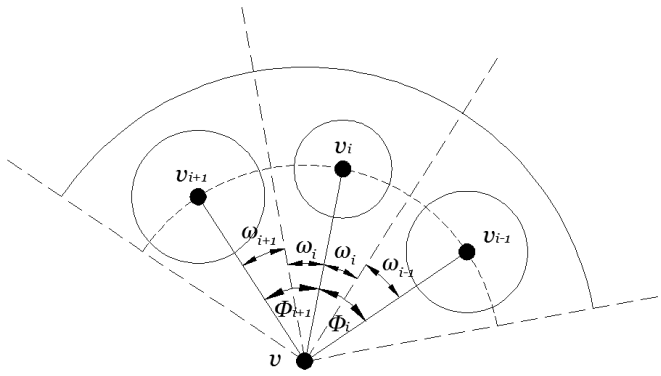
6.1.2. Szögtartományok kiszámítása

A [26] cikk nem tér ki a W_i szögtartományok szögeinek meghatározására, viszont a szögfelbontás és a szögarány javításához éppen ezen szögtartományok manipulálását veszi alapul. A feladat az tehát, hogy megadjuk a W_i szögtartományok θ_i szögét. Nem egyenlő szögű esetben egyszerű a helyzet, ugyanis tetszőlegesen behúzhatjuk W_i két szögszárát azzal a feltétellel, hogy a szomszédos v_{i-1}-hez és v_{i+1}-hez

tartozó befoglaló köröket ne messe. Egyenlő szögű esetben azonban már nehezebb a helyzet, ugyanis W_i-t az egyenlő szögű esetben felezi a vv_i félegyenes. Jelöljük ennek a két részszögtartománynak a szögét ω_i-vel. Ezeket kell meghatározunk a Φ_i szögek ismeretében. $\omega_i + \omega_{i-1} = \Phi_i$ (lásd 6.3 ábra), tehát a következő lineáris egyenletrendszert ki kell elégítenie az ω_i szögeknek:

$$\begin{pmatrix} 1 & 0 & \dots & & 0 & 1 \\ 1 & 1 & 0 & & \dots & 0 \\ 0 & 1 & 1 & 0 & \dots & 0 \\ \vdots & & \ddots & \ddots & & \\ 0 & \dots & & 0 & 1 & 1 \end{pmatrix} \begin{pmatrix} \omega_1 \\ \omega_2 \\ \vdots \\ \omega_{n-1} \\ \omega_n \end{pmatrix} = \begin{pmatrix} \Phi_1 \\ \Phi_2 \\ \vdots \\ \Phi_{n-1} \\ \Phi_n \end{pmatrix} \tag{6.1}$$
$$\omega_i \geq 0$$

A megoldás egyértelműségéhez meg kell vizsgálni a feltétel mátrix determinánsát.



6.3. ábra. Szögek meghatározása egyenlő szögű esetben.

6.1.3. Állítás. *A következő $n \times n$ mátrix determinánsa 0, ha n páros, és 2, ha n páratlan.*

$$\begin{vmatrix} 1 & 0 & \dots & & 0 & 1 \\ 1 & 1 & 0 & & \dots & 0 \\ 0 & 1 & 1 & 0 & \dots & 0 \\ \vdots & & \ddots & \ddots & & \\ 0 & \dots & & 0 & 1 & 1 \end{vmatrix}$$

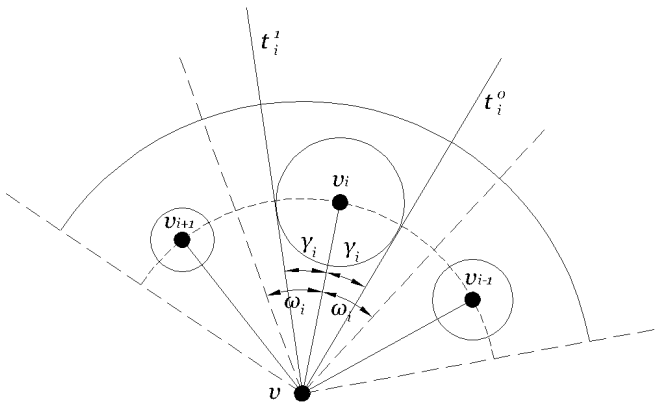
Bizonyítás. Azt állítjuk, hogy mátrixnak csak két nem nulla kifejtési tagja van, azaz olyan elemek szorzata, melyekből a mátrix minden sorában és minden oszlopában pontosan egy elem szerepel. Az egyik a főátlóbeli elemekhez tartozó tag, másik a főátló alatt lévő elemek és az első sor utolsó eleme. Ugyanis, ha az első oszlop első elemét választjuk, akkor a második sorban nem választhatjuk az első elemet, és csak a 2. elem nem nulla, úgyhogy azt fogjuk választani. Ekkor nyilván nem választhatjuk a 3. oszlopban a 2. elemet, tehát az egyetlen másik nem nulla elemet, a harmadikat fogjuk választani. A módszert folytatva megkapjuk a főátlóbeli elemeket. A másik esetben, ha az első oszlopban nem az első, hanem a második elemet választjuk, akkor a második oszlopban nem választhatjuk az első elemet, így a harmadikat fogjuk

választani. Az utolsó sorban az első elemet fogjuk választani. Mindkét esetben az elemek szorzata 1. Határozzuk meg a hozzájuk tartozó permutációk előjelét! A főátlóelemek az identitáshoz tartoznak, így ennek pozitív az előjele. A másik tag az $(1,2,...,n-1,n) = (n-1,n)\dots(1,2)(2,3)$ permutáció, melynek előjele -1^{n-1}, ami páros n-re 1, páratlanra -1. Így az összeg valóban 0 vagy 2 az n paritásától függően. □

Tehát a (6.1) egyenletrendszernek páros esetben nem egyértelmű a megoldása, páratlan esetben egyértelmű. A $\omega_i \geq 0$ feltételnek azonban még így sem feltétlenül tesz eleget a megoldásunk.

A megoldás létezéséhez a (6.1) feltételeken kívül mást is figyelembe kell venni: egyik szögtartomány sem érhet bele a szomszédjának a befoglaló körébe. Ezt úgy küszöbölhetjük ki, ha azt is feltesszük, hogy a befoglaló körök érintői által meghatározott szögtartományba nem érhet bele a szomszédjainak a szögtartománya. Legyen t_i^0 és t_i^1 a v-ből a v_i befoglaló köreihez húzott két érintő, és legyen a vv_i félegyenes és a t_i^1 (illetve t_i^0) által bezárt szög γ_i. A feltétel úgy is megfogalmazható, hogy minden ω_i szögtartomány magába foglalja a γ_i szögtartományokat (lásd 6.4 ábra). Ekkor a (6.1) egyenletrendszerhez az ω_i-ről még fel kell tenni a következőket:

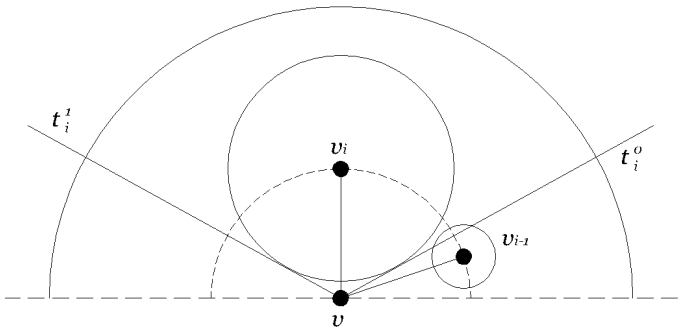
$$\begin{aligned} \omega_i &\leq \Phi_i - \gamma_{i-1} \\ \omega_i &\leq \Phi_{i+1} - \gamma_{i+1} \end{aligned} \tag{6.2}$$



6.4. ábra. A v_i csúcsokhoz tartozó befoglaló körök érintőjébe nem metszhet bele a W_i szögtartomány.

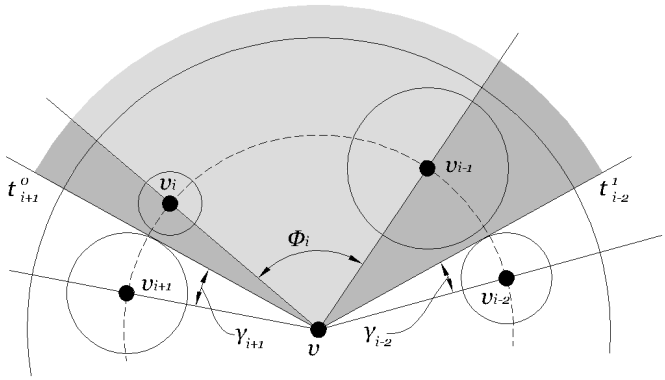
Ha a (6.2) és (6.1) feltételeket feltesszük, még akkor sem biztos, hogy létezik jó megoldás. Az egyik probléma az lehet, hogy ha túl közel vannak a körök egymáshoz, azaz ha az egyik kör érintője belemetsz a mellette lévő körbe (lásd 6.5. ábra). Ez kiküszöbölhető úgy, hogy ha kezdetben a $C_b(v)$ kerületét minél kisebbnek választjuk és úgy osztjuk el a v_i-ket, hogy befoglaló köreik a lehető legközelebb legyenek egymáshoz. A fennmaradó, minimális "hézagot" a $C_b(v)$ körön pedig egyenletesen elosztjuk a v_i befoglaló körei között. Ezek után a úgy növeljük a $C_b(v)$ r sugarát, hogy a fent említett eset ne álljon fent. Ez nyilván megtehető, hiszen r sugarának növelésével a v_i befoglaló köre egyre távolabb kerül a szomszédjai befoglaló köreitől, míg a körök érintőihez tartozó szögtartomány egyre kisebb lesz.

Egy másik probléma lehet, ha az egyik Φ_i szög nagyobb mint a két szomszédos szögtartományból lefoglalható szögek összege, azaz $\Phi_i > (\Phi_{i-2} - \gamma_{i-2}) + (\Phi_i - \gamma_{i+1})$. Ekkor ugyanis (6.2) és (6.1) egyenlőtlenségeknek biztosan nincs megoldása (lásd 6.6. ábra). Ha a fent említett transzformációt elvégezzük, ez



6.5. ábra. Nem tudjuk megválasztani a szögeket, ha az egyik kör érintője belemetsz a másik körbe.

nem fordulhat elő, hiszen ez azt jelentené, hogy a Φ_i szög aránytalanul nagy a Φ_{i-1} és a Φ_{i+1} szögekhez képest. Ha viszont kezdetben arányosan osztottuk el a v_i-ket, akkor az r sugarának növelésével ilyen aránytalan elosztás nem állhat fenn.



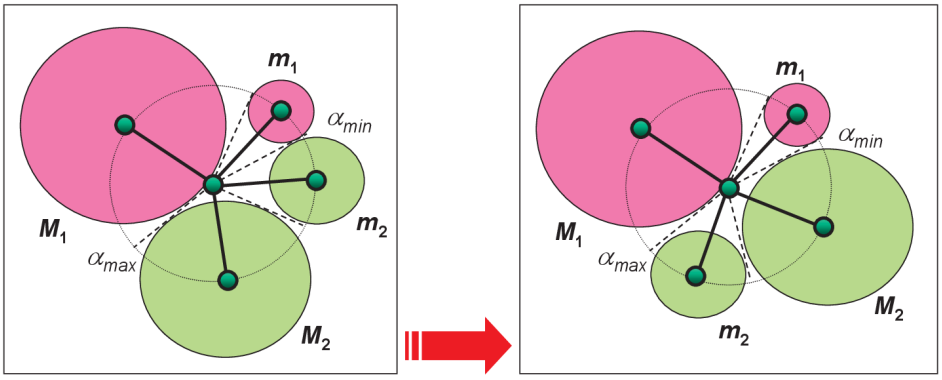
6.6. ábra. Ha aránytalanul nagy a távolság két csúcs között, a világos sávot nem tudjuk elosztani a sötétebbek között.

Összefoglalva azt mondhatjuk, hogy bizonyos esetekben meghatározhatóak a W_i szögtartományok szögei, de általánosságban nehéz megmondani, hogy a (6.1) és (6.2) feltételeknek van-e megoldása. A továbbiakban olyan eseteket vizsgálunk, amikor az adott gráfra kiszámítottuk a W_i szögtartományok szögét.

6.2. Szögarány és szögfelbontás optimalizálása egyenlő szögű esetben

Képzeljük el, hogy van egy fix buborék lerajzolásunk a 6.1. alfejezetben bemutatott algoritmus alapján. Idézzük fel, hogy a egy lerajzolás szögfelbontása (angular resolution) azt jelenti, hogy mekkora a legkisebb szög, melyet a gráf egy csúcsából kiinduló két szomszédos él bezár. A szögfelbontás optimális, ha ez a szög a lehető legnagyobb. A szögarány (aspect ratio) pedig akkor a legjobb, ha gráfban a legnagyobb és a legkisebb szög hányadosa a lehető legkisebb. Egy fa konkrét buborék lerajzolásában ezek a szögek viszont változhatnak, ha a egy csúcs körül az egyes gyermekeknek, és persze ezzel együtt a hoz-

zájuk tartozó W_i szögtartományoknak megváltoztatjuk a sorrendjét. Valóban, két szomszédos csúcsba mutató él közötti szög a két csúcshoz tartozó W_v szögtartományok méretétől függ. Ha megcseréljük a szögtartományok sorrendjét, akkor tehát más szögek lesznek az élek között, és így a szögfelbontás és a szögarány is változhat (lásd 6.7. ábra).



6.7. ábra. Egy csúcs körül a gyermekek közti szögek változhatnak, ha megcseréljük a szögtartományok sorrendjét.

A következő algoritmusban azt vizsgáljuk, hogy egy konkrét lerajzolásban milyen sorrendben kell venni a csúcsok gyermekeit, hogy optimális legyen az adott csúcs körül a lerajzolás szögfelbontása illetve szögaránya.

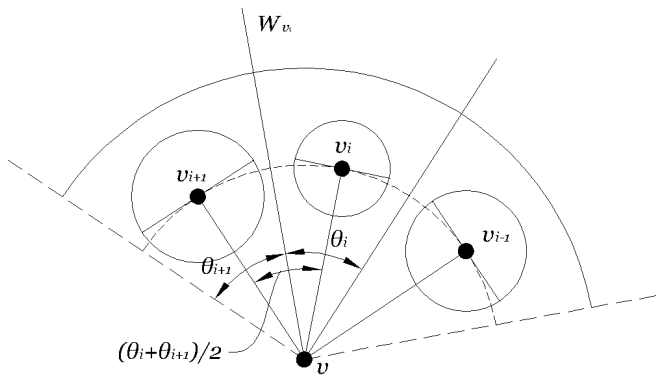
Legyen S a T gyökeres fa egy fix buborék lerajzolása a 6.1. fejezetben leírt algoritmus szerint. Legyen v a T gyökeres fa tetszőleges csúcsa. Legyenek v gyermekei $v_1, v_2, ..., v_n$, és jelöljük a v_i gyermekekhez tartozó szögtartományt W_i-vel, a W_i szögét pedig θ_i-vel. Legyen B_i az a körcikk, melyet a $C_k(v)$ befoglaló körből a W_i szögtartomány metsz ki. Jelöljük az $\{1, ..., n\}$ halmaz összes lehetséges permutációját tartalmazó halmazt Σ^k-val, és legyen $\sigma \in \Sigma^n$. A v **szögeinek felcserélése a σ szerint** azt jelenti, hogy a B_i körcikkeket - és bennük a v_i-k rajzait-, a v körül a σ sorrendjének megfelelően helyezzük el. Mivel a buborék lerajzolásban a csúcsok befoglaló körei nem metszik egymást, a két csúcs szögeinek felcserélését tetszőleges sorrendben elvégezve ugyanazt a lerazolást kapjuk. Egy tetszőleges S buborék lerajzolásban **a szögek felcserélésének művelete** alatt azt értjük, hogy néhány v csúcs szögeit felcseréljük egy $\sigma_v \in \Sigma^{d(v)}$ permutáció szerint. Amikor v **szögeinek sorrendjéről** beszélünk, akkor a B_i körcikkek elhelyezkedését értjük alatta.

Célunk tehát, hogy egy olyan algoritmust mutassunk, mely egy tetszőleges buborék lerajzolásnak megadja minden csúcs szögeinek azt sorrendjét, mely optimális szögfelbontású és szögarányú rajzot ad a szögek felcserélésének műveletére nézve. Az algoritmus csak egy csúcs körül keres optimális szögfelbontású és szögarányú rajzot, de be foguk látni, hogy valójában az egész gráfra optimális lerajzolást ad.

6.2.1. Algoritmus

Legyen v egy tetszőleges csúcs és tegyük fel, hogy a $v_1, v_2, ..., v_n$ gyermekeinek adva van a egy buborék lerajzolása a 6.1. alfejezetben vázolt algoritmus alapján. A v_i-hez tartozó W_i szögtartomány szöge legyen θ_i. Ekkor a v-ből v_i-be és a v_{i+1}-be vezető élek közötti szög éppen:

$$\frac{\theta_i + \theta_{i+1}}{2} \tag{6.3}$$



6.8. ábra. Két csúcs közötti szög kiszámítása a tartalmazó szögtartományok szögei alapján.

Ezt szöget jelöltük a 6.1. alfejezetben Φ_{i+1}-el. Vegyük az $\{1, ..., n\}$ halmaz egy tetszőleges σ permutációját. Ez fogja megadni a v_i gyermekek sorrendjét a $C_b(v)$ belső körön. Mivel σ szögeiről innentől kezdve beszélhetünk úgy, mint a θ_{σ_i} szögekről, vagy $\frac{\theta_{\sigma_i}+\theta_{\sigma_{i+1}}}{2}$ szögekről ezért az egyértelműség kedvéért az előbbi esetben σ-ban szereplő szögekként, vagy σ szögeiként fogjuk említeni, utóbbi esetben σ-hoz tartozó szögeket mondunk. Ekkor a (6.3) egyenlet alapján a v csúcs szögfelbontása:

$$AngResl_\sigma = \min_{1\le i\le n}\left\{\frac{\theta_{\sigma_i}+\theta_{\sigma_{i+1}}}{2}\right\}.$$

A v csúcs körül az optimális szögfelbontás pedig:

$$OptAngResl_\sigma = \max_{\sigma\in\Sigma^n}\{AngResl_\sigma\} = \max_{\sigma\in\Sigma^n}\left\{\min_{1\le i\le n}\left\{\frac{\theta_{\sigma_i}+\theta_{\sigma_{i+1}}}{2}\right\}\right\}$$

Az általánosság megszorítása nélkül tegyük fel, hogy minden θ_i különböző, és $\theta_1, ..., \theta_n$-t rendezzük növekvő sorrendbe. A sorba rendezés után bevezetünk egy új jelölést a szögekre:

$$m_1 < m_2 < ... < m_k < mid < M_k < M_{k-1} < ... < M_1, \tag{6.4}$$

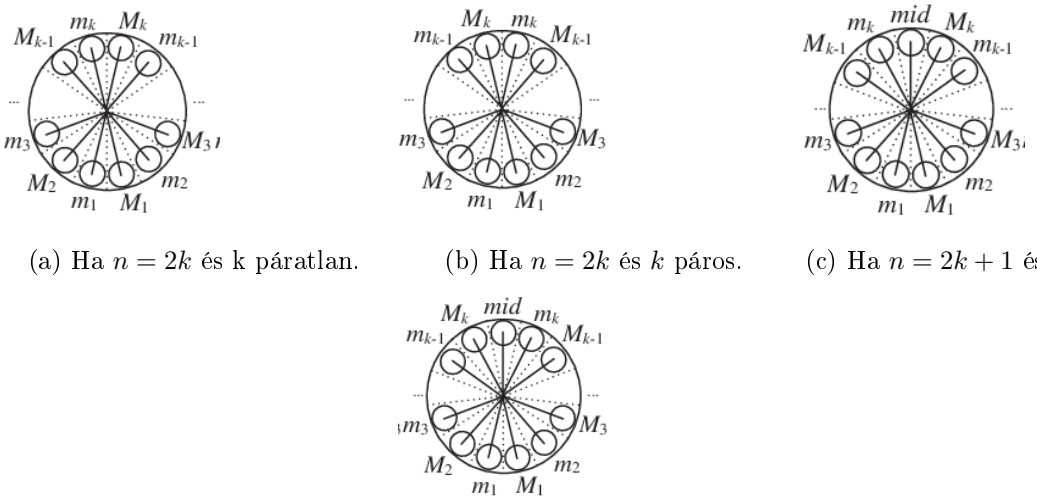
ha n páratlan és

$$m_1 < m_2 < ... < m_k < M_k < M_{k-1} < ... < M_1, \tag{6.5}$$

ha n páros.

6.2.1. Tétel. *Legyen egy v tetszőleges csúcsnak és a $v_1, v_2, ..., v_n$ gyermekeinek adva egy S buborék lerajzolása a 6.1 alfejezet algoritmusa alapján. Jelöljük v_i gyermekekhez tartozó szögeket (6.4) illetve (6.5) szerint. Vegyük a szögeknek azt a sorrendjét, melyet n paritásától függően (6.6) határoz meg (lásd 6.9. ábra). Ez a sorrend optimális szögfelbontású és szögarányú rajzot ad a v csúcs körül a szögek felcserélésének a műveletére nézve.*

$$
\begin{aligned}
&m_1, M_2, m_3, ..., M_{k-1}, m_k, M_k, m_{k-1}, ..., M_3, m_2, M_1, && \textit{ha } n = 2k \textit{ és } k \textit{ páratlan,} \\
&m_1, M_2, m_3, ..., m_{k-1}, M_k, m_k, M_{k-1}, ..., M_3, m_2, M_1, && \textit{ha } n = 2k \textit{ és } k \textit{ páros,} \\
&m_1, M_2, m_3, ..., M_{k-1}, m_k, mid, M_k, m_{k-1}, ..., M_3, m_2, M_1, && \textit{ha } n = 2k+1 \textit{ és } k \textit{ páratlan,} \\
&m_1, M_2, m_3, ..., m_{k-1}, M_k, mid, m_k, M_{k-1}, ..., M_3, m_2, M_1, && \textit{ha } n = 2k+1 \textit{ és } k \textit{ páros,}
\end{aligned}
\tag{6.6}
$$



(a) Ha $n = 2k$ és k páratlan. (b) Ha $n = 2k$ és k páros. (c) Ha $n = 2k + 1$ és k páratlan.

(d) Ha $n = 2k + 1$ és k páros.

6.9. ábra. Szögek sorrendje optimális lerajzolásban [26].

Bizonyítás. Egy erősebb állítást fogunk belátni. Azt állítjuk, hogy a(6.6) által meghatározott sorrendben a legkisebb szög a lehető legnagyobb és a legnagyobb szög a lehető legkisebb a v csúcs szögeinek felcserélésének műveletére nézve. Vizsgáljuk azt az esetet, amikor $n = 2k + 1$ és k páratlan:

$$\sigma := (m_1, M_2, m_3, ..., M_{k-1}, m_k, mid, M_k, m_{k-1}..., M_3, m_2, M_1) \tag{6.7}$$

6.2.2. Állítás. *σ legkisebb szöge vagy $\frac{m_k + mid}{2}$, vagy $\frac{M_i + m_{i-1}}{2}$ valamilyen $1 \leq i \leq k$-ra. σ legnagyobb szöge $\frac{M_k + mid}{2}$ vagy $\frac{M_{i-1} + m_i}{2}$.*

Bizonyítás. Nyilvánvalóan a fent említett szögek mindig szomszédosak a σ-ban. Nézzük meg a σ-hoz tartozó szögek viszonyát σ körüljárási sorrendjének megfelelően!

$$\frac{M_1 + m_2}{2} < \frac{M_3 + m_2}{2} > \frac{M_3 + m_4}{2} < ... > \frac{M_{l-1} + m_l}{2} < ... > \frac{M_j + m_{j-1}}{2} < ... < \frac{M_k + mid}{2} >$$
$$\frac{m_k + mid}{2} < \frac{M_{k-1} + m_k}{2} > ... > \frac{M_4 + m_3}{2} < \frac{M_2 + m_3}{2} > \frac{M_2 + m_1}{2} < \frac{M_1 + m_1}{2} < \frac{M_1 + m_2}{2}.$$

Ezek alapján $\frac{M_l + m_{l-1}}{2}$-et és $\frac{mid + m_k}{2}$-t felülről becsüli a két szomszédja, míg $\frac{M_{j-1} + m_j}{2}$-t $\frac{mid + M_k}{2}$-t pedig alulról becsüli. Tehát a maximum illetve a minimum csak ezen az elemek közül kerülhet ki. □

Most belátjuk, hogy σ-nak optimális a szögfelbontása. Tegyük fel, hogy σ legkisebb szöge $\frac{M_i + m_{i-1}}{2}$-en vétetik fel valamilyen $1 \leq i \leq k$-ra. A másik eset hasonlóan kezelhető. Legyen $\delta \in \Sigma^n$ egy olyan permutáció, melynek optimális a szögfelbontása, azaz δ-ban két olyan szög került egymás mellé, melyek átlaga éppen a (6.2.1) egyenletben bevezetett $OptAngResl$-el egyenlő.

6.2.3. Állítás. $OptAngResl = \frac{M_i + m_{i-1}}{2}$, *azaz δ és σ szögfelbontása egyenlő.*

Bizonyítás. Indirekt tegyük fel, hogy $\frac{M_i + m_{i-1}}{2}$ nem σ legkisebb szöge. Ez vagy úgy lehet, hogy $\frac{M_i + m_{i-1}}{2}$ nem szerepel a δ-hoz tartozó szögek között, azaz m_{i-1} és M_i szögek nem szomszédosak, vagy úgy, hogy nem ez a legkisebb szög.

Első eset. m_{i-1} és M_i szögek nem szomszédosak δ-ban. Legyen m_{i-1} két szomszédja δ-ban $x < y$. Ekkor nyilván igaz, hogy $M_i < x < y$, különben δ legkisebb szöge nem lenne nagyobb mint σ-é. Definiáljuk R_A, R_B, és R_C halmazokat a következőképpen:

$$\underbrace{m_1 < m_2 < ... < m_{i-2}}_{R_A} < m_{i-1} < \underbrace{...}_{R_B} < M_i < \underbrace{M_{i-1} < ... < x < ... < y < ... < M_1}_{R_C}$$

Ekkor R_A-nak minden szomszédja R_C-ből fog kikerülni, hiszen ha nem így lenne, akkor lenne egy olyan szög, amely δ-hoz tartozik és kisebb, mint $\frac{M_i + m_{i-1}}{2}$, ami ellent mond δ optimalitásának. R_A-nak $i-2$ eleme van, így $i-1$ szomszédjának kell lennie R_C-ben. R_C-nek $i-1$ eleme van, de ebből x és y egyik oldala m_{i-1}-hez tartozik. Így csak $i-2$ szabad szomszéd "oldal" kerülhet ki R_C-ből, pedig $i-1$-re lenne szükség δ optimalitásához.

Második eset. m_{i-1} és M_i szögek szomszédosak δ-ban. Ekkor azt állítjuk, hogy $OptAngResl = \frac{M_i + m_{i-1}}{2}$. Ugyanis, ha $OptAngResl < \frac{M_i + m_{i-1}}{2}$, akkor σ legkisebb szöge nagyobb mint δ-é, ami ellentmond δ optimalitásának. $OptAngResl > \frac{M_i + m_{i-1}}{2}$ pedig azért nem lehet, mert δ-hoz tartozik $\frac{M_i + m_{i-1}}{2}$ szög is, tehát a legkisebb szög legfeljebb ennyi lehet. □

Hasonlóan beláthatjuk azt is, hogy két szomszédos él között a legnagyobb szög $\frac{M_k + mid}{2}$ vagy $\frac{M_{i-1} + m_i}{2}$. Ebből nyilván következik, hogy a legnagyobb és a legkisebb szög hányadosa a lehető legkisebb, azaz σ szögaránya optimális.

□

Ha tehát a T minden csúcsára vesszük a (6.6) által meghatározott sorrendet, akkor az nem csak optimális szögfelbontást és szögarányt ad csúcsonként, hanem minden csúcsnál a legkisebb szög a lehető legnagyobb, a legnagyobb szög a lehető legkisebb lesz. Ezt nevezzük **lokálisan optimális** lerajzolásnak. Azt állítjuk, hogy ekkor a teljes gráfra is igaz lesz ez a tulajdonság, azaz **globálisan optimális** lesz a lerajzolás. Ebből persze következik, hogy a (6.6) sorrend optimális szögfelbontású és szögarányú a teljes gráfra a szögek felcserélésének műveletére nézve.

6.2.4. Állítás. *Legyen S egy tetszőleges lerajzolása a $T = (V, E)$ gráfnak. Tegyük fel, hogy S-ben bármely v csúcs két szomszédos éle által bezárt szög közül a legkisebb szög maximális az összes lehetséges lerajzolás közül és a legnagyobb szög pedig minimális. Ekkor az egész gráfra igaz lesz, hogy a legkisebb szög egy csúcs két szomszédos éle között minimális lesz, és a legnagyobb szög egy csúcs két szomszédos éle között pedig maximális.*

Bizonyítás. Indirekt tegyük fel, hogy van U egy globálisan jobb lerajzolásunk, azaz aminek a legkisebb szöge nagyobb mint a lokálisan optimális lerajzolás legkisebb szöge. Legyen az U lerajzolásnak a legkisebb szöge α, az S legkisebb szöge β. A feltevés szerint $\alpha > \beta$. Legyen a β-hoz tartozó csúcsnál a legkisebb szög az U-ban γ. Mivel S lokálisan optimális, ezért $\beta \geq \gamma$, viszont α a legkisebb szög U-ban, ezért $\gamma \geq \alpha$, azaz $\beta \geq \alpha$, ami ellentmondás. A bizonyítás hasonlóan megy a legnagyobb szög minimalitására. □

6.3. Terület csökkentése

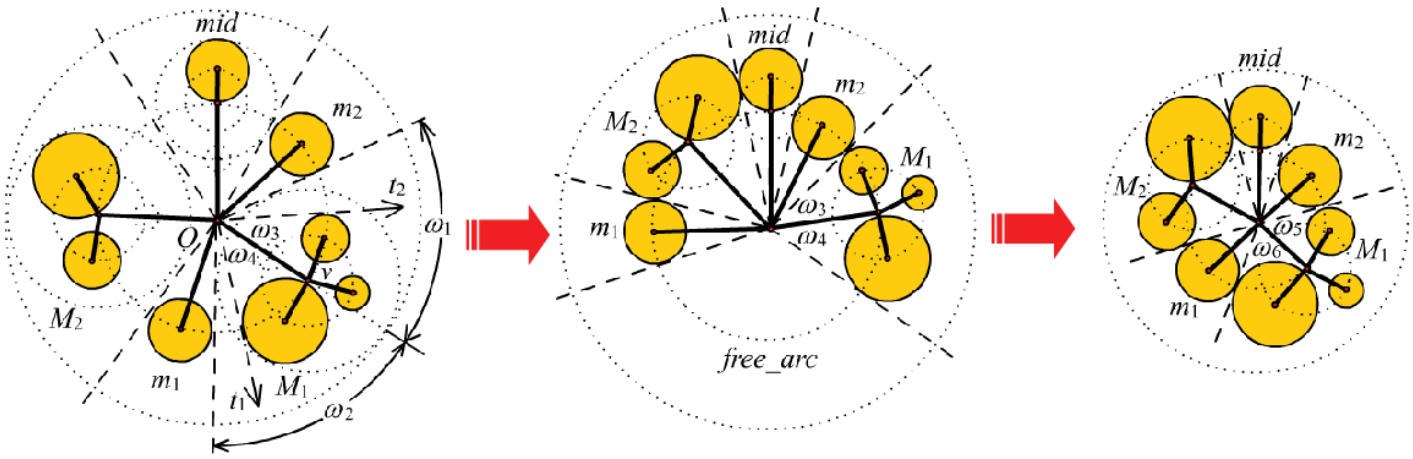
Ebben a részben mutatunk egy példát arra, hogyan állíthatunk elő nem egyenlő szögű buborék lerajzolást. Az egészet az motiválja, hogy a 6.1. alfejezetben vázolt algoritmus ábrája túl nagy területű, és ezért szeretnénk módosítani rajta. Vegyünk egy, a 6.1. alfejezetben vázolt algoritmus alapján készült buborék lerajzolást, mely egyenlő szögű ábrát ad. A rajzban egy tetszőleges v csúcs gyermekei legyenek $v_1, v_2, ..., v_n$. A v_i-k részfái egy-egy W_i szögtartományba esnek, melyet a vv_i egyenes éppen felez. Viszont az 6.1.2. megjegyzés alapján lehetnek "hézagok" v_i részfáinak rajzai között, hiszen azok befoglaló körei nem feltétlenül érintik W_i két szögszárát. Ha ezeket a hézagokat megpróbáljuk csökkenteni, azáltal, hogy valahogyan közelebb húzzuk a v_i részfáit egymáshoz, kisebb területű ábrát kapunk. Sőt, ha elhagyjuk a rajznak azt a feltételét, hogy egy tetszőleges csúcshoz tartozó szögtartománynak a csúcs teljes befoglaló körét kell tartalmaznia, hanem elég csak a csúcs gyermekeinek befoglaló köreit tartalmaznia, akkor még kisebb ábrát kaphatunk. Mi ez utóbbi esetben adunk algoritmust a terület javítására.

Legyen t_i^0 és t_i^1 a az a két félegyenes, amely v_i gyermekek befoglaló köreit érinti. Ha most azt a W_i' szögtartományt rendeljük v_i-hez, melyet t_i és t_{i+1} határol, akkor azt a vv_i egyenes már nem feltélenül felezi. Forgassuk egymás mellé a W_i' szögtartományokat úgy, hogy a "hézag" szögtartomány W_1' és W_n' közé essen. Ezek után transzformálhatjuk úgy a részfákat, hogy a vv_i élek hosszát csökkentve minden W_i' és W_{i+1}' szögtartomány érintse egymást, és ezzel az ábra befoglaló köre kisebb lesz. Ezt úgy a legegyszerűbb elképzelni, mintha a rajz egy v középpontú körlapon lenne. A v_i-k részfái fixek, mintha botok lennének az élek, melyeknek az egymással bezárt szögei is fixek. A vv_i élek viszont mintha fonalak lennének és ki vannak vezetve a v középponton a körlap alá. Ha most a fonalakat elkezdjük húzni addig, amíg minden gyermek részfája valahol összeér, akkor pont egy olyan nem egyenlő szögű rajzot kapunk, ahol minden v_i-hez tartozó szögtartomány éppen W_i', és ennek nyilván kisebb a területe, mint az eredeti rajzé (lásd 6.10. ábra). A transzformációval járó probléma, hogy az ererdeti lerajzoláshoz képest a szögfelbontás és szögarány elromolhat. A továbbiakban arra keresünk megoldást, hogy hogyan lehet egy ilyen lerajzolásnak a szöfelbontását és szögarányát javítani.

6.4. Szögfelbontás maximalizálása nem egyenlő szögű esetben

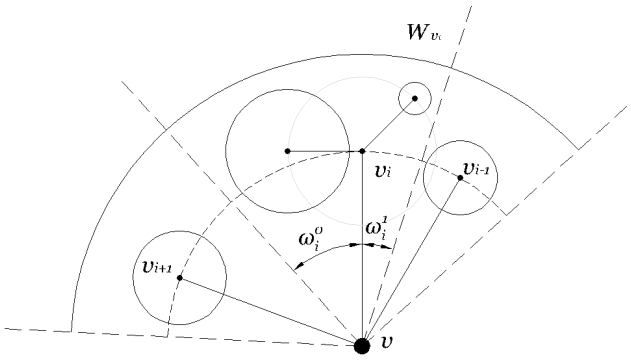
Vegyünk most is egy fix, nem egyenlő szögű lerajzolást, melynek a szögfelbontását és a szögarányát szeretnénk optimalizálni. Egyrészt elvégezhetjük az 6.2. alfejezetben szereplő algoritmust most is, amely a szögek felcserélésének műveletére nézve optimális megoldást ad. Azonban tovább javíthatunk a szögfelbontáson, ha egy másik műveletet végezhetünk a rajzon; egészen pontosan ha a v szülőből a v_i gyermekbe vezető félegyenesre tükrözzük a v_i részfájának a rajzát.

Legyen a v_i gyermekekhez tartozó W_i szögtartományok szöge θ_i. Ekkor a vv_i félegyenes két szögtartományra osztja W_i-t, melyek legyenek ω_i^0 és ω_i^1. Ekkor ha tükrözünk a vv_i egyenesére, akkor a v_{i-1} és

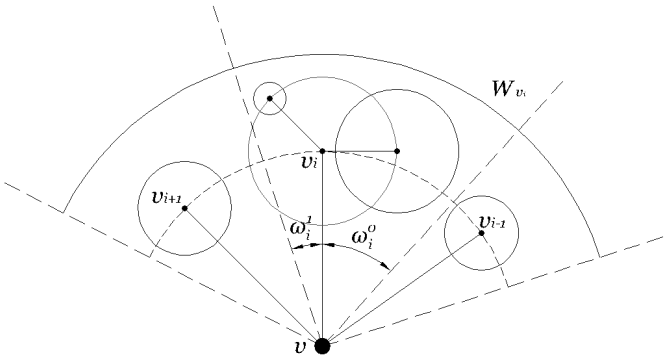


6.10. ábra. Terület csökkentése [26].

v_i közötti, valamint a v_i és v_{i+1} közötti szög nagysága megváltozik (lásd 6.11. és 6.12. ábrák).



6.11. ábra. Nem egyenlő szögű eset tükrözés előtt.



6.12. ábra. Nem egyenlő szögű eset tükrözés után. A v_i és szomszédjai közötti szögek változnak.

6.4.1. Megjegyzés. *Előfordulhat, hogy a tükrözés után a W_i szögtartomány átlóg a szomszédjának a szögtartományába. Ez úgy küszöbölhető ki, hogy a többi szögtartományt forgjatuk el a v középpont körül. Mivel a tükrözésnél a W_i szögtartomány szöge nem változik meg, ezért ez megtehető (lásd 6.11. és 6.12.*

ábrák).

A szögfelbontás és a szögarány optimalizálásához úgy kerülhetünk közelebb, ha bevezetünk egy új jelölést és felírjuk a v_i gyermekekhez tartozó W_i szögtartományok részszögtartományait ciklikus sorrendben:

$$\underbrace{\omega_1^{t_1}\omega_1^{t'_1}}_{W_1}\underbrace{\omega_1^{t_2}\omega_1^{t'_2}}_{W_2}\underbrace{\dots}_{\dots}\underbrace{\omega_n^{t_n}\omega_1^{t'_n}}_{W_n} \tag{6.8}$$

Itt t_i és t'_i {0,1}-beli elemek és $t_i + t'_i = 1$. Ezek alapján a szögfelbontás és a szögarány így írható fel:

$$AngResl = \min_{1\le i\le n}\{\omega_i^{t'_i} + \omega_{i+1}^{t_i+1}\}$$

$$AspRatio = \frac{\max_{1\le n}\{\omega_i^{t'_i} + \omega_{i+1}^{t_i+1}\}}{\min_{1\le i\le n}\{\omega_i^{t'_i} + \omega_{i+1}^{t_i+1}\}}$$

A probléma tehát az, hogy a t_i-ket és t'_i-ket úgy válasszuk meg, hogy optimalizáljuk az *AngResl* és az *AspRatio* értékeit. A t_i és t'_i értékek tehát azt jelentik, hogy a tükrözés során végül melyik irányban lesz W_i két részszögtartománya. Ha minden t_i és t'_i értéket kiválasztottunk, akkor azt egy szögkiosztásnak fogjuk hívni.

A következőkben arra keressük a választ, hogy egy csúcs körül mennyire lehet kicsi a szögfelbontás és mennyire nagy a szögarány. A feladatot a következő problémákban definiáljuk majd belátjuk, hogy megoldásuk egy teljes párosítási problémára vezethető vissza.

6.4.2. Definíció (Szögfelbontás (szögarány) probléma). *Legyen egy v tetszőleges csúcsnak legyen és $v_1, v_2, ..., v_n$ gyermekeinek adva van a egy S buborék lerajzolása nem egyenlő szögekkel az (6.8) jelöléseivel. Legyen r egy valós szám. Feladat, hogy eldöntsük van-e a t_i-nek és t'_i-nek olyan értéke, melyre $t_i, t'_i \in \{0,1\}$ és $t_i + t'_i = 1$, úgy, hogy $AngResl \le r$ ($AspRatio \ge r$).*

6.4.3. Tétel. *A Szögfelbontás és a szögarány probléma egy csúcs körül $O(n^{2.5})$ időben megoldható.*

Bizonyítás. Először a szögarány problémát nézzük, a szögfelbontás probléma bizonyítása hasonlóan megy. Legyen r a kívánt alsó határa *AspRatio*-nak. Vezessünk be a v_i csúcsokhoz tartozó W_i szögtartományok részszögtartományaira a következő jelölést:

$$\underbrace{b_1, b'_1}_{W_1}\underbrace{b_2, b'_2}_{W_2}\underbrace{\dots}_{\dots}\underbrace{b_n, b'_n}_{W_n} \tag{6.9}$$

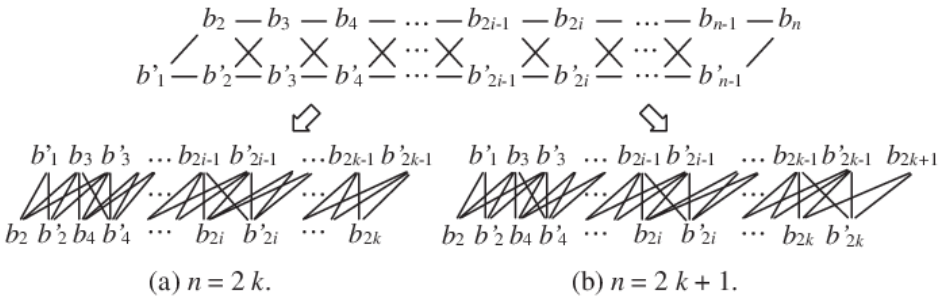
Tehát b_i és b'_i tartozik a v_i szögtartományához egy konkrét buborék lerajzolásban. b_i-nek a szomszédjai b_{i+1}, b'_{i+1} és b_{i-1}, b'_{i-1} közül kerülnek ki, attól függően, hogy W_i, W_{i-1} és W_{i+1} részszögeit hogyan tükröztük. Például, ha b_i szomszédja b'_{i+1}, akkor a vv_i és vv_{i+1} egyenesek közötti szög éppen $b_i + b'_{i+1}$. A feladat az, hogy minden b_i-nek meghatározzuk a szomszédját, úgy, hogy a szögarány legfeljebb r legyen. Az algoritmus minden lépésben megvizsgál egy (x, y) párt, ahol $x \in \{b_i, b'_i\}$ és $y \in \{b_i + 1, b'_i + 1\}$, és eldönti, hogy létezik-e olyan legfeljebb r szögarányú szögkiosztás, amelyben a legkisebb szög éppen (x, y)-hoz tartozik. Azaz a legkisebb szög két szomszédos csúcs között $x + y$ lesz. Ehhez készítünk egy $G_{x,y}$ páros gráfot, melyben pontosan akkor van teljes párosítás, ha létezik ilyen szögkiosztás.

Az egyszerűség kedvéért legyen a vizsgált szögpár a b_1, b'_n, azaz tegyük fel, hogy a legkisebb szög értéke $\Phi = b_1 + b'_n$. Ekkor készítsünk $G_{b_1,b'_n} = ((U, W), E)$ a páros gráfot a következő képpen:

$$U = \{b'_1, b_3, b'_3, ..., b_{2i+1}, b'_{2i+1}, ..., b_{2k-1}, b'_{2k-1}\}.$$
$$W = \{b_2, b'_2, b_4, b'_4, ..., b_{2i}, b'_{2i}, ..., b_{2k-2}, b'_{2k-2}, b_{2k}.\}$$

Minden $i \in \{2,3,...,k-1\}$-re $s \in \{b_{2i-1}, b'_{2i-1}\}$, $t \in \{b_{2i-2}, b'_{2i-2}, b_{2i}, b'_{2i}\}$ csúcs között akkor megy él, ha $\Phi \leq s+t \leq r\Phi$. Így garantáljuk, hogy a legkisebb szög Φ legyen, és a szögarány, azaz a legkisebb és legnagyobb szög hányadosa is legfeljebb r legyen. b'_1-hez és b_n-hez pedig így kapcsolódhatnak élek:

$$(s,t) \in E, \text{ ha } s \in \{b_{2k-1}, b'_{2k-1}\},\ t \in \{b_{2k-2}, b'_{2k-2}, b_{2k}\} \text{ és } \Phi \leq s+t \leq r\Phi$$
$$(b_1,t) \in E, \text{ ha } t \in \{b_2, b'_2\} \text{ és } \Phi \leq s+t \leq r\Phi$$



6.13. ábra. A G_{b_1,b'_n} gráf szerkezete [26].

Most belátjuk, hogy ha G_{b_1,b'_n}-ben akkor és csak akkor létezik teljes párosítás, ha van olyan szögkiosztás, melynek a legkisebb szöge Φ és a szögfelbontása legfeljebb r. Először tegyük fel, hogy G_{b_1,b'_n}-ben létezik egy M teljes párosítás.

Ahhoz, hogy legyen érvényes szögkiosztás, az kell, hogy M-ben minden b_i és b'_i csúcspárnak pontosan egy szomszédja legyen a $\{b_{i-1}, b'_{i-1}\}$ halmazban és egy a $\{b_{i+1}, b'_{i+1}\}$ halmazban. Ha ez igaz M-re, akkor konstruálhatunk egy lerajzolást, amelyben minden (s,t) élhez tartozó két részszögtartomány egymás mellé fog kerülni, és b_1 pedig b'_n mellett lesz. Ekkor ráadásul $AspRatio \leq r$ és $AngResl = \Phi = b_1 + b'_n$ a gráf konstrukciója miatt.

Nézzük tehát az M párosítást. b'_1 mindenképpen párja b_2-nek vagy b'_2-nek, mert csak beléjük vezet él a gráfban. Tegyük fel, hogy b_2 az ő párja. Ekkor viszont b_2-nek is biztosan lesz egy párja b_3 vagy b'_3 közül. Így tovább b_i és b'_i közül biztosan lesz egy, amelyiknek b_{i-1} és b'_{i-1} lesz a párja és emiatt a másiknak a párja biztosan b_{i+1} vagy b'_{i+1} lesz. Végül b_{2k-2} és b'_{2k-2} közül pontosan egy marad párosítatlanul, úgyhogy az lesz b_{2k} párja. Így van egy érvényes szögkiosztásunk, amely a korábbiak alapján teljesíti a feltételeket.

Most tegyük fel, hogy létezik egy D buborék lerajzolás, melynek a szögaránya legfeljebb r és a szögfelbontása $b_1 + b'_n$. Ekkor a W_1 szögtartomány másik részszögének, b'_1-nek a szomszédja D-ben biztosan b_2 vagy b'_2. Tegyük fel, hogy b_2. Mivel D szögfelbontása $b_1 + b'_n$ és szögaránya legfeljebb r, ezért $b_1 + b'_n \leq b_1 + b_2 \leq r(b_1 + b'_n)$. Így a (b'_1, b_2) él szerepelni fog a G_{b_1,b'_n} gráfban. Hasonlóan, a többi szomszédos szögtartományhoz tartozó élek is szerepelni fognak a gráfban.

Tehát ha minden $1 \leq i \leq n$-re megvizsgáljuk, hogy az (x,y) párra, melyre $x \in \{b_i, b'_i\}$ és $y \in \{b_i+1, b'_i+1\}$ létezik-e $x+y$ szögfelbontású lerajzolás, melynek a szögaránya legfeljebb r, akkor megkap-

juk, hogy van-e egyáltalán r-nél nagyobb szögfelbontású lerajzolás. Tehát a szögfelbontás és szögarány probléma megoldható.

Becsüljük meg az algoritmus lépésszámát! A Φ érték összesen legfeljebb $O(n)$ értéket vehet fel, mert minden b_i szögtartománynak összesen legfeljebb 4 szomszédja lehet a $G_{x,y}$ gráfban egy tetszőleges (x, y) értékre. A szögfelbontás probléma megoldásához először sorba rendezzük ezt az $O(n)$ értéket, majd mindegyik (x, y) párra eldöntjük a $G_{x,y}$ gráfról, hogy létezik-e benne teljes párosítás. Ennek eldöntése $O(\sqrt{mn})$-ben megtehető, ahol m az élek száma. Így összesen $O(n \cdot \log n) + n\sqrt{mn} = O(n \cdot \sqrt{n}n)$ idejű algoritmust adtunk a szögfelbontás és a szögarány problémára.

□

Tehát a 6.1. alfejezetben konstruáltunk egy buborék lerajzolást, melyet a 6.2. alfejezetben úgy módosítottunk, hogy optimális szögarányú és szögfelbontású ábrát kapjunk a szögek felcserélésének a műveletére nézve. Ezután kisebb területű ábrát készítettünk de egyidejűleg rontottuk a szögarányt és szögfelbontást. Eredményképp nem egyenlő szögű lerajzolást kaptunk. Végül a fent bemutatott algoritmussal az egyes szögtartományok részszögeit tükröztük, hogy kijavítsuk a szögfelbontást és a szögarányt.

6.5. Terület optimalizálása

Egy tetszőleges buborék lerajzolás területet úgy csökkenthetjük tovább, ha elhagyjuk azt a feltételt, hogy egy csúcs gyermekeinek egy körön kell feküdnie a csúcs körül. Megjegyezzük, hogy ennek az esztétikai elvnek az elvesztését például úgy tudjuk kompenzálni, hogy színezzük a csúcsokat. Legyenek az egy szinten lévő csúcsok ugyanolyan színnel színezve, és a szín árnyalata pedig legyen arányos a csúcsnak a szülőjétől vett távolsággal [34].

Most is egy csúcs körül szeretnénk a $C_k(v)$ befoglaló kör területét minimalizálni, először egyenlő szögű, majd nem egyenlő szögű esetben. Tegyük fel, hogy adva van a v csúcs gyermekeinek a buborék lerajzolása. Szeretnénk úgy meghatározni a v és a v_i gyermekek közötti távolságot, hogy a gyermekek $C_k(v_i)$ befoglaló körei ne messék egymást. Az egyenlő szögű esetben induljunk ki abból, hogy bármely két csúcsra vv_i és vv_{i+1} közötti szög pontosan $\frac{2\pi}{n}$ (lásd 6.14. ábrát, ahol a gyermek csúcsok c_i-kel, és a szülő csúcs c_0-al van jelölve.) Legyen a v_i csúcs távolsága v-től d_i, és R_i a v_i-hez tartozó befoglaló kör sugara. Ekkor a $C_k(v)$ befoglaló kör sugara $\max_{1 \le i \le n} R_i d_i$ lesz. A terület optimalizálását egy konkrét buborék lerajzoláshoz felírhatjuk a következő kvadratikus programozási feladattal:

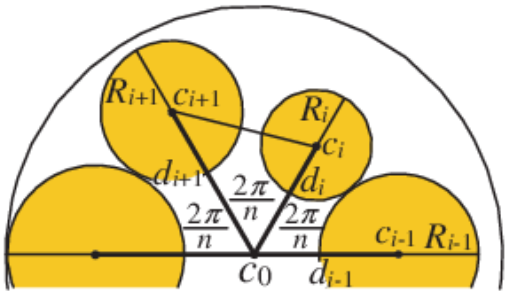
$$\min z \tag{6.10}$$

$$d_i \sin(\frac{2\pi}{n}) \ge R_i, \quad 1 \le i \le n$$

$$d_i^2 + d_{i+1}^2 - 2d_i d_{i+1} \cos(\frac{2\pi}{n}) \ge (R_i + R_{i+1})^2, \quad 1 \le i \le n \tag{6.11}$$

$$z \ge R_i + d_i. \quad 1 \le i \le n \tag{6.12}$$

Itt $z, d_i \in \mathbb{R}$ változók. A (6.11) egyenlőtlenség arra az esetre vonatkozik, amikor a $C_k v_i$ befoglaló kör érinti a vv_{i+1} vagy vv_{i-1} élet. A (6.12) egyenlőtlenség pedig azt jelenti, hogy a két szomszédos v_i és v_{i+1} csúcs közötti távolság legyen nagyobb mint $R_i + R_{i+1}$, s így a két befoglaló kör ne messen bele egymásba.



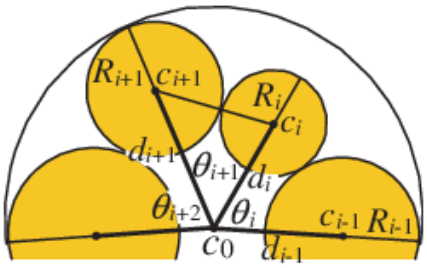
6.14. ábra. Terület probléma egyenlő szögű esetben

z minimalizálásával elérjük, hogy $z = \max_i R_i + d_i$ legyen, és ezzel a területet is minimalizáljuk. Az így felírt feladat egy lineáris programozási feladat kvadratikus megszorításokkal (QLCP).

A nem egyenlő szögű esetben a d_i távolságokon kívül azt is változónak kell tekinteni, hogy mekkora legyen a két szomszédos csúcsba menő él közötti szög. Legyen a vv_{i-1} és vv_i élek közötti szög értéke θ_i (lásd 6.15 ábrát, ahol c_0 a szülő és c_i-k a gyermek csúcsok). A terület optimalizálásának problémája így írható fel:

$$
\begin{aligned}
\min z & \\
d_i \sin(\theta_i) \geq R_i, & \quad 1 \leq i \leq n \\
d_i \sin(\theta_{i+1}) \geq R_i, & \quad 1 \leq i \leq n \\
d_i^2 + d_{i+1}^2 - 2d_i d_{i+1} \cos(\theta_{i+1}) \geq (R_i + R_{i+1})^2, & \quad 1 \leq i \leq n \\
z \geq R_i + d_i, & \quad 1 \leq i \leq n
\end{aligned}
$$

Itt d_i, z és θ_i változók. Ez a probléma nemlineáris programozási feladat, lineáris célfüggvénnyel, és itt már a megszorítások sem csak kvadratdratikusak, hanem trigonometrikus tag is van közöttük.

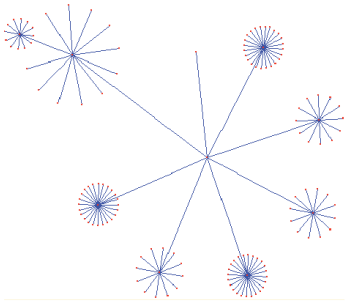


6.15. ábra. Terület probléma egyenlő szögű esetben

6.6. További eredmények

A fejezetben bemutatott buborék lerajzolást Carriere és Kazman egy gráf lerajzoló program kapcsán publikálta 1995-ben [8]. Ezt az algoritmust módosította a szögfelbontás és szögarány javításának céljából Lin és Yen [26]. Mint ahogy azt a 6.1. alfejezetben láttuk, Carriere és Kazman cikkében ez az algoritmus nem volt minden részletében kidolgozva. Hasonló alapokon nyugvó lerajzolást mutatott Melancon és

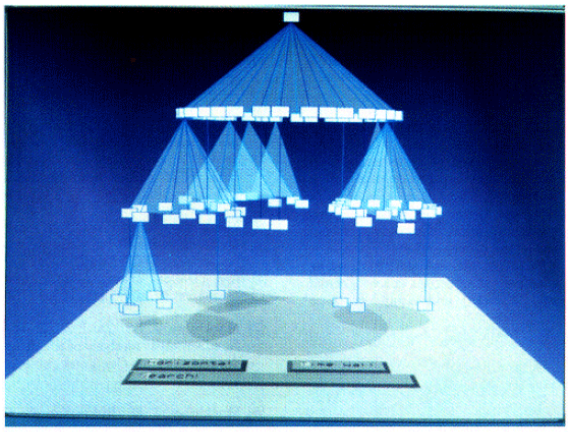
Herman 1998-ban, mely a 6.1.1. állítás összes tulajdonságát teljesíti [29]. ők adtak először olyan leírást a a buborék lerajzolás elkészítéséről, mely pontosan ki volt dolgozva.



6.16. ábra. Példa Melancon és Herman buborék lerajzolására.

A fejezet elején említett fraktál modell alapja, hogy a gyermekek mind egy kör kerületén helyezkednek el a szülő körül, illetve minden egy szinten lévő csúcs befoglaló köreinek sugara azonos. Ilyen lerajzolási módszert mutat be Koike és Yoshihara [23]. Itt az m. csúcshoz tartozó befoglaló kör sugara rekurzívan számolható ki: $r_m = \gamma \cdot r_{m-1}$, ahol $0 < \gamma < 1$. A fraktál modell hátrányát már említettük: a gyökértől távolodva túlzottan lecsökken. Előnye viszont az általunk ismertetett algoritmussal szemben az, hogy ténylegesen tökéletes szögfelbontású rajzot készíthetünk fraktál modellben. (Az általunk bemutatott algoritmus a szögtartományok felcserélésének műveletére nézve adott tökéletes szögfelbontású rajzot.) Összehasonlításképpen lásd 6.1. ábrát.

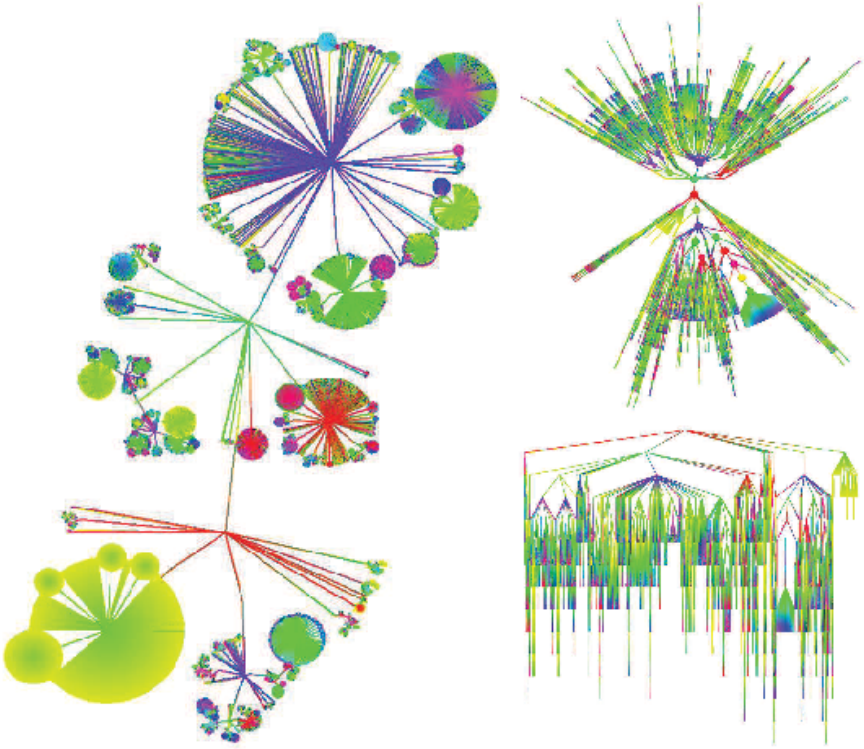
Egy másik érdekes buborék lerajzolási módot mutatott be 1998-ban Jeong és Pang. [19] cikkükben gyökeres fák háromdimenziós "lerajzolásáról" olvashatunk, melyeket tölcsérfáknak neveznek el. Egy csúcs gyermekei egy kúp alapkörének kerületén helyezkednek el, a kúp csúcsában pedig a szülőjük található. Ennek egy sík vetülete szintén egy buborék lerajzolást ad (lásd 6.17 ábra).



6.17. ábra. Tölcsér lerajzolás három dimenzióban, melynek sík vetülete buborék lerajzolást ad [31].

Grivet, Auber, Domenges és Melancon 2006-ban egy olyan algoritmust mutat be rendezett fákra, mely hasonlít az itt bemutatott buborék lerajzolásra [17]. Grivet-ék algoritmusa is minden gyermeket egy meghatározott szögtartományban helyez egy a szülő csúcs körül úgy, hogy a gyermekek részfájának befoglaló köre ne messe egymást, viszont a gyermekek nem lesznek pontosan ugyanakkora távolságra a szülőjüktől. Az éleket sem egyenes vonallal ábrázolja, hanem megenged mindegyik élen legfeljebb egy töréspontot. Az algoritmust implementálták, és egy több mint 270000 csúcsból álló Linux fájlrendszert

reprezentáló gráfon összehasonlították a saját algoritmusukat a szintezett lerajzolás Walker féle változatával [36], illetve a Eades által bemutatott sugárirányú lerajzolással [15]. Az eredmények alapján elmondhatjuk, hogy a gyakorlatban Grivet-ék algoritmusa jobb a szögfelbontású rajzot ad, mint a másik két algoritmus. Gyakorlati szempontból még két előnyt emelnek ki. Egyrészt, ha a gráfon kis változtatást végzünk, akkor a lerajzoláson is csak kis különbségek keletkeznek. Másrészt könnyen beazonosítható a rajzon, ha két részgráf izomorf egymással.



6.18. ábra. 2700 csúcsból álló linux fájlrendszer ábrázolása. Baloldalt Grivet-ék buborék lerajzolása, jobboldalt fent Eades sugárirányú lerajzolása, jobboldalt lent Walker szintezett lerajzolása. [17]

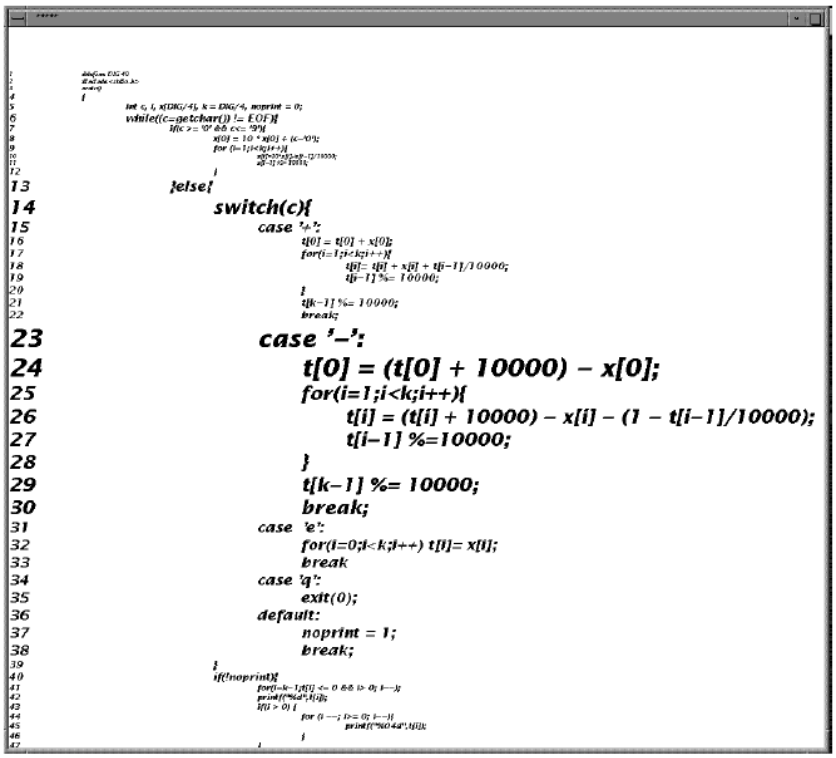
6.7. Alkalmazások

A buborék lerajzolás különböző változatait alapvetően nagy - akár több ezer csúcsú - gráfok megjelenítésére használják, például a már említet fájl rendszerek hierarchiájának bemutatására, illetve weboldalak struktúrájának ábrázolása. Másik tipikus alkalmazása a már sokszor említett elmetérképek lerajzolása.

Egy érdekes alkalmazást mutatott be Yun és Lin 2007-ben [25] cikkben, ahol programkódok vizualizálására használják a buborék lerajzolás elvét. Ma a legtöbb programozó valamilyen szövegszerkesztőt használ a programozáshoz. Ennek egyik előnye, hogy könnyebben érthető kódolni bennük, mert lehetőség van a kódsorokat valamilyen szempontból egyszerűen átláthatóbbá tenni. A program hierarchikus felépítését jobban érthetővé tenni magas prioritású feladat. Ehhez az kell, hogy a lényeges információkat kiemeljük a kevésbé lényegesektől.

A cikkben kódsorok betűméretét változtatják aszerint, hogy mennyire lényeges egy sor. Kitüntetnek egy sort, aminek az elhelyezkedését, szerepét szeretnénk vizsgálni az adott programban. Ezt a sort nevezik fókusznak. Először a programkódhoz egy gyökeres fát készítenek a kód egymásba ágyazott

struktúrájának megfelelően, melyben minden csúcshoz egy sor tartozik. Ezután a fa csúcsaihoz egy szám értékeket rendelnek melyben figyelembe veszik azt is, hogy melyik csúcshoz tartozik a fókusz. Cikkükben három algoritmust hasonlítanak össze bemutatva az egyes algoritmusok előnyeit és hátrányait. Az összehasonlított algoritmusok közül az egyik Furnas halszemes lerajzolásán alapul [16], másik a fraktál módszeren - ez nem azonos a korábban bemutatott fraktál modellel [22] - a harmadik pedig a buborék lerajzoláson alapuló eljárás. Megjegyezzük, hogy a buborék és a fraktál módszerekkel néhol túl nagy, illetve túl kicsi a szórása a meghatározott értékeknek, így a tényleges alkalmazásnál logaritmikus vagy exponenciális függvénnyel átskálázzák az értékeket.



6.19. ábra. Programkód vizualizálás a fraktál módszerrel. Fókusz a 24. soron [22].

7. fejezet

Nehéz utas lerajzolás

Sok alkalmazás keres olyan lerajzolást, mely egyszerre teljesíti a következő esztétikákat:

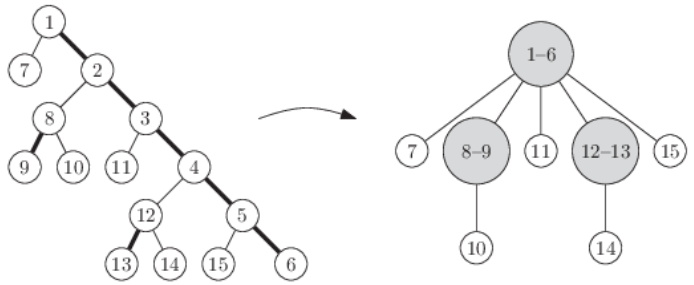
1. egyenesvonalas rajz,
2. keresztezés mentes rajz,
3. polinomiális területű,
4. tökéletes szögfelbontású.

tökéletes szögfelbontású Duncan és Eppstein elsőnek ír le egy olyan algoritmust, amely egyszerre teljesíti a fenti esztétikákat [14]. Eredményük meglehetősen friss, 2011-ből származik. Ebben a fejezetben ezt az algoritmust ismertetjük. Az algoritmus hasonló alapokon nyugszik, mint a 6. fejezetben bemutatott buborék lerajzolás. Itt is minden v-re a v gyermekei egy v középpontú befoglaló körön belül helyezkednek el, de a buborék lerajzolással ellentétben nem feltétlenül vannak egy kör kerületén. Fontos megjegyezni, hogy ez az algoritmus csak nem rendezett fákra ad polinomiális területű lerajzolást.

7.1. Nehéz út felbontás

Az egész algoritmus az úgynevezett nehéz út felbontáson alapszik. Legyen T egy r gyökerű fa. Legyen tetszőleges v csúcs gyermeke u. Ekkor u csúcsot a v textbfnehéz gyermekének, vagy **nehéz csúcsnak** hívjuk, ha v minden más w gyermekére w gyökerű részfa elemszáma kisebb, mint az u gyökerű részfa elemszáma, azaz $|T_u| \geq |T_w|$, ha $u \neq w \in child(v)$. Ha több gyermeke is van v-nek, melyeknek a részfája maximális elemszámú, akkor lerögzítünk egyet, melyet a v nehéz gyermekének hívunk. Minden más csúcsot **könnyű csúcsnak**, vagy könnyű gyermeknek hívjuk. A v könnyű gyermekeinek halmazát jelöljük $K(v)$-val, és legyen $k(v) = 1 + \sum_{u \in K(v)} |T_u|$, melyet a v csúcs **könnyű méretének** hívunk. Egy $u = p(v)v$ él **nehéz él**, ha a v nehéz csúcsot köt össze a szülőjével, különben könnyű él. A csak nehéz élekből álló utat **nehéz útnak** nevezzük. Képezzünk egy gráfot a T-hez úgy, hogy a maximális nehéz utakat összehúzzuk egy-egy csúccsá. Így nyilván egy fát kapunk, melyben a nehéz utakhoz tartozó csúcsok között akkor vezet él, ha a két nehéz út között T-ben van egy könnyű él. Ezen kívül még "lóghatnak" le könnyű élen levelek a nehéz utakból kapott csúcsokból. Ezt a fát nevezzük a T **nehéz út felbontásának**, és $N(T)$-vel jelöljük (lásd a 7.3 ábra).

7.1.1. Állítás. *Ha egy T fa elemszáma n, akkor a hozzá tartozó $N(T)$ nehéz út felbontás fa $h(N(T))$ magassága legfeljebb* $\log_2 n$.

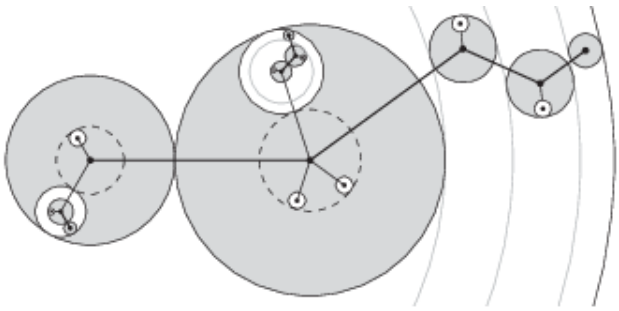


7.1. ábra. Példa nehéz út felbontásra. Baloldalt az eredeti gráf, benne vastag vonallal a nehéz élek, mellette a gráfhoz tartozó nehéz út felbontás fa [14].

Bizonyítás. Egy nehéz gyermek részfájának elemszáma legfeljebb feleannyi lehet, mint a szülőjének az elemszáma, különben az ő részfája minden más testvér csúcs részfájánál nagyobb lenne. Mivel $N(T)$-ben minden él T-ben könnyű élhez tartozik, ezért $N(T)$-ben egy levéltől a gyökérig vezető úton minden csúcsnál kétszereződik a csúcs részfájának elemszáma. Így legfeljebb csak $\log_2 n$ csúcsa lehet egy ilyen útnak, azaz a fa magassága is legfeljebb ennyi lehet. □

7.2. Algoritmus

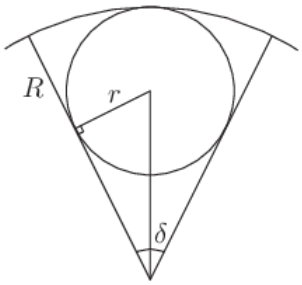
A lerajzolás előkészítéséhez a T fa egy postorder bejárásával elkészítjük az $N(T)$ nehéz út felbontását. A T lerajzolása az $N(T)$ postorder bejárásának sorrendjében történik. Legyen adva egy $P = (v_1, v_2, .., v_k)$ nehéz út. Minden v_i csúcs egy D_i kör középpontjába fog kerülni. A D_i körbe kerülnek a v_i könnyű gyermekeihez tartozó részfák rajzainak befoglaló körei, melyek egészen pontosan v_i körül két koncentrikus körön helyezkednek el. Ennél a lépésnél figyelünk a tökéletes szögfelbontásra is úgy, hogy ekkor már a $v_i v_{i-1}$ és $v_{i+1} v_i$ élek irányát is meghatározzuk, csak a hosszukat határozzuk meg később. A következő lépésben pedig a D_i köröket helyezzük el úgy, hogy az első P -beli csúcs, v_1 körül koncentrikus körgyűrűk belsejébe kerüljenek. A P-hez tartozó rajz így végül egy D befoglaló körbe kerül. A [14] cikkben belátják, hogy D sugara lineáris a P elemszámában és exponenciális a P-hez tartozó leszármazottak számában.



7.2. ábra. Példa neház út felbontás lerajzolásra. [14].

Jelöljük az R sugarú kör δ szögű körszeletét (R, δ)-val. A későbbiekben szükségünk van az alábbi állításra:

7.2.1. Lemma. *A legnagyobb kör sugara , mely egy (R, δ) körszeletbe belefér, éppen* $R\dfrac{\sin(\frac{\delta}{2})}{1+\sin(\frac{\delta}{2})}$.



7.3. ábra. Az R sugarú, δ szögű körcikk és a legnagyobb kör, amely benne elfér [14].

A továbbiakban az egyszerűség kedvéért az $N(T)$ nehéz út felbontás fa $h_{N(T)}$ magasságát jelöljük h-val. Először lássuk az algoritmust egy a nehéz út gyermekeinek az elhelyezésére.

7.2.2. Lemma (Könnyű gyermekek elhelyezése). *Legyen v egy T-beli nehéz csúcs, mely az $N(T)$ fa j. szintjéhez tartozó úton van, legalább kettő a fokszáma, és legalább két nehéz él vezet ki belőle. Tegyük fel, hogy v minden $u \in K(v)$ könnyű gyermekéhez adva van egy D_u, $r_u = 2 \cdot 8^{h-j-1}|T_u|$ sugarú befoglaló kör, melynek u a középpontjában helyezkedik el, és tartalmazza a T_u lerajzolását. Ekkor van egy D kör, melyben elkészíthetjük v egy lerajzolását úgy, hogy a D középpontjában a v van, D-ben minden D_u kör benne van, és igazak a következők:*

1. *minden él v és $u \in K(v)$ között egy egyenes, mely csak a D_u kört metszi a könnyű gyermekekhez tartozó D_w befoglaló körök közül,*
2. *a v-hez tartozó nehéz élek nem metszik egyik D_u kört sem,*
3. *a D_u körök nem metszenek egymásba,*
4. *v körül tökéletes szögfelbontású a rajz, azaz v bármely két szomszédos éle közötti szög éppen $\frac{2\pi}{d(v)}$,*
5. *a két nehéz él közötti szög $\frac{2\pi}{3}$ és $\frac{4\pi}{3}$ közé esik,*
6. *a D kör sugara $r_v = 8^{h-j}k(v)$.*

Bizonyítás. Helyezzük el a v szülőjéhez tartozó nehéz élt vízszintesen azonos koordinátára v-vel, tőle balra. Rajzoljuk meg a D, r_v sugarú kört a v középpontból, és a t_1, t_2, ...,$t_{d(v)}$ félegyeneseket melyek között éppen $\frac{2\pi}{d(v)}$ szög van. A D_u befoglaló körök ezekre a félegyenesekre fognak kerülni a D körön belül. A nehézség a feladatban az, hogy hogyan helyezzük el a köröket úgy, hogy ne érjenek egymásba.

Legyen a legnagyobb D_u kör sugara R_{max}. Az algoritmus kettéosztja a D kört egy A körgyűrűre és egy B v középpontú körre melynek a sugara $R = r_v - R_{max}$. Azokat a köröket, melyek elférnek a B körön belül a t_i félegyeneseken úgy, hogy két egymás melletti ne messe egymást, azokat **kis köröknek** nevezzük és a B körön belül helyezzük el őket. Azokat amelyek nem férnek el ilyen módon, **nagy köröknek** hívjuk és ügyesen elhelyezzük az A körgyűrűn a félegyenesekre úgy, hogy ne messék egymást. A kis köröknek tehát bele kell férniük egy R sugarú $\frac{2\pi}{d(v)}$ szögű körszeletbe. A 7.2.1. lemma alapján ez azokra a D_u körökre igaz, melyeknek a sugarára $r_u \leq R\dfrac{\sin(\frac{2\pi}{d(v)})}{1+\sin(\frac{2\pi}{d(v)})}$. Ha a nagy köröket elhelyeztük, akkor a kis köröket tetszőlegesen elhelyezhetjük a t_i félegyeneseken. A nagy körök elhelyezéséhez be fogjuk látni, hogy van egy C kör, melynek az átmérőjére minden D_u nagy kör elfér úgy, hogy a körök középpontja a kör átmérőjére essen. Sőt, ez a kör belefér a D kör egy $(r_v, \frac{\pi}{4})$ körszeletébe, azaz D egy körnegyedébe. Mielőtt azonban ezt belátnánk, nézzük meg az algoritmus befejezését. Tegyük a C kört függőlegesen a v csúcs fölé úgy, hogy érintse a D kört és a D_u körök vízszintesen helyezkedjenek el a

C átmérőjén. Így minden nagy kör a v felett egy vízszintes egyenesen van, a D kör egy környegyedében. Tekintsük azt a v középpontú kört, melynek a sugara $r_v - R_{max}$. Ekkor, ha függőlegesen felfelé toljuk a D_u nagy köröket úgy, hogy a középpontjuk ennek a körnek a kerületére essenek akkor minden kör a $(r_v, \frac{\pi}{4})$ körszelet és az A körgyűrű metszetébe kerül. Ez már majdnem jó, már csak el kell helyeznünk a köröket úgy, hogy mindegyik egy t_i félegyenesre essen. Ehhez a v csúcs körül elforgatjuk a köröket az óramutató járásával megegyező irányban a v csúcs körül úgy, hogy a legbaloldaliabb kör középpontja az első legközelebbi félegyenesre essen. Majd azt a kört hátrahagyva a többi kört forgatjuk tovább úgy, hogy a sorban következő kerüljön egy félegyenesre. Ezt az eljárást folytatva elvben mindegyik nagy kört el tudjuk helyezni a félegyenesekre. Persze azt be kell látni, hogy így mindegyik nagy kör elfér. Mivel kezdetben minden kör egy $\frac{\pi}{4}$ fokos szögtartományban elfért, ezért ha belátjuk, hogy összesen legfeljebb $\frac{3\pi}{4}$ szöggel forgattuk el a köröket, akkor biztosan mindegyik kör el fog férni. Egy kört legfeljebb csak $\frac{2\pi}{d(v)}$ fokkal forgatunk el és összesen annyiszor forgattunk, ahány nagy kör van. Tehát, ha a nagy körök nincsenek "túl sokan", akkor el fognak férni.

7.2.3. Állítás. *A nagy körök n_N száma legfeljebb $\frac{3d(v)}{8}$, ha $d(v) \geq 5$.*

Tehát ha $d(v)$ legalább 5, akkor a fenti algoritmus helyes. Mivel kezdetben minden kör elfért egy $\frac{\pi}{4}$ fokos körszeletben, és legfeljebb $\frac{3\pi}{4}$ szöggel forgattuk el mindegyiket, összesen legfeljebb $\frac{3d(v)}{8}$-szor ezért ezt az eredeti körszeletet legfeljebb $\frac{\pi}{4} \cdot \frac{3d(v)}{8} = \frac{3\pi}{4}$ szöggel növelhettük. Tehát összesen π szöget, azaz az A körgyűrűnek legfeljebb a felét használhattunk el a nagy körök elhelyezésére.

Ha $d(v)$ ennél kisebb, akkor is könnyen elkészíthetjük a nehéz utas lerajzolást. Ha $d(v) = 2$, akkor nincsen könnyű gyermeke v-nek, így ekkor a két nehéz szomszédját vízszintesen helyezzük el, v szülőjét balra, v nehéz gyermekét jobbra. Ha $d(v) = 3$, akkor csak egy könnyű gyermek van és három félegyenes lesz. Ebből a jobb oldalra már lefixáltuk a nehéz szülőhöz tartozó élt. A könnyű gyermeket bármelyik félegyenesen elhelyezhetjük, az (5)-ös tulajdonság így teljesülni fog. Ha $d(v) = 4$, akkor a két könnyű gyermeket egymással szemben helyezzük el függőlegesen, a két nehéz szomszédot pedig vízszintesen és így biztosan nem fogják metszeni az élek egyik befoglaló kört sem.

A két nehéz él elhelyezése a $d(v) \geq 5$ esetben is egyszerűen megtehető. A szülőhöz vezető élt már lefixáltuk a v-től balra vízszintesen vezető félegyenessel. Mivel az A körgyűrűnek legfeljebb a felét használtuk fel, ezért megtehetjük, hogy a megmaradt nehéz élt a másikkal pontosan szemben helyezzük el, ha $d(v)$ páros. Ha viszont $d(v)$ páratlan, akkor a π-hez legközelebb eső t_i, t_{i+1} félegyenesek közül választhatunk tetszőlegesen.

Azt még be kell látnunk, hogy van olyan C kör, melynek az átmérőjén minden nagy kör elfér. Igazából mi olyan kört találunk, amire minden könnyű gyermekhez tartozó befoglaló kör ráfér. Ugyanis figyeljük meg, hogy:

$$4 \sum_{u \in K(v)} r_u = 4 \sum_{u \in K(v)} 2 \cdot 8^{h-j-1} |T_u| = 8^{h-j} \sum_{u \in K(v)} |T_u| < 8^{h-j} k(v) = r_v$$

Azaz egy $\frac{r_v}{4}$ sugarú kör átmérőjén az összes D_u kört elhelyezhetjük. Rendezzük sorba a D_u köröket nagyság szerint csökkenő sorrendbe a kör átmérőjén. Így készíthetünk két kört, melyek közül az egyik csak a nagy köröket tartalmazza a másik pedig csak a kis köröket. Legyen a C kör a nagy köröket tartalmazó kör. Mivel ennek a körnek a sugara is legfeljebb $\frac{r_v}{4}$, ezért a 7.2.1. lemma alapján a belefér egy $(r_v, \frac{\pi}{4})$ körszeletbe.

A fenti algoritmussal tehát megkaptuk, hogy hogyan lehet elhelyezni egy nehéz gyermek körül a könnyű gyermekek részfáinak rajzait úgy, hogy a lemma (1-6) tulajdonságai teljesüljenek.

□

Megjegyezzük, hogy a fenti algoritmus működik a T fa gyökerére is, ha a szülő élet nem rajzoljuk be. Könnyen elkészíthetjük egy olyan $P = (v_1, v_2, ..., v_k)$ nehéz úton a v_1 csúcs rajzát is, ahol a P-hez tartozó csúcs szintje az $N(T)$ fában legalább egy. Itt a v_1 szülőjébe vezető $p(v_1)v_1$ él könnyű él, de a lemmát így is alkalmazzuk, mintha ez lenne a másik nehéz él. Végül bármely nehéz út utolsó csúcsa minden esetben egy levél lesz, így a rajza triviális.

A feladat most az, hogy a nehéz csúcsok tartalmazó körök rajzait összekössük egymással.

7.2.4. Lemma (Nehéz út lerajzolása). *Legyen adva egy $P = (v_1, v_2, .., v_k)$ nehéz út és minden v_i csúcsának lerajzolása a 7.2.2. lemma tulajdonságaival. Jelöljük az egyes v_i csúcsok befoglaló köreit D_i-vel, melyek sugarát jelöljük r_i-vel. Ekkor van egy r sugarú D kör, melyben elkészíthetjük a P-nek és minden leszármazottjának egy lerajzolását úgy, hogy D középpontjában v áll és melyre a következők igazak:*

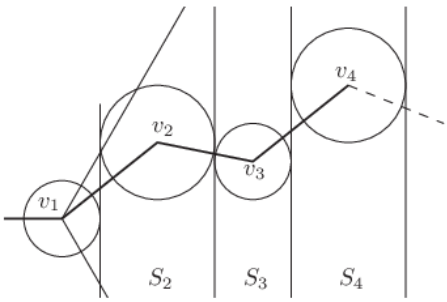
1. *minden v_iv_{i+1} él egy egyenes, mely csak a D_i és D_{i+1} köröket metszi.*
2. *a v_1-et a $p(v_1)$ szülőjével összekötő könnyű él a D_1 kör kivételével nem metszi P rajzát,*
3. *a D_i körök nem metszik egymást,*
4. *a rajz minden v_i csúcsa körül tökéletes szögfelbontású,*
5. *a D kör sugara $r = 2\sum_{i=1}^{k} r_i$.*

Bizonyítás. Rajzoljuk meg a D kört az origóba, és helyezzük a középpontjába a v_1 csúcsot úgy, hogy az ő $p(v_1)$ szülője v-től vízszintesen balra essen. Legyenek S_2, S_3, ..., S_k olyan függőleges sávok D_i-től jobbra egymás mellett, amelyekre az i. sáv szélessége $2r_i$, és az S_2 baloldali határoló egyenese érinti D_1-et. Legyen továbbá f_i az i. sávot felező egyenes. Hosszabbítsuk meg azt az élet, melyet a v_1 nehéz gyermekének fixáltunk le a 7.2.2. lemma algoritmusában, egészen addig, amíg metszi az f_2 felező egyenest. Erre a metszéspontra kerül a v_2 csúcs, és így a D_2 kör teljes egészében az S_2 sávban helyezkedik el (lásd 7.4. ábra). Mivel a v_1 körül úgy helyeztük el az éleket, hogy a két nehéz él közötti szög az $\frac{2\pi}{3}$ és $\frac{4\pi}{3}$ közé essen, ezért a v_2 csúcs egy $\frac{2\pi}{3}$ szögtartományba esik, mely szimmetrikus az x tengelyre. Hasonlóan az előzőekhez, minden v_i csúcs nehéz gyermekéhez tartozó élt hosszabbítsuk meg addig, amíg nem metszi az S_{i+1}-et felező f_{i+1} egyenest, és erre a metszéspontra helyezzük a v_{i+1} csúcsot. Ehhez biztosítanunk kell, hogy a nehéz él meghosszabbítása mindig messe a következő sáv felező egyenesét. Ehhez elég, hogy ha minden v_i-re a v_i nehéz gyermekhez tartozó félegyenes a W szögtartományban marad. A 7.2.2. lemma alapján a v_i csúcshoz tartozó két nehéz él közötti szög $\frac{2\pi}{3}$ és $\frac{4\pi}{3}$ közé esik, ezért ha a v_i rajzát, vagy annak a $v_{i-1}v_i$ él egyenesére vett tükörképét helyezzük a D_i kör belsejébe, akkor a v_i nehéz gyermekéhez tartozó él a W szögtartományba fog esni. Ezen kívül a v_i csúcsok koordinátái egy x koordinátában növekvő sorozatot fognak alkotni.

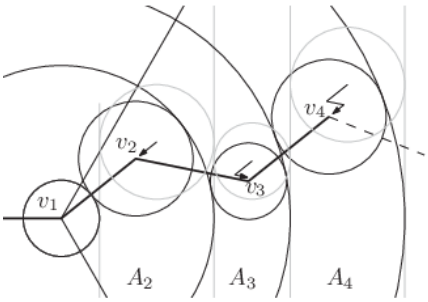
Az így elkészült rajzra igaz, hogy bármely két D_i, D_j kör nem metszi egymást, hiszen külön sávokban helyeztük el őket, melyek szeparálják egymástól a köröket, így teljesül a (3)-as tulajdonság. Egy v_iv_{i+1} él egyenes vonal két szomszédos sáv között, így nem fog belemetszeni más körbe csak a D_i-be és D_{i+1}-be ((1)-es tulajdonság). Mivel a v_i csúcsok körül az élek egymáshoz való viszonyát nem változtattuk meg, ezért megmarad a tökéletes szögfelbontás ((4)-es tulajdonság). Végül, mivel a v_1 és $p(v_1)$ közötti él egy

vízszintes egyenes v_1-től balra, így minden S_i sávtól balra esik és így nem metszi egyik D_j kört sem D_1-en kívül ((2)-es tulajdonság). A rajz szélessége pont a D_i körök átmérőinek összege, azaz $2\sum_{i=1}^{k} r_i$, de ez nem feltétlenül fér bele egy v középpontú körbe, melynek a sugara szintén az átmérők összege. Ahhoz, hogy ezt elérjük, készítünk $k-1$ körgyűrűt, A_2, A_3, ..., A_k-t ahol az A_i szélessége éppen $2r_i$. Ezután a v_2-vel kezdve egyesével minden v_iv_{i+1} él hosszát lecsökkentjük, vagy meghosszabbítjuk annak érdekében, hogy a D_{i+1} kör teljes egészében az A_{i+1} körgyűrűbe kerüljön. Egy ilyen transzformáció során az összes v_{i+1} utáni $(v_{i+2}, v_{i+3}, ..., v_k)$ utat és a hozzá tartozó D_{i+2}, D_{i+3}, ..., D_k köröket is elmozgatjuk pont akkora és ugyanolyan irányú és hosszúságú vektorral, mint amellyel a v_iv_{i+1} élet megrövidítettük, vagy meghosszabbítottuk (lásd 7.5 ábra). Ezt úgy is elképzelhetjük, mintha a $(v_{i+2}, v_{i+3}, ..., v_k)$ út és a hozzájuk tartozó D_{i+2}, D_{i+3}, ..., D_k körök egy szilárd szerkezetet alkotnának, és azt egyben tudnánk mozgatni a v_iv_{i+1} él hosszának megváltoztatásakor. Végül minden D_i kör az A_i körgyűrűbe fog kerülni úgy, hogy annak a két határoló körét érinti. Mivel a körgyűrűk diszjunktak, ezért a körök diszjunktsága megmarad. Az élek hosszainak változtatása nem változtatja meg az élek irányát, így a többi tulajdonság is érvényben marad. Így végül a teljes rajz belefért egy $r_1 + 2\sum_{i=2}^{k} r_i < 2\sum_{i=1}^{k} r_i$ sugarú körbe és ezzel az (5)-ös tulajdonságot is teljesítettük.

□



7.4. ábra. Az v_i köröket S_i sávokban helyezzük el. [14].



7.5. ábra. Az S_i sávokból az A_i körgyűrűkbe mozgatjuk egyesével a köröket. Egy lépésben az összes sorban balra lévő kört is mozgatjuk. [14].

Végül a 7.2.2. és a 7.2.4. lemmák alkalmazásával megkaphatjuk egy tetszőleges fa nehéz utas lerajzolását.

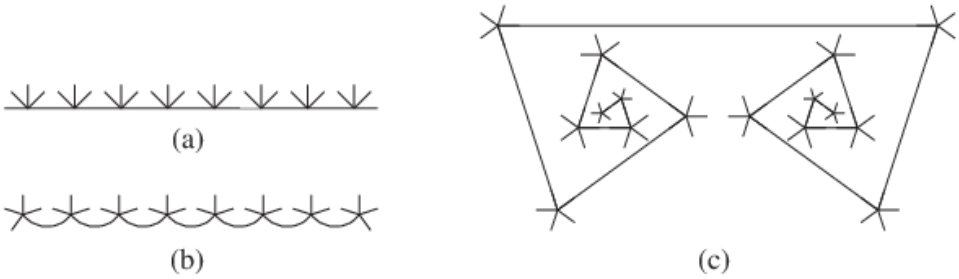
7.2.5. Tétel. *Legyen T egy nem rendezett fa n csúccsal. Ekkor tudunk készíteni T-hez egy olyan lerajzolást, mely teljesíti a következőket:*

1. *egyenesvonalas rajz,*

2. *keresztezés mentes,*
3. *tökéletes szögfelbontású*
4. *belefér egy legfeljebb $2 \cdot 8^h n$ sugarú D körbe, ahol h-val jelöljük a T nehéz út felbontásának magasságát. Mivel a 7.1.1. állítás alapján $h \leq \log_2 n$, ezért a D sugara legfeljebb $2n^4$.*

Bizonyítás. Legyen $P = (v_1, v_2, ..., v_k)$ egy tetszőleges nehéz út a nehéz út felbontás j. szintjén. Ekkor a 7.2.2. lemma alapján minden csúcs P-ben lerajzolható egy legfeljebb $r_v = 8^{h-j}k(v)$ sugarú körbe. A 7.2.4. lemma alapján akkor P lerajzolható egy $2\sum_{i=1}^{k} 8^{h-j}k(v_i) = 2 \cdot 8^{h-j} \sum_{i=1}^{k} 1 + \sum_{u \in K(v_i)} |T_u| = 2 \cdot 8^{h-j} n(P)$, ahol $n(P)$-vel jelöljük a P út és az összes leszármazottjának az elemszámát. Mivel ez minden P nehéz útra igaz, ezért az $N(T)$ nehéz felbontás gyökerére is, melynek a leszármazottjai az összes csúcsot tartalmazza. Így igaz az állítás. □

7.2.6. Megjegyzés. *A fenti 7.2.5. tétel (1-3) tulajdonságát teljesítő lerajzolást polinomiális területtel csak nem rendezett fák esetében tudunk készíteni. Vegyük például azt a gráfosztályt, mely egy útból áll, és annak minden belső csúcsán három további levélcsúcs lóg le. Így az út két végpontján kívül minden csúcs fokszáma 5. A csúcsok rendezése olyan, hogy a három levél él minden csúcsnál ugyanarra az oldalra kerüljön. Ennek a tökéletes szögfelbontású, keresztezés mentes rajza mindenképpen egy spirál. A legkisebb területű lerajzolás egy szimmetrikus dupla spirál. Ennek a területe viszont exponenciális a gráf csúcsszámában (lásd 7.6. ábra).*



7.6. ábra. Példa nem rendezett fára, melynek egyenesvonalas, tökéletes szögfelbontású rajza csak exponenciális területű lehet. (a) a kiinduló gráf, (b) a gráf Lombardi lerajzolása, (c) a gráf egyenesvonalas rajza tökéletes szögfelbontással [14].

7.3. További eredmények

A 7.2.6. megjegyzés példája alapján rendezett fákra nem lehet keresztezés mentes egyenesvonalas lerajzolást készíteni és tökéletes szögfelbontással polinomiális területtel. Ha viszont elhagyjuk azt a feltételt, hogy egyenesvonalas rajzot készítsünk, akkor készíthetünk polinomiális területű, tökéletes szögfelbontású rajzot. A [14] cikkben még egy nehéz út felbontáson alapuló lerajzolást is bemutatnak, melyet **Lombardi lerajzolásnak** neveztek el Mark Lombardi művész munkái alapján (Lombardi kézzel rajzolt szociális hálókat bemutató ábrákat) [18]. Ebben a lerajzolásban az élek nem egyenesek, hanem körívek lesznek. Ez esetben a szögfelbontást úgy értelmezzük, hogy az adott csúcsnál lévő élekhez vett érintők által bezárt szögek legyenek pontosan $\frac{2\pi}{d(v)}$ nagyságúak. A cikkben belátják, hogy a Lombardi lerajzolással ilyen értelemben tökéletes szögfelbontású rajzot kapunk, mely $O(n^3)$ területű.

Összegzés

A dolgozat céljának azt fogalmaztuk meg, hogy megismerjük és megismertessük a fák lerajzolásának főbb módszereit. Az eljárások gyűjtése közben kiderült, hogy a gyakorlatban az algoritmusok nem feltétlenül épülnek alapjaiban különböző ötletekre, hanem egy ötletnek rengeteg különböző változatát alkalmazzák. Az algoritmusok sokszor nincsenek az egyes irodalmakban pontosan kidolgozva, például az ötödik fejezetben tárgyalt buborék lerajzolás esetében sem találunk minden fára működő pontosan megfogalmazott lerajzolást. A módszerek gyakran az egyes alkalmazásokban szereplő gráfokra specifikusan működnek jól. Sokszor egy matematikailag nagy jelentőségű lerajzolási módszer kevésbé alkalmazható a gyakorlatban. Ennek oka, hogy az alkalmazások általában több esztétikai feltételnek akarnak egyszerre megfelelni, amelyeket nem lehet egyszerre optimálisan teljesíteni. Ha pedig egy esztétikai szempont szerint optimalizálunk, akkor gyakran más feltételek csorbulnak oly mértékben, hogy az a gyakorlatban már nem kielégítő. Végül sokszor tapasztalati úton határozzák meg, hogy az adott lerajzolás közül melyik a legmegfelelőbb az információk bemutatására, melyet a gráf magában hordoz.

Köszönetnyilvánítás

Szeretném megköszönni témavezetőm, Bérczi Kristóf munkáját. Tanácsai, bátorítása és útmutatása nagy mértékben hozzájárult a dolgozat elkészítéséhez. Maximálisan támogatta munkámat, és nagyon sokat tanultam tőle.

Szeretnék még köszönetet mondani családomnak és barátaimnak a sok támogatásért és bátorításért, amit munkám során nyújtottak.

Irodalomjegyzék

[1] http://www.csd.uoc.gr/ hy583/papers/ch8.pdf. 9, 11, 15, 17, 25

[2] G. article and N. Keshary. Radial tree graph drawing algorithm for representing large hierarchies. *University of Connecticut*, 2001. 20

[3] M. Bernard and S. Mohammed. Labeled radial drawing of data structures. pages 479–484, 2003. 20

[4] M. A. Bernard. On the automated drawing of graphs. pages 43–55, 1981. 20

[5] A. Bloesch. Aesthetic layout of generalized trees. *Software: Practice and Experience*, 23(8):817–827, 1993. 15

[6] C. Buchheim, M. Jünger, and S. Leipert. Improving walker Š s algorithm to run in linear time. pages 344–353, 2002. 15

[7] c. Wetherell and A. Shannon. Tidy drawings of trees. *Software Eng. on*, pages 514–520, 1979. 14

[8] J. Carriere and R. Kazman. Research report. interacting with huge hierarchies: beyond cone trees. pages 74–81, 1995. 36, 50

[9] T. M. Chan, M. T. Goodrich, S. R. Kosaraju, and R. Tamassia. Optimizing area and aspect ratio in straight-line orthogonal tree drawings. *Computational Geometry*, 23(2):153–162, 2002. 33

[10] E. H. Chi, P. Pirolli, and J. Pitkow. The scent of a site: A system for analyzing and predicting information scent, usage, and usability of a web site. pages 161–168, 2000. 20

[11] E. H. Chi, J. Pitkow, J. Mackinlay, P. Pirolli, R. Gossweiler, and S. K. Card. Visualizing the evolution of web ecologies. pages 400–407, 1998. 20

[12] P. Crescenzi, G. Di Battista, and A. Piperno. A note on optimal area algorithms for upward drawings of binary trees. *Computational Geometry*, 2(4):187–200, 1992. 27

[13] P. Crescenzi and P. Penna. Minimum-area hv drawings of complete binary trees. pages 371–382, 1997. 27

[14] C. A. Duncan, D. Eppstein, M. T. Goodrich, S. G. Kobourov, and M. Nöllenburg. Drawing trees with perfect angular resolution and polynomial area. pages 183–194, 2011. 55, 56, 57, 60, 61

[15] P. Eades. Drawing free trees. 1991. 20, 52

[16] G. W. Furnas. Generalized fisheye views. 17(4), 1986. 53

[17] S. Grivet, D. Auber, J.-P. Domenger, and G. Melancon. Bubble tree drawing algorithm. pages 633–641, 2006. 51, 52

[18] R. C. Hobbs and M. Lombardi. Mark lombardi: Global networks. 40, 2003. 61

[19] C.-S. Jeong and A. Pang. Reconfigurable disc trees for visualizing large hierarchical information space. pages 19–25, 1998. 51

[20] T. Kamada. Visualizing abstract objects and relations. 1989. 27

[21] S. K. Kim. Hv drawings of binary trees. *Software Visualisation*, 7:101, 1996. 27

[22] H. Koike. Fractal views: a fractal-based method for controlling information display. *ACM Transactions on Information Systems (TOIS)*, 13(3):305–323, 1995. 53

[23] H. Koike and H. Yoshihara. Fractal approaches for visualizing huge hierarchies. pages 55–60, 1993. 51

[24] C. Kuratowski. Sur le probleme des courbes gauches en topologie. *Fundamenta mathematicae*, 15(1):271–283, 1930. 1

[25] C.-C. Lin and H.-C. Yen. Balloon views of source code and their multiscalable font modes. pages 53–58, 2007. 52

[26] C.-C. Lin and H.-C. Yen. On balloon drawings of rooted trees. *J. Graph Algorithms Appl.*, 11(2):431–452, 2007. 7, 35, 36, 37, 43, 46, 48, 50

[27] K. Marriott and P. Sbarski. Compact layout of layered trees. 62:7–14, 2007. 15, 16

[28] K. Marriott and P. J. Stuckey. Np-completeness of minimal width unordered tree layout. *J. Graph Algorithms Appl.*, 8(2):295–312, 2004. 15

[29] G. Melancon and I. Herman. Circular drawings of rooted trees. *Information Systems (INS)*, 1998. 51

[30] E. M. Reingold and J. S. Tilford. Tidier drawings of trees. *Software Engineering, IEEE Transactions on*, pages 223–228, 1981. 9, 14

[31] G. G. Robertson, J. D. Mackinlay, and S. K. Card. Cone trees: animated 3d visualizations of hierarchical information. pages 189–194, 1991. 51

[32] C.-S. Shin, S. K. Kim, and K.-Y. Chwa. Area-efficient algorithms for straight-line tree drawings. *Computational Geometry*, 15(4):175–202, 2000. 33

[33] T. H. Smith. Graphic statistics in management. 1924. 21

[34] S. T. Teoh and M. Kwan-Liu. Rings: A technique for visualizing large hierarchies. pages 268–275, 2002. 49

[35] I. Tollis, P. Eades, G. Di Battista, and L. Tollis. Graph drawing: algorithms for the visualization of graphs. 1, 1998. 6, 7, 14, 17, 20, 23, 24, 29, 30, 31, 32

[36] J. Q. Walker. A node-positioning algorithm for general trees. *Software: Practice and Experience*, 20(7):685–705, 1990. 15, 52