Turinys

Anotacija

Šio darbo tikslas – sukurti prototipą, kurio pagalba būtų galima atlikti efektyvų viename kompiuteryje esančių skatmeninių vaizdų paieškos rezultatų pateikimą subjektui. Prototipo veikimas yra paremtas skaitmeninių vaizdų įverčių skaičiavimo algoritmu. Pagrindinis akcentas – tai skaitmeninių vaizdų paieškos rezultatų vertinimas bei prioretizavimas. Prioretizacija vykdoma apskaičiuojant kiekvienam skaitmeniniui vaizdui įvertį, kuris nurodo pastarojo svarbą rezultatų aibės kontekste. Įverčio paieška vykdoma naudojant panašumo grafą bei taikant jame tikrinio vektoriaus centriškumo algoritmą. Darbo metu aptarti algoritmo privalumai bei trūkumai. Pateikiamos prielaidos, kurios leistų optimizuoti pastarojo veikimą. Optimizacija yra paremta daugiamačių duomenų indeksavimo spartos gerinimu. Taip pat siūlomas techninis sprendimas leidžiantis efektyviai naudoti sukurtąjį prototipą realioje operacinėje sistemoje. Darbe siūloma naudoti KD medžius. Tai alternatyva *Henry Rowly*, *Shumeet Baluja* ir *Yushi Jing* pasiūlytiems kamuoliniams medžiams. Darbo rezultatas – prototipas, kurio pagalba galima atlikti efyktyvų skaitmeninių vaizdų paieškos rezultatų pateikimą subjektui. Pasinaudojant pastarąja prgrama atlikta praktinė analizė. Tyrimo metu nustatyta, kad KD medžių panaudojimas ne tik supaprastina algoritmą, bet ir leidžia įvykdyti viename kompiuteryje esančių skaitmeninių vaizdų paiešką greičiau nei naudojant kamuolinius medžius, o sukurtojo prototipo panaudojimas leidžia efektyviai prioretizuoti bei pateikti rezultatus. Skaitmeniniai vaizdai surūšiuojami interpretuojant, kurie iš jų labiausiai tikėtinai yra užklausą tenkinantys skaitmeniniai vaizdai.

Summary

Computing Image Ranks by Applying Graph Algorithms

In this paper the problem of image search is addressed. The primary goal of the paper is to present a prototype that can be used to identify and effectively return search results to the end user (i.e., return the most relative images to some given search criteria). All the search is performed to find images stored in one personal computer. To accomplish this task an image rank algorithm is utilized. Algorithm's advantages and disadvantages are pointed out. As a result an optimization issue is discussed and a way to increase the performance of multidimensional data indexing is presented. The optimization is based on KD trees. The assumption is that these trees might work faster than ball trees. Those were proposed by authors of the original image rank algorithm. The result is a prototype that is used for searching and presenting images based on ranks that indicate relative importance of each of the image to some given search criteria. The results are prioritized. The prioritization might be thought of as sorting images by their similarities. By using the application a practical analysis was performed and the results were described in details. It was found that KD trees can be effectively used to index multidimensional data while calculating image ranks. It not only simplifies the algorithm but also provides better performance results than ball trees.

Įvadas

Paieškos sitemose, tokiose kaip Google, Yahoo, Bing ir pan., atliekant skaitmeninių vaizdų paiešką, užklausos yra vykdomos analizuojant tinklalapyje esančią tekstinę informaciją, tokią kaip puslapio turinys, antraštės, skaitmeninių vaizdų pavadinimai, meta duomenys (duomenys apie duomenis, kurie vartotojui dažniausiai yra nematomi) ir kt. Analogiškai duomenų paieška vykdoma ir asmeniniuose kompiuteriuose. Daugelyje šiuolaikinių operacinių sistemų tam yra skirta paieškos (angl. search) funkcija. Vartotojas įveda raktinį žodį, o sistema gražina visas bylas, kurios tenkina paieškos kriterijų. Rezultatai priklauso nuo įvairios tekstinės infomacijos, susietos su sistemoje esančiomis bylomis ar katalogais. Abejais minėtaisiais atvejais esminė problema yra tai, jog nekreipiamas dėmesys į skaitmeninių vaizdų turinį. Todėl yra didelė tikimybė, jog subjektui (vartotojui, programinei įrangai) pateikus paieškos užklausą, ne visi rezultatų aibės elementai atitiks jo lūkesčius.

Skaitmeninių vaizdų paieška, kuri yra paremta lyginamaja analize, pateikia tikslesnius rezultatus. Pastaruoju atveju analizuojami skaitmeniai objektai, o ne vien tekstinė informacija, esanti tame pačiame puslapyje, kuriame buvo aptiktas skaitmeninis vaizdas (jei paieška vykdoma viename kompiuteryje – informacija, susieta su pačia byla).

Bendruoju atveju skaitmeninių vaizdų paieškos, paremtos lyginamąja anilze, algoritmas susideda iš pradinės skaitmeninių vaizdų aibės radimo (tekstinė paieška), skaitmeninių vaizdų analizės bei gautųjų rezultatų pateikimo subjektui. Pateikiant užklausos rezulatus reikia nustatyti, kuris skaitmeninis vaizdas, esantis gautojoje rezultatų aibėje, tiksliausiai atitinka nurodytąją užklausą (aktualiausia informacija). Procesas, kurio metu atliekamas rezultatų įvertinimas remiantis skaitmeninių vaizdų svarba (šiuo atveju tai atitiktų skaitmeninių vaizdų prioretizavimą remiantis jų turiniu) vadinamas skaitmeninio vaizdo įverčio (angl. image rank) radimu [3, 4].

Henry Rowly, *Shumeet Baluja* ir *Yushi Jing* savo darbuose [3,4] nagrinėjo nekorektiškų paieškos rezultatų (semantiniu požiūriu) problemą bei pasiūlė jos sprendimo būdą. Šiame darbe tęsiamas minėtųjų žmonių darbas. Pastarieji gilinosi į problemos sprendimą saityno kontekste. Šiame darbe algoritmas taikomas viename kompiuteryje esantiems skaitmeniniams vaizdams aptikti bei efektyviai pateikti sistemos naudotojui, t.y., pakeista algoritmo taikymo sritis.

Esminiai darbo metu atlikti darbai:

1. Išanalizuotas *Henry Rowly*, *Shumeet Baluja* ir *Yushi Jing* pasiūlytas saityno paieškos

sistemoms pritaikytas skaitmeninių vaizdų įverčių skaičiavimo algoritmas.

2. Pakeista algoritimo taikymo sritis. Algoritmas buvo pritaikytas viename kompiuteryje esantiems skaitmeniniams vaizdams aptikti, prioretizuoti bei atitinkamai pateikti.
3. Realizuotas paieškos sistemos prototipas. Praktiškai panaudotas prototipas atliekant tiriamąją analizę.
4. Pasiūlytas techninis sprendimas leidžiantis efektyviai naudoti sukurtąjį prototipą realioje operacinėje sistemoje.
5. Modifukuotas *Henry Rowly*, *Shumeet Baluja* ir *Yushi Jing* pasiūlytas skaitmeninių vaizdų įverčių skaičiavimo algoritmas. Pasiūlyta daugiamačių duomenų indeksavimui naudoti KD [JUO07] medžius. Pateikta palyginamoji analizė, siekiant parodyti, kodėl viename kompiuteryje esantiems skaitmeniniams vaizdams aptikti KD medžiai gali būti naudingesni nei kamuoliniai medžiai. Tyrimo metu nustatyta, kad KD medžių panaudojimas ne tik supaprastina algoritmą, bet ir leidžia įvykdyti viename kompiuteryje esančių skaitmeninių vaizdų paiešką greičiau nei naudojant kamuolinius medžius.
6. Aprašyti gautieji rezultatai.

1. Sąvokos

Norint tinkamai suprasti skaitmeninių vaizdų įverčio skaičiavimo algoritmą, pirmiausia reikėtų susipažinti (prisiminti) su pagrindinėmis sąvokomis, kurios yra plačiai naudojamos šiame darbe.

1.1. Grafas

Grafas – tai aibių pora $G=(V, E)$, čia V - viršūnių aibė, E - viršūnių nesutvarkytųjų porų $e:=(u, v)=:uv=vu$, $u, v \in V$ arba briaunų (lankų) aibė. Kai poros $(u, v) \in E$ laikomos sutvarkytosiomis, G vadinamas digrafu. Viršūnės u ir v vadinamos $(u, v) \in E$ briaunos galais arba jai incidenčiomis viršūnėmis. Viršūnė u yra vadinama briaunos pradžia, o viršūnė v – pabaiga. Jos yra vadinamos gretimomis. Dydis $|V|=n$ yra vadimas grafo eile, o dydis $|E|=m$ yra vadinamas grafo didumu [Man98].

Grafas yra vadinamas paprastuoju jei bet kurias dvi grafo viršūnes $u, v \in V$ jungia tik viena briauna $(u, v) \in E, u, v \in V$. Jei viršūnes $u, v \in V$ jungia daugiau nei vieną briauną, tai toks grafas yra vadinamas multigrafu. Jeigu paprastasis grafas turi kilpų (kilpa – tai briauna arba lankas, kurios pradžią ir pabaigą atitinka ta pati viršūnė, t.y. $(u, v) \in E, u, v \in V, u=v$) tai jis yra vadinamas pseudo grafu.

Grafo realizacija plokštumoje vadinama kuri nors geometrinė schema (brėžinys), kurioje viršūnės pažymėtos taškais, o briaunos (lankai) $(u, v) \in E, u, v \in V$ kreive, jungiančia viršūnes u ir v. Jeigu grafas yra orientuotas (digrafas), tuomet kiekvienai briaunai nurodoma kryptis (grafiškai tai atitinka rodyklė). Grafas turi daug skirtingų realizacijų. Schemos, kurios yra to paties grafo realizacijos, vadinamos izomorfinėmis [Nor07].

1.2. Gretimumo matrica

Gretimumo matrica naudojama apibrėžti grafo viršūnių tarpusavio ryšius. Remiantis ja galima nusakyti, kurios grafo viršūnės yra tarpusavyje sujungtos briaunomis. Matricos stulpeliai bei eilutės atitinka grafo viršūnes. Pastarųjų indeksas sutampa su stulpelio (atitinkamai eilutės) eilės numeriu. Jei grafą sudaro n viršūnių, tuomet gretimumo matricą atitiks kvadratinė n eilės matrica.

Kiekvienas matricos A elementas žymimas skaičiumi 0 arba skaičiumi 1. Jeigu matricos A elementas $A_{u,v}$ yra lygus 1, tai reiškia, kad viršūnė u yra sujungta briauna su viršūne v (daroma prielaida, kad grafas yra neorentuotas). Jeigu matricos elementas lygus 0, tai reiškia, kad viršūnė u nėra briauna susieta su viršūne v. Jeigu grafas neturi kilpų, tai pagrindinėje diagonalėje visi

elementai yra lygūs 0. Jeigu grafas yra neorentuotas, tai matrica yra simetrinė.

Grafo briaunoms gali būti priskiriami įverčiai. Įverčių reikšmė priklauso nuo konteksto. Tokio tipo grafams gretimumo matricos elementai, kurie žymi sujungtus tarpusavyje mazgus, gali būti prilyginti briaunų įverčiams, o ne pažymėti skaičiumi 1.

1.3. Atstumų matrica

Atstumų matrica – tai matrica, kuria remiantis galima apibrėžti trumpiausius atstumus, skiriančius bet kurias grafo viršūnes. Matricos stulpeliai bei eilutės atitinka grafo viršūnes. Pastarųjų indeksas sutampa su stulpelio (atitinkamai eilutės) eilės numeriu. Jei grafą G sudaro n viršūnių, tuomet atstumų matricą D atitiks kvadratinė n eilės matrica. Matricos D elementas $D_{u,v}$ yra lygus trumpiausiam atstumui nuo viršūnės u iki viršūnės v.

1.4. Tikriniai vektoriai

Kvadratinės matricos A tikriniais vektoriais x vadinami tokie nenuliniai vektoriai, kurie net ir po sudaugos su minėtąja matrica išlieka proporcingi pradiniam dydžiui. Dydis, kuris nurodo, kiek pakito tikrinis vektorius (įvykdžius sandaugą) yra vadinamas tikrine reikšme [San05]:

$$A\,x = \lambda\,x\,, \qquad (1)$$

čia λ – tikrinė reikšmė, x – tikrinis vektorius, A – matrica.

Grafo tikrinę rekšmę atitinka grafo gretimumo matricos (žr. 1.2. poskyrį) tikrinė reikšmė.

1.5. Sąsūkos operacija

Sąsūkos operacija (*angl*. convolution) – tai viena esminių operacijų, naudojamų skaitmeninių vaizdų apdorojime. Jos metu paimama $(2E+1)\times(2S+1)$, $E \in \mathbb{N}$, $S \in \mathbb{N}$, eilės matrica B, kuri yra vadinama branduoliu (*angl*. kernel), bei du $M \times N$, $M \in \mathbb{N}$, $N \in \mathbb{N}$ matmenų skaitmeniniai vaizdai - P, kuriam bus taikoma ši operacija bei kuris atitiks operacijos rezultatą. Prieš panaudojant branduolį tolimesniems skaičiavimams, pirmiausia atliekamas jo elementų atvaizdavimas centrinio elemento atžvilgiu, t.y. $B(m,n) = B(-m,-n)$. Tada nuosekliai imamas kiekvienas pikselis $P'_{m,n} = (m \in [0, M), n \in [0, N))$ bei $(2E+1)\times(2S+1)$ sritis, kuriame jis yra. Sritis išskiriama taip, kad minėtasis pikselis būtų pastarosios centre. Kiekviename žingsnyje atliekama branduolio ir išskirtosios srities atitinkamų elementų daugyba bei gautųjų reikšmių sumavimas. Gautasis skaičius

yra naujoji pikselio $P'_{m,n}=(m\in[0,M),n\in[0,N))$ reikšmė. Šie žingsniai atliekami tol, kol apeinami visi P esantys pikseliai.

Matematiškai sąsūkos operacija užrašoma taip [Jah05]:

$$P'_{m,n}=\sum_{m'=0}^{E-1}\sum_{n'=0}^{S-1}B_{m',n'} \qquad (1)$$

1.6. Filtravimas

Vienas esminių skaitmeninio vaizdo apdorojimo etapų yra tiriamojo objekto identifikavimas bei jo išskirimas iš tam tikros, jį supančios aplinkos. Intuityviai **objektas** yra suprantamas kaip pastovaus spalvingumo sritis, besiskirianti nuo ją supančios kaimyninės srities intensyvumų [Jah05]. Praktikoje toks objekto interpretavimas galimas tik nedaugeliui skaitmeninių vaizdų, kadangi dažniausiai objektų intensyvumai kinta, ir dėl to sunku nuspresti, kuriuos vaizdo elemenentus priskirti objektui, o kurių ne. Taip pat dažnai pasitaiko vaizdo iškraipymų, kuomet įvairiose skaitmeninio vaizdo srityse atsiranda pavieniai pikseliai, savo intensyvumu besiskiriantys nuo juos supančios aplinkos, ir tokiu būdu suformuojantys nepageidaujamą vaizdo defektą [YGV98]. Pastarasis reiškinys dar vadinamas triukšmu.

Triukšmą slopina bei tokiu būdu identifikavimo procesą gerina filtravimas. Šis procesas dažniausiai remiasi 1.1. skyriuje aprašyta sąsūkos operacija. Filtravime naudojamas branduolys yra vadinamas filtru, o jo elementai – svoriais. Svoriai nulemia filtravimo efektyvumą.

1.7. Gauso Filtras

Gauso filtras – tai vienas dažniausiai naudojamų glodinimo filtrų, skirtų triukšmo pašalinimui, skaitmeninių vaizdų detališkumo mažinimui bei skaitmeninių objektų identifikavimo proceso gerinimui. Skaitmeninių vaizdų glodinimas, panaudojant Gauso filtrą, gali būti atliktas dviem būdais:

1. Atliekamas vienmatis filtravimas – horizontalia bei vertikalia kryptimis. Gauso filtras sudaromas panaudojant Gauso pasiskirstymo funkciją [FPW+94]:

$$G(x)=\frac{1}{\sqrt{2\pi\sigma^2}}\mathrm{e}^{-\frac{x^2}{2\sigma^2}}. \qquad (2)$$

2. Atliekamas dvimatis filtravimas. Gauso filtras sudaromas panaudojant Gauso pasiskirstymo funkciją [FPW+94]:

$$G(x)=\frac{1}{2\pi\sigma^2}\mathrm{e}^{-\frac{x^2+y^2}{2\sigma^2}}, \qquad (3)$$

čia σ yra standartinis pasiskirstymo nuokrypis, *x,y* – atstumai nuo koordinačių pradžios atitinkamai O_x ir O_y ašyse.

Filtravimo stiprumas priklauso nuo standartinio pasiskirstymo nuokrypio. Kuo σ didesnis, tuo filtravimas stipresnis. Filtravimo procesas atliekamas remiantis 1.1. poskyryje aprašyta sąsūkos operacija. Gautasis rezultatas – suglodintas skaitmeninis vaizdas (angl. smooth), kuriame yra sumažintas objektų detališkumas bei panaikintas triukšmas.

1.8. SIFT

SIFT (angl. Scale Invariant Feature Transform) – tai lokalių skaitmeninių vaizdų savybių nustatymo, aprašymo bei atpažinimo algorimtas. Remiantis juo, galima identifikuoti skaitmeniniame vaizde esančius objektus bei atlikti jų atpažinimą pateiktojoje skaitmeninių vaizdų aibėje. Esminis algorimto privalumas yra tai, jog lokalios savybės (angl. local features) gerai atpažįstamos net ir įvykdžius skaitmeniame vaizde esančių objektų pasukimo, ištempimo, sutraukimo ar postūmo transformaciją ar apšvietimą [LOW04].

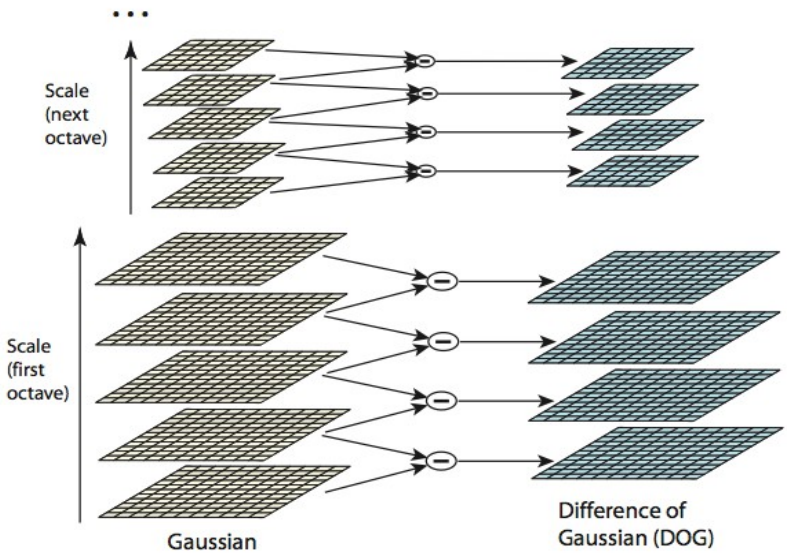
Algoritmą sudaro du esminiai žingsniai:

- Lokalių savybių išskyrimas
  - Ekstremumų radimas
  - Įdomumo taškų nustatymas
  - Orientacijų priskyrimas
  - Lokalių savybių deskriptorių sudarymas (vektorių sudarymas)
- Lokalių savybių atpažinimas kituose skaitmeniniuose vaizduose

Pirmasis žingsnis, kurį reikia atlikti, norint išskirti lokalias skaitmeninių vaizdų savybes, tai skaitmeninių vaizdų piramidžių konstravimas. Pastarasis procesas vykdomas pirmiausiai pasirenkant sutartinį oktavų bei kiekvieną jų sudarančių lygių skaičius (žr. 1.8.1 Pav.). Kiekvieną oktavą sudaro vienodo dydžio skaitmeniniai vaizdai. Kiekvienas iš jų yra filtruojamas Gauso filtru (žr. 1.7 poskyrį). Kiekviename oktavos lygyje dydis σ yra proporcingai didinamas, t.y. $\sigma_i=\sigma\cdot k_i$ (k_i – *i*-

ajame lygyje naudojamas svoris. Pavyzdžiui, pirmajame lygyje $k = 1$, antrajame $k = 2$ ir t.t.). Kadangi σ įtakoja glodinimo stiprumą, tai kiekvienoje oktavoje skaitmeniniai vaizdai skirtinguose lygiuose glodinami vis stipriau. Suglodinus visuose lygiuose esančius vaizdus, paimamas paskutinysis skaitmeninis vaizdas ir sumažinamas du kartus. Gautasis rezultatas naudojamas kitoje oktavoje. Procesas analogiškai kartojamas visose oktavose.

Atlikus glodinimą, skaičiuojamas gretimų skaitmeninių vaizdų skirtumas (angl. difference of Gaussian). Rezultatas – aibė skaitmeninių vaizdų, gautų atimant dviejuose gretimuose lygiuose esančius skaitmeninius vaizdus (žr. 1.8.1 Pav.). Ši rezultatų aibė bus vienetu mažesnė nei yra oktavos lygių. Skaitmeninių vaizdų skirtumas skaičiuojamas visose oktavose.



1.8.1 Pav. Gauso skirtumo skaičiavimas [LOW04]

Trečiasis žingsnis – įdomumo taškų (angl. interest points) nustatymas. Taškai randami tikrinant, ar nagrinėjamas pikselis yra lokalus ekstremumas (šiame žingsnyje naudojami antajame žingsnyje gautieji skaitmeniniai vaizdai). Tikrinimas vykdomas naudojant 8 kaimynus (iš oktavos lygio, kuriame yra nagrinėjamas pikselis) bei 18 pikselių iš gretimuose lygiuose esančių skaitmeninių vaizdų (po 9 iš kiekvieno skaitmeninio vaizdo. Nagrinėjamos srities koordinatės yra analogiškos tikrinamojo pikselio kaiminystės koordinatėms). Jei nustatoma, kad nagrinėjamas pikselis yra mažiausias arba didžiausias iš visų 26 kaimyninių (8 iš nagrinėjamo lygio bei 18 iš gretimų oktavos lygių) tai šis pikselis pažymimas kaip potencialus įdomumo taškas. Šiuo žingsniu užtikrinamas duomenų atpažinimo proceso atsparumas mąstelio keitimui. Atlikus šį žingsnį, įsimenamas oktavos,

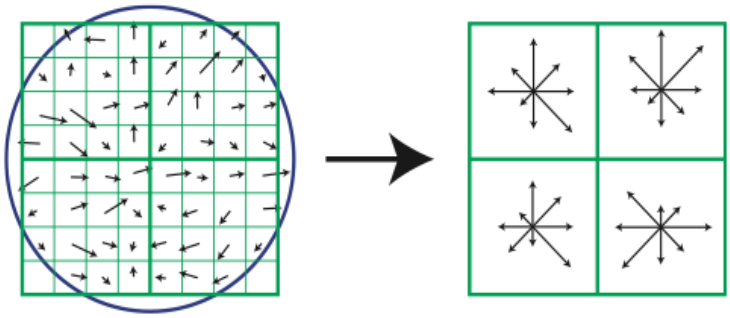
kurioje buvo rastas įdomumo taškas, numeris. Pastarasis dydis yra laikomas lokalios savybės mąsteliu. Jis bus naudojamas lokalių savybių aprašui generuoti. Toliau visi veiksmai vykdomi naudojant skaitmeninį vaizdą iš to oktavos lygio, kuriame buvo rastas minėtasis įdomumo taškas.

Suradus visus potencialius įdomumo taškus, atmetami nepakankamai gerais laikomi taškai. Tai taškai, kurie yra tam tikro, skaitmeniniame vaizde esančio, objekto krašto dalis arba tai yra žemo kontrasto taškai.

Toliau kiekvienam įdomumo taškui priskiriama orientacija. Šiuo žingsniu siekiama užtikrinti, kad lokalių savybių atpažinimas bus efektyvus net ir atlikus posūkio transformaciją. Orientacijos priskyrimo procesas vykdomas apskaičiuojant įdomumo taško kaiminystėje esančių taškų gradiento dydžius bei pastarųjų orientacijas. Remiantis šia informacija, sudaroma orientacijų histograma (histogramą sudaro 36 stulpeliai). Maksimalaus stulpelio indeksas laikomas nagrinėjamo įdomumo taško orientacija. Taip pat atrenkami ir kiti histogramos stulpeliai kurie atitinka 80% dominuojančią gradiento orientaciją. Sukuriami nauji įdomumo taškai, kurių orientacija atitinka šį kriterijų (mastelis ir pozicija yra tokia pati kaip ir nagrinėjamo įdomumo taško).

Galiausiai vykdomas išskirtųjų įdomumo taškų aprašymas. Aprašymui sugeneruoti išskiriama sritis, kurios centras yra įdomumo taškas. Šio algoritmo autorius [LOW04] pataria išskirti 16x16 sritį. Pastaroji suskaldoma į 16 mažesnių 4x4 dydžio sričių. Kiekvienai iš pastarųjų skaičiuojamos orientacijų histogramos. Šiuo atveju histogramą sudaro 8 stulpeliai (nagrinėjamos 8 kryptys). Kiekvieno histogramos stulpelio dydis yra įtakojamas gradiento dydžio. Taigi kiekviena 4x4 sritis aprašoma 8 skaičiais/orientacijomis. Iš viso gaunama 128 skaičiai (16 sričių po 8 dydžius). Gautieji dydžiai papildomai apdorojami, panaudojant, taip vadinamą, Gauso langą (žr. 1.8.2 Pav. pažymėta apskritimu). Pastarosios operacijos metu apskaičuotieji dydžiai dauginami iš tam tikro svorio. Kuo labiau tolstama nuo įdomumo taško, tuo labiau didėja minėtasis svoris. Todėl toliau esančių taškų apskaičiuotieji dydžiai mažėja.

Iš gautųjų 128 dydžių sudaromas vektorius. Pastarasis yra vadinamas lokalios savybės detektoriumi. Jis yra naudojamas analizuojant pateiktąją skaitmeninių vaizdų aibę, siekiant palyginti jos elementus su tu, kurio lokalių savybių deskriptoriai buvo sugeneruoti.



1.8.2 Pav. Lokalių savybių deskriptoriaus kūrimas [LOW04]

1.9. Kamuoliniai medžiai

Kamuolinis medis (angl. ball tree) – tai dvejetainis medis, kuris yra naudojamas metrinių duomenų (duomenų, kuriems galima apibrėžti atstumo operaciją) indeksavimui. Kamuolinių medžių viršūnėse saugomi ne pavieniai taškai, bet taškų aibės [LMG03]. Erdvė skaidoma į hipersferas. Kiekvienoje hipersferoje esančių taškų aibės dydis yra mažesnis nei ją gaubiančiojoje hipersferoje. Kiekvieną medžio viršūnę atitinka viena hipersfera.

Aprašysime kamuolinio medžio struktūrą. Tarkime, kad turime kamuolinį medį T. Tada kiekvienoje viršūnėje $v \in T$ saugomi visi kairiajame bei dešniajame pomedžiuose esantys taškai, t.y. jei $N(v)$ žymi viršūnėje $v \in T$ saugomų taškų aibę, $v.k$ žymi kairįjį vaiką, o $v.d$ – dešinį, tai [LMG+03]:

$$N(v) = N(v.k) \cup N(v.d) . \tag{4}$$

Kamuolinių medžių dešinysis bei kairysis pomedžiai neturi bendrų taškų:

$$\emptyset = N(v.k) \cap N(v.d) . \tag{5}$$

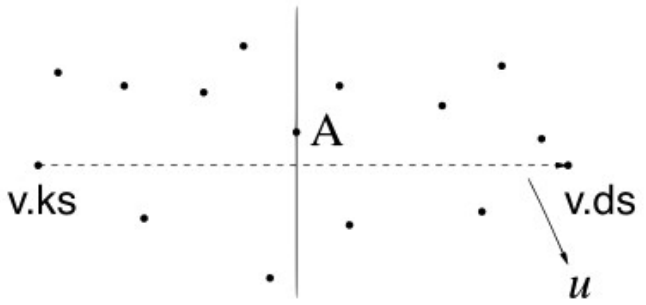
Medžio viršūnėse saugomų elementų aibių skaidymui į sritis atliki, t.y. kriterijaus, kuriuo remiantis erdvės taškai talpinami į kairįjį arba dešinįjį pomedžius, gali būti naudojamos įvairios strategijos. Dažniausiai duomenų skaidymui parenkami du slenksčiai $v.ks$ ir $v.ds$, tokie kad [LMG+03]:

$$|v.ks - v.ds| = max_{p_1, p_2 \in N(v)} |p_1 - p_2| . \tag{6}$$

Tada suprojektuojami visi viršūnės taškai į vektorių $\vec{u}$:

$$\vec{u} = v.\vec{ds} - v.\vec{ks} \,, \tag{7}$$

bei surandamas vidurinysis taškas išilgai jo. Visi suprojektuoti taškai esantys į kairę nuo (7) vektoriaus viduriniojo taško A , talpinami į kairįjį pomedį, t.y. į $v.k$. Visi suprojektuoti taškai esantys į dešinę nuo (7) vektoriaus viduriniojo taško A , talipinami į dešinįjį pomedį, t.y. $v.d$ (žr. 1.5.2 pav.). Galimi ir kitokie slenkstinių taškų parinkimo algorimtai.



1.5.2 Pav. Erdvės taškų skaidymas į poaibius [LMG+03]

Duomenų paieška kamuoliniame medyje vykdoma remiantis paskaičiuotaisiais slenksčiais. Paieškos efektyvumas stipriai priklauso nuo slenksčio parinkimo strategijos. Tarkime, kad duomenų įterpimui buvo naudojamas slenkstis A . Tada vykdant duomenų paiešką, tikrinama, ar paieškos taškas q yra į kairę nuo slenksčio A . Jei paieškos taškas q yra į kairę nuo slenksčio A , tai paiekša toliau vykdoma kairiajame pomedyje. Priešingu atveju – dešiniajame. Kiekviename paieškos žingsnyje taip pat saugoma artimiausio kaimyno reikšmė x . Ši reikšmė žymi artimiausią elementą, aptiktą vienoje iš medžio viršūnų. Atliekant paiešką bei vaikštant po medį, tkrinama ar viršūnėje $v \in T$ saugomi taškai yra potencialiai artimesni už jau surastą artimiausią kaimyną. Tikrinimo sąlyga [LMG+03]:

$$|v.centras - q| - v.d \geq r \,, \tag{8}$$

čia r - atstumas skiriantis artimiausią kaimyną ir paieškos elementą.

Jeigu (8) sąlyga neišpildoma, tai viršūnė $v \in T$ yra praleidžiama, t.y. paieška nevykdoma tos viršūnės pomedžiuose, o tęsiama sekančiose (kuriose paieška dar nebuvo atlikta) viršūnės. Taip vykdant paiešką galima išspręsti artimiausio kaimyno problemą, t.y., sursti elementą, kuris yra arčiausiai paieškos taško.

1.10. KD Medžiai

KD medžiai – tai dvejetainės paieškos medžiai, išreiškiantys rekursyvų geometrinės erdvės skaidymą (k-1) matmenų hiperplokštumomis [JUO07]. Akronimas KD reiškia k-matmenų (angl. k-dimensional). Erdvės skaldymas vykdomas viena iš *k* galimų krypčių (dimensijų). Skaidymas vykdomas tik esant taškų persipildymui erdvėje/poerdvyje. Kiekviename skaldymo žingsnyje kryptis parenkama cikliškai (taip pat galima skaldymo kryptis parinkti panaudojant kitą strategiją; tokiu atveju gautume, taip vadinamuosius, apibendrintus KD medžius). Pavyzdžiui, jei nagrinėjama 3-matė erdvė, tai pirmojo skaldymo metu erdvė dalinama į dalis x koordinatės kryptimi, antrojo skaldymo metu erdvė yra skaldoma y kryptimi, o trečiojo – z kryptimi. Jei įterpus naują elementą į medį reikia dar kartą skaldyti erdvę, tai skaldymas vykdomas vėl naudojant x dimensiją, t.y. vykdomas antras ciklas. Analogiškai procesas tęsiamas įterpiant kitus taškus.

Duomenų įterpimas vykdomas panašiai kaip ir dvejetainiame paieškos medyje. Pirmiausia surandama vieta, kur turėtų būti talpinamas taškas, o tada medis papildomas minėtuoju tašku. Tam, kad būtų galima nustatyti, kuria kryptimi tęsti paiešką, kiekvienoje viršūnėje saugomas ne tik taškas, bet ir specialus įrašas (vadinamas diskriminantine reikšme), kuris nurodo nagrinėjamame lygyje naudotą skaldymo kryptį. Remiantis pastaruoju įrašu, lyginama įterpiamojo/ieškomojo taško koordinatė su saugojamo viršūnėje taško koordinate (kuri koordinatė lyginama yra nustatoma pagal diskriminantinę reikšmę).

Kiekvienas duomenų taškas asocijuojamas su kurios nors viršūnės tašku KD medyje [JUO07]. Kiekvienai sričiai gali priklausyti daugiausiai vienas taškas. Jei vykdant įterpimo operaciją sritis persipildo, t.y. taškų skaičius srityje yra didesnis už vienetą, tai vykdomas srities skaldymas. Todėl medžio struktūra priklauso nuo taškų įterpimo tvarkos.

2. Centriškumas

Grafų teorijoje naudojami įvairūs metodai, kurių pagalba galima įvertinti tam tikros grafo viršūnės svarbą kitų viršūnių atžvilgiu. Šis įvertis – tai sąlyginis dydis, priklausantis nuo konteksto,

kuriame yra nagrinėjamas probleminis klausimas. Jis apskaičiuojamas kiekvienai grafo viršūnei pritaikant tam tikrą taisyklę (priklauso nuo konkretaus metodo). Grafo viršūnės svarbai nusakyti naudojama viršūnės (mazgo) centriškumo (angl. Centrality) sąvoka.

Šio tiriamojo darbo kontekste, grafo viršūnės svarba yra naudojama nustatyti, kiek tam tikras skaitmeninis vaizdas atitinka paieškos kriterijų (daroma prielaida, kad grafas yra naudojamas skaitmeninių vaizdų sąryšiams apibrėžti). Pavyzdžiui, paieškos sistemoje pateikiama užklausa ir gaunama rezultatų aibė. Ši abė subjektui turi būti pateikiama taip, kad pirmiausia būtų rodomas rezultatas, kuris labiausiai tenkina užklausą, toliau rodomas mažiau tinkamas rezultatas ir t.t. Aibės elementai analizuojami bei remiantis pastarųjų turiniu prioretizuojami. Prioretizavimas vykdomas nustatant kiekvieno aibės elemento svarbą. Jeigu duomenų analizei naudojamas grafas, o pastarojo mazgas atitiks skaitmeninį vaizdą, tai skaitmeninio vaizdo svarbos skaičiavimas atitiks mazgo svarbos įverčio apskaičiavimą.

Keletas iš plačiai naudojamų grafo viršūnės svarbumo įverčio skaičiavimo metodų: kampo (angl. degree centrality), tarpusavio būvio (angl. Betweenness centrality), artimumo (angl. Closeness-like centrality) bei tikrinio vektoriaus (dar vadinmo charakteringuoju vektoriumi) centriškumo metodai.

2.1. Tikrinio vektoriaus centriškumas

Tikrinio vektoriaus centriškumas (angl. eigenvector centrality) – tai metodas, kuriuo naudojantis galima apskaičiuoti grafo viršūnės svarbą kitų viršūnių atžvilgiu. Naudojantis šiuo metodu kiekvienai viršūnei priskiriamas įvertis. Įvertis apskaičiuojamas remiantis idėja, jog sąryšiai su aukštesnį įvertį turinčiomis viršūnėmis daro didesnę įtaką nagrinėjamos viršūnės įverčiui nei sąryšiai su mažesnį įvertį turinčiomis viršūnėmis. Todėl viršūnės įvertis yra didesnis kuo pastaroji yra arčiau „svarbios“ viršūnės. Skaitmeninių vaizdų prioretizavime tai atitiktų rezultatų aibės elementų, labiausiai atitinkančių užklausą, „dėjimą“ į rezultatų aibės pradžią.

Matematiškai aprašysime tikrinių vektorių centriškumo metodą. Tarkime yra duota gretimumo matrica A bei ją atitinkantis grafas G. Grafo G viršūnės i įvertis bus proporcingas sumai viršūnių, kurios yra sujungto su viršūne i, įverčių:

$$x_i = \frac{1}{\lambda} \sum_{j=1}^{N} A_{ij} x_j , \qquad (9)$$

čia N yra visų grafo viršūnių keikis, λ – konstanta, A – gretimumo matrica

Ši išraiška gali būti perrašyta matricine forma. Ji atitiks matricos tikrinio vektoriaus išraišką:

$$Ax = \lambda x, \tag{10}$$

i-oji vektoriaus *x* reikšmė atitiks viršūnės *i* svarbos įvertį.

Konstanta λ bei tikrinis vektorius gali būti iteraciškai apskaičiuojami remiantis taisykle [Cho05]:

$$W(t+1) = Aw(t), \tag{11}$$

$$w(t+1) = \frac{(W(t+1))}{\|W(t+1)\|}. \tag{12}$$

Taisyklė pradedama naudojant pradinį nenulinį vektorių w(0).

Apibendrinus (9), (10), (11) ir (12) formules gauname skaitmeninio vaizdo įverčio skaičiavimo formulę:

$$I = A I, \tag{13}$$

čia *I* atitinka (12) formulėje apskaičiuotą matricą, *A* – normalizuota pagal stulpelį gretimumo matrica.

Vektorius *I* skaičiuojamas iteraciškai taikant (13) formulę. Kiekvienoje iteracijoje įvertis konverguoja į matricos tikrinį vektorių, kurio pagalba randami skaitmeninių vaizdų įverčiai. Iteratyviai taikant taisyklę, dydis konverguos tik tada, jei matrica *S* yra neredukuojama. Tam, kad būtų užtikrintas šis reikalavimas papildomai naudojamas parametras *d* [JB08]*:*

$$I = d\, A\, I + (1-d)\, p, \tag{14}$$

kai

$$p = [\frac{1}{n}], \tag{15}$$

čia *n* – nagrinėjamoje aibėje esančių skaitmeninių vaizdų skaičius, *p* – matrica stulpelis, *A* – normalizuota pagal stulpelį gretimumo matrica.

2.2. Kampo centriškumas

Tai bene paprasčiausias centriškumo skaičiavimo metodas. Šiuo atveju grafo viršūnes centriškumas yra lygus viršūnių, su kuriomis yra sujunga nagrinėjama viršūnė, skaičiui, t.y. incidenčių briaunų skaičiui.

Matematiškai aprašysime kampo centriškumo (angl. degree centrality) metodą. Tarkime duota grafo *G* gretimumo matrica *A*. Konstruojant gretimumo matricą, nekreipiamas dėmesys į briaunų įverčius. Matricos elementai lygūs 1, jei viršūnės jungiamos briauna, bei 0, jei viršūnės nėra sujungtos briauna. Tada kampo centriškumas apskaičiuojamas remiantis formule [BE06]:

$$c_i = \sum_i A_{ij}, \tag{16}$$

(16) formule gali būti išreikšta matriciniu pavidalu:

$$c = A\,I, \tag{17}$$

čia I – matrica stulpelis, kurios visi elementai lygūs 1.

Taigi pastebima, jog kampo centriškumo skaičiavimas, tai vienetinių kelių, prasidedančių nagrinėjamoje grafo viršūnėje, skaičiaus radimas. Šis metodas, tai atskiras *k*-ilgio centriškumo, kai *k* = 1, atvejis.

2.3. Artimumo centriškumas

Artimumo centriškumo metodas (angl. Closeness-like centrality) yra labai panašus į kampo centriškumo metodą (žr. 2.2 skyrių). Šis metodas yra paremtas trumpiausių kelių, skiriančių grafo viršūnes, skaičiavimu.

Matematiškai aprašysime artimumo centriškumo metodą. Tarkime yra duotas grafas *G* bei atstumu matrica *D*. Tada grafo viršūnės centriškumas apskaičiuojamas remiantis taisykle [BE06]:

$$c_i = \sum_j D_{ij}, \tag{18}$$

(18) formule gali būti išreikšta matriciniu pavidalu:

$$c = D\,I\,, \tag{19}$$

čia I – matrica stulpelis, kurios visi elementai lygūs 1.

2.4. Tarpusavio būvio centriškumas

Tarpusavio būvio (angl. Betweenness centrality) centriškumo skaičiavimas vykdomas įvertinus trumpiausių kelių, skiriančių grafo viršūnes, skaičių. Grafo *G* viršūnės *k* centriškumas randamas nustatant, kiek trumpiausių kelių skiriančių grafo viršūnes *u* ir *v* eina per viršūnę *k*.

Matematiškai aprašysime tarpusavio būvio centriškumo metodą. Jei g_{ij} žymi skaičių trumpiausių kelių tarp grafo viršūnių *i* ir *j*, o g_{ikj} žymi skaičių trumpiausių kelių tarp grafo viršūnių *i* ir *j* einančių per viršūnę *k*, tai viršūnės *k* tarpusavio būvio centriškumas yra [BE06]:

$$c_k = \sum_i \sum_j \left(\frac{g_{ikj}}{g_{ij}}\right). \tag{20}$$

Jeigu egzistuoja tik vienas trumpiausias kelias tarp bet kurių grafo viršūnių, tai centriškumas bus lygus trumpiausių kelių, einančių per viršūnę *k,* skaičiui. Taip pat egzistuoja keletas šio metodo skaičiavimo variacijų. Kartais vietoje trumpiausių kelių nagrinėjimo, skaičiuojama betkokio ilgio keliai, kurie jungia grafo viršūnes *u* ir *v* bei eina per viršūnę *k*.

3. Susiję darbai

Duomenų paieška yra plačiai nagrinėjama tema. Daugelyje pirmųjų saityno paieškos sistemų paieška buvo vykdoma paprasčiausiai ieškant sistemos naudotojo pateiktųjų raktažodžių. *Lary page* ir *Sergey Brin* pastebėjo, kad tokio tipo paieška yra neefektyvi. Rezultatų aibėje pateikiama per daug mažai su pateiktąja užklausa susijusių rezultatų. Kadangi saitynas yra plačiai naudojamas kaip terpė vykdyti komercinę veiklą, tai dažnai pasitaikydavo atvejų, kai saityno puslapių autoriai stengėsi pasinaudoti sistemos trūkumais bei dirbtinai padidinti puslapio svarbą. Sistemą buvo galima nesunkiai suklaidinti įterpiant į puslapio aprašus su puslapio turinio nesusijusių, tačiau populiarių raktažodžių. Tokiu būdu į rezultatų aibę patekdavo visiškai paieškos neatitinkančių rezultatų.

Lary page ir *Sergey Brin* pasiūlė panaudoti saityno puslapiuose esančias nuorodas bei su jomis susijusią informaciją [BP98]. Jų tikslas buvo išnaudoti šią informaciją taip, kad sistemos naudotojui būtų pateikti prioretizuoti paieškos rezultatai. Šis siekis buvo grindžiamas tuo, kad saityne esančių

duomenų kiekis nepaliaujamai sparčiai didėja. Tačiau žmogaus sugebėjimas peržvelgti visą susijusią informaciją yra ribotas. Sistemos naudotojai dažniausiai nori peržvelgti labiausiai tenkinančius užklausą rezultatus. Apsiribojama pakankamai nedideliu skaičiumi rezultatų, pavyzdžiui, pirmaisiais dešimčia.

Lary page ir *Sergey Brin* pateikė stambaus saityno paieškos variklio prototipą. Pastarąjį pavadino Google (šiuo metu tai dominuojanti rinkoje saityno paieškos sistema). Šios sistemos pagrindinis tikslas – rezultatų tikslumas. Tam, kad būtų galima nustatyti rezultatus, kurie yra aktualiausi sistemos naudotojui, jie pasiūlė įvertinti kiekvieną saityno puslapį. Šį įvertį pavadino PageRank [BPM+98][BCH+06]. Įvertis randamas nagrinėjant saitų struktūrą. Šis įvertis naudojamas prioretizuojant saityno puslapius. Autoriai palygino pasiūlytąjį paieškos metodą su akademinėje benduomenėje paplitusiu mokslinių straipsnių svarbos vertinimu [BP98]. Kuo daugiau straipsnis yra cituojamas kituose moksliniuose straipsniuose, tuo jis yra laikomas svarbesnis. Taip pat veikia ir pasiūlytasis algoritmas. Kuo daugiau yra nuorodų į puslapį (šios nuorodos yra vadinamos balsais), tuo jis yra laikomas svarbesniu. Tiesa, šiuo atveju ne visos nuorodos yra vertinamos vienodai. Jeigu nuoroda yra įdėta į svarbų puslapį, tai saityno puslapis, į kurį rodo ši hipertekstinė nuoroda, yra laikomas taip pat svarbesniu. Kitaip tariant, vienų puslapių įvertis daro įtaką kitų puslapių įverčio skaičiavimui.

Įverčių skaičiavimui galima panaudoti sąsajų matricą. Tai yra nuorodų gretimumo matrica (dar vadinama Google matrica). Matricos elementai žymi, ar saityno puslapiuose yra nuorodos į kitus puslapius. Šios matricos vyraujantis tikrinis vektorius atitinka saityno puslapių įverčius (PageRank).

Lary page ir *Sergey Brin* tegė, kad pasiūlytojo algoritmo apskaičiuojama puslapių svarba atitinka tai, ką sistemos naudotojai subjektyviai laiko svarbiu turiniu.

Henry Rowly ir *Shumeet Baluja* kaip ir *Lary page* bei *Sergey Brin* taip pat tyrinėjo nekorektiškų saityno paieškos rezultatų problemą. Tiesa, skirtingai nei pastarieji, jie domėjosi skaitmeninių vaizdų paieška [JB08]. *Henry Rowly* ir *Shumeet Baluja* atkreipė dėmesį, kad šiuolaikinės paieškos sistemose paieška atliekama remiantis arba vien tekstine informacija, arba panaudojant tik globaliąsias skaitmeninių vaizdų savybes (pavyzdiui, analizuojant histogramas). Tokia paieška dažniausiai yra nekorektiška, kadangi rezultatų aibėje gali atsidurti su paieškos užklausa nesusijusių skaitmeninių vaizdų. Semantiškai nesusiję vaizdai pateikiami chaotiškai. Jie pasiūlė atlikti skaitmeninių vaizdų paiešką panaudojant algoritmą, kurio veikimo principas yra panašus į *Lary page* ir *Sergey Brin* pasiūlytą puslapių įverčio skaičiavimo algoritmą. Esminis skirtumas yra tai, kad

vietoje tekstinių sąsajų analizuojamos vaizdinės sąsajos.

Henry Rowly ir *Shumeet Baluja* pasiūlė atlikti mišrią paiešką. Pirmiausiai išrenkami skaitmeniniai vaizdai atliekant teksto analize paremtą paiešką, o vėliau atliekama skaitmeninių vaizdų analizė. Išanalizavus skaitmeninių vaizdų turinį, galima nustatyti, kiek jie tarpusavyje yra panašūs. Remiantis šia informacija galima skaitmeninius vaizdus prioretizuoti bei pateikti rezultatus sistemos naudotojui. Šis įvertis yra vadinamas ImageRank arba skaitmeninių vaizdų įverčiu.

Nuosekliai tęsdami savo darbą *Henry Rowly, Shumeet Baluja* ir *Henry Rowley* [JBR07] taip pat tyrinėjo kitą susijusią problemą. Jie norėjo sukurti algoritmą, kuris pagal pateiktąją užklausą sugebėtų grąžinti vieną, patį tinkamiausią skaitmeninį vaizdą. Duomenų indeksavimui pasiūlė naudoti kamuolinius medžius.

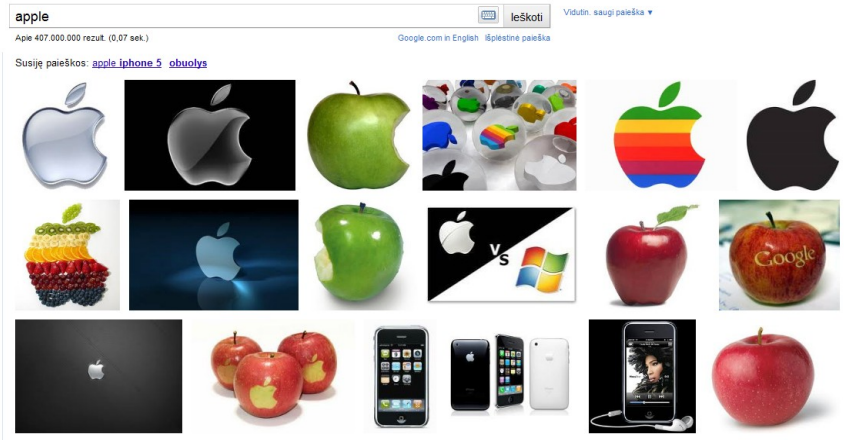
David G. Lowe savo darbuose tyrinėjo bei aprašė algoritmą [LOW04][LOW99], kuris leistų išskirti bei su didele tikimybe aptikti objektus, esančius keliuose skirtinguose skaitmeniniuose vaizduose. Sukurtasis algoritmas buvo pavadintas SIFT (angl. Scale Invariant Feature Transform). SIFT - tai lokalių skaitmeninių vaizdų savybių nustatymo, aprašymo bei atpažinimo algorimtas. Remiantis juo galima identifikuoti skaitmeniniame vaizde esančius objektus bei atlikti jų atpažinimą pateiktojoje skaitmeninių vaizdų aibėje. Esminis algorimto privalumas yra tai, jog lokalios savybės (angl. local features) gerai atpažįstamos net ir įvykdžius skaitmeniame vaizde esančių objektų pasukimo, ištempimo, sutraukimo ar postūmo transformaciją.

Neeraj Kumar, *Li Zhang*, and *Shree Nayar* [KZN08] tyrinėjo skaitmeninių vaizdų panašumo įvertinimą. Daugiausiai dėmesio skyrė algoritmų našumui įvertinti. Atlikto palyginamąją įvairių daugiamačių duomenų struktūrų analizę. Tyrimų metu nustatė, kad greičiausiai artimiausio kaimyno paieška atliekama kamuoliniuose medžiuose.

4. Skaitmeninių vaizdų įvertis

Paieškos sitemose atliekant skaitmeninių vaizdų paiešką, užklausos yra vykdomos analizuojant tekstinę informaciją. Dažniausiai nekreipiamas dėmesys į skaitmeninių vaizdų turinį. Vykdant teksto analize paremtas užklausas dažniausiai apimama tik nedidelė dalis skaitmeninio vaizdo informacijos. Dėl šios priežasties gautieji rezultatai ar pastarųjų pateikimo tvarka ne visada atitinka išankstinius lūkesčius. Pavyzdžiui, įvedus į saityno paieškos lauką raktažodį „apple“ (liet. obuolys), tikimasi rasti skaitmeninius vaizdus, kuriuose yra pavaizduotas minėtasis vaisius. Vienuose skaitmeniniuose vaizduose šis elementas bus dominuojantis objektas, kituose tik nežymi dedamoji

dalis. Tačiau bet kuriuo atveju šis elementas bus pastebimas pirmuosiuose rezultatų aibės elementuose. Įvedus šį raktažodį į „Google“ paieškos langą, gaunami kiek netikėti rezultatai (žr. 4.1 pav). Kaip matoma, rezultatų aibėje yra chaotiškai pateikiami obuolius simbolizuojantys skaitmeniniai vaizdai kartu su skaitmeniniais vaizdais, kuriuose pavaizduoti mobilieji įrenginiai. Tai nėra gerai, subjektui aktuali informacija turi būti pateikiama nuosekliai nuo aktualiausios iki mažiausiai aktualios (kitaip tariant, nuo labiausiai iki mažiausiai užklausą tenkinančios informacijos).



4.1 pav. Rezultatų aibės fragmentas gautas įvedus raktinį žodį „apple“ (liet. obuolys)

Taigi kodėl taip atsitiko? Kaip ir minėta, standartinės paieškos sistemos analizuoja tekstinę informaciją. Žodis „apple“ reiškia ne tik vaisiaus tipą bet ir kompanijos pavadinimą. Jei saityno puslapyje yra dažnai minimas šis žodis, t.y. šiuo atveju minimi mobilieji įrenginiai, kurie yra pagaminti kompanijos „apple“, bei į šį puslapį rodo (naudojant hypertekstines nuorodas) pakankamai daug kitų puslapių (ši informacija yra naudojama skaičiuojant saityno puslapio svarbą), tai visi skaitmeniniai vaizdai, esantys minėtajame puslapyje, laikomi užklausą tenkinčiais skaitmeniniais vaizdais. Daroma prielaida, kad puslapyje dedami tik tie skaitmeniniai vaizdai, kurie yra susiję su puslapio turiniu. Rezultatai pateikiami pagal puslapių įvertinimą (PageRank). Analogiška problema egzistuoja ir asmeniniuose kompiuteriuose įdiegtoje paieškos funkcijoje. Šiuo atveju analizuojama tekstinė informacija, kuri yra susieta su bylomis bei katalogais. Rezultatų eiliškumas priklauso nuo apdorojamų bylų eiliškumo.

Skaitmeninių vaizdų paieška, panaudojant lyginamąją analizę, leidžia įvertinti kiekvieną skaitmeninį vaizdą (analizuojant skaitmenius objektus, o ne vien tekstinę informaciją) bei

interpretuoti, kiek jis subjektui yra aktualus. Pasinaudojus šiomis galimybėmis, galima pateikti prioretizuotus rezultatus. Rezultatai pateikiami nuosekliai išdėstant skaitmeninius vaizdus nuo labiausiai iki mažiausiai užklausą tenkinančių skaitmeninių vaizdų.

Atsižvelgus į minėtosios duomenų paieškos rezultatų netikslumus buvo pasiūlytas patobulintas skaitmeninių vaizdų paieškos bei rezultatų pateikimo metodas. Jis remiasi skaitmeninių vaizdų vertinimu [JBR07][JB08]. Šis įvertis yra vadinamas skaitmenini vaizdų įverčiu (angl. Image rank). Panaudojus skaitmeninių vaizdų įvertį, galima prioretizuoti skaitmeninius vaizdus.

Pagrindinis skaitmeninių vaizdų įverčio skaičiavimo tikslas – surasti bendrais bruožais (turinio prasme) pasižyminčius skaitmeninius vaizdus bei nustatyti, kurie iš jų subjektui yra aktualiausi. Bendri bruožai nėra išskiriami žmogui suprantama forma, t.y. neieškoma objektų, kurie žmogui yra intuityviai suprantami. Pavyzdžiui nesiekiama rasti skaitmeninių vaizdų, kuriuose yra pavaizduota gėlė, paukštis, obuolys ar kitas objektas. Žmogus sugeba abstrahuoti vaizdą, o kompiuteris – ne. Minėtieji objektai žmogui yra lengvai pastebimi (rega), o kompiuteriui tai pakankamai sudėtingas uždavinys. Kiekvienas objektas turi būti griežtai matematiškai apibrėžtas arba pateikiamas algoritmas kaip, vykdant komandų seką, atpažinti norimą objektą. Labai sudėtinga „išmokyti“ kompiuterį suprasti, ką reiškia obuolys, gėlė ar bet kuris kitas realaus pasaulio objektas.

Analizuojant skaitmeninių vaizdų aibę, iš anksto nėra žinoma, kas yra bendra. Bendri bruožai yra nustatomi formuojant charakteringuosius vektorius (deskriptorius), kuriuos sudaro objektai, aptinkami skaitmeniniuose vaizduose. Bendri bruožai tai skaitmeninių vaizdų fragmentai, kurie duominuoja rezultatų aibėje (nagrinėjant atsižvelgiama į tai, jog objektai gali būti transformuoti, t.y. ištempti, sutraukti, pasukti ir pan.). Gavus objektų aibę, nustatoma, kokie skaitmeniniai objektai dažniausiai aptinkami nagrinėjamuose vaizduose. Vienur šie objektai yra esminis, kituose mažiau matomas elementas. Šio elemento (-ų) dominavimas (visoje rezulatų aibėje) leidžia daryti prielaidą, jog tai užklausą tenkinantis elementas (-ai). Todėl subjektui įdomiausi (svarbiausi) skaitmeniniai vaizdai bus tie, kuriuose minėtieji objektai labiausiai pasireiškia.

Suformavus deskriptorius, konstruojamas panašumo grafas. Kiekviena grafo viršūnė atitinka skaitmeninį vaizdą, o briauna sąryšį tarp vaizdų, kuris išreiškiamas priskiriant briaunai skaliarą, t.y. įvertį. Skaitmeniniai vaizdai jungiami tarpusavyje remiantis jų panašumu. Kuo vaizdai turės daugiau bendrų bruožų, tuo jie bus laikomi panašesni. Atitinkamai briaunos, jungiančios šiuos vaizdus, įvertis bus didesnis. Turint panašumo grafą bei pasinaudojus grafo viršūnes centriškumo skaičiavimo algoritmais, kiekvienai viršūnei apskaičiuojamas įvertis (paprastai tariant, priskiriamas

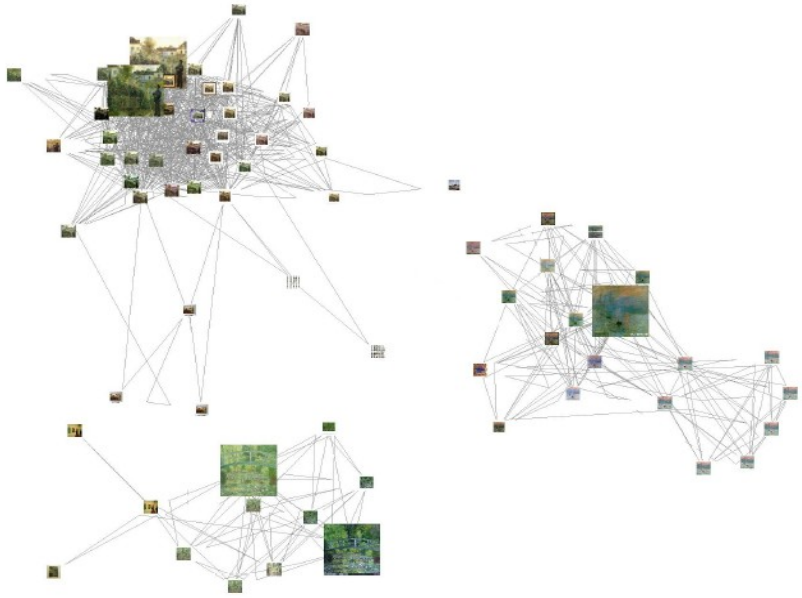
skaičius). Šis dydis atitinka skaitmeninio vaizdo svarbą rezultatų aibės elementų kontekste. Galiausiai subjektui pateikiami rezultatai – pirmiausia pateikiamas skaitmeninis vaizdas, kuio įvertis yra didžiausias, tada seka skaitmeninis vaizdas su mažesniu įverčiu ir t.t.

4.1. Skaitmeninio vaizdo įverčio skaičiavimo procedūra

Skaitmeninio vaizdo įverčio skaičiavimo procedūra:

1. Pirmasis žingsnis, kurį reikia atlikti, tai surasti pradinę skaitmeninių vaizdų aibę, kurioje ieškosime panašiausių vaizdų. Panašiai kaip ir saityne, pradinė aibė randama analizuojant meta duomenis – bylų bei katalogų pavadinimus, savybes, parametrus ir pan. Šio žingsnio rezultatas – n skaitmeninių vaizdų. Reikia atkreipti dėmesį, kad darbo metu nagrinėjamas algoritmas yra skirtas paieškai, kuri vykdoma naudojant raktinį žodį, t.y., siekiama, kad gautieji rezultatai semantiškai būtų susiję su paieškos kriterijumi. Piešingu atveju (pavyzdžiui, jei paieškai naudojamas kitas skaitmeninis vaizdas) šis žingsnis gali būti praleistas arba supaprastintas, pavyzdžiui, išreikštiniu būdu pateikiant šakninį katalogą (jame rekursyviai išrenkame skaitmeninius vaizdus bei atliekame lyginamąją analizę).
2. Gavus skaitmeninių vaizdų aibę, reikia išskirti bruožus, kuriais pasižymi kiekvienas aibės elementas, bei suformuoti deskriptorius. Deskriptorius – tai skaičių vektorius, apibūdinantis skaitmeninius objektus, aptikus nagrinėjamame vaizde (šie objektai dar vadinami savybinėmis sritimis). Deskriptoriai formuojami kiekvienam aibės vaizdui atskirai, nustatant įdomumo taškus bei formuojant savybines sritis. Deskriptorių formavimui naudojamas SIFT algoritmas (žr. 1.8 poskyrį).
3. Trečiasis žingsnis – tai panašumo grafo konstravimas (žr. 6 skyrių). Ši duomenų struktūra yra naudojama skaitmeninių vaizdų įverčiui rasti. Kiekvienas mazgas atitinka skaitmeninį vaizdą, o briaunos sąryšius tarp jų. Kiekvienai briaunai priskiriamas skaliaras, kuris nusako jungiamųjų mazgų sąryšio stiprumą. Įverčiai priskiriami analizuojant bendruosius bruožus, kuriais pasižymi nagrinėjami skaitmeniniai vaizdai. Bendrų bruožų paieška vykdoma analizuojant bruožų deskriptorius (žr. 2 žingsnį). Tarkime duoti du deskriptoriai D_u ir D_v, apibrėžiantys savybes, kurios buvo aptiktos skaitmeniniuose vaizduose u ir v. Tada panašumą tarp šių dviejų skaitmeninių vaizdų atitiks skaičius bruožų, kurie aptinkami abiejuose skaitmeniniuose vaizduose. Jei konstruojant panašumo grafą aptinkama skaitmeninių vaizdų, kurie nėra susiję su nei vienu kitu vaizdu, tai tokie vaizdai yra laikomi „išsišokeliais“ bei gali

būti pašalinami iš nagrinėjamos aibės (daroma prielaida, kad jie subjektui yra neaktualūs). Šiame darbe panašių deskriptorių paieškai atlikti naudojami KD medžiai (alternatyva kamuoliniams medžiams).



4.1 pav. Panašumo grafai [JB08]

4. Paskutinysis žingsnis – tai skaitmeninių vaizdų įvertinimas bei pateikimas subjektui. Įvertis randamas naudojant tikrinio vektoriaus centriškumo metodą. Skaitmeniniai vaizdai pateikiami taip: pirmiausia pateikiamas vaizas, kuris labiausiai tenkina užklausą, t.y. grafo viršūnė turi didžiausią įvertį, tada mažiau tinkamas vaizas ir t.t.

4.2. Paieškos paremtos skaitmeninio vaizdo įverčio skaičiavimu trūkumai

Vykdant pateiktąjį įverčio skaičiavimo metodą neišvengiamai susiduriama ir su esminiais trūkumais. Vienas iš labiausiai pastebimų – tai didelis kompiuterio resursų poreikis. Skaičiuojant skaitmeninių vaizdų įverčius, atliekama daug bei sudėtingų skaitmeninių vaizdų apdorojimo operacijų. Šios operacijos yra „brangios“ resursų atžvilgiu. Tuo labiau, kad apdorojus skaitmeninių vaizdų turinį yra konstruojami panašumo grafai, naudojamos gretimumo matricos. Yra pakankamai didelė tikimybė, kad minėtoji matrica bus išretinta (angl. sparse matrix). Tai yra neefektyvu. Taikant šį metodą saityne esantiems vaizdams prireiktų begalo galingų kompiuterių tam, kad būtų galima vygdyti lyginamąją analizę esančiam milžiniškam skaitmeninių vaizdų kiekiui. Praktiškai toks

taikymas yra (beveik) neįmanomas. Todėl yra siūloma alternatyva, kuri yra paremta mišriu globalių bei lokalių skaitmeninių vaizdų savybių naudojimu bei gautųjų rezultatų taikymu vertinant kiekvieną iš jų. Šis mišrus būdas grindžiamas idėja, jog asmuo (organizacija) į savo puslapį dės skaitmeninius vaizdus, kurie yra susiję su puslapio turiniu, o ne talpins atsitiktinius nuo bendrojo konteksto nutolusius vaizdus. Tokiu atveju, ištyrus tam tikrą aibę puslapių (teksto analize paremtų užklausų vykdymas), liktų atlikti kokybinį filtravimą (skaitmeninių vaizdų analize paremtų užklausų vykdymas), t.y. įverčio skaičiavimą bei remiantis gautaisiais rezultatais skaitmeninių vaizdų pateikimą subjektui. Visgi pastaruoju atveju nėra aišku, kokia yra riba, kai vis dar galima taikyti šį metodą. Kaip ir minėta ankstensiuose skyriuose, bendri bruožai – tai skaitmeninių vaizdų fragmentai, kurie duominuoja rezultatų aibėje. Tai reiškia, kad esant dideliems duomenų kiekiams, t.y. skaitmeninių vaizdų skaičius didelis, bei esant pakankamai dideliam kiekiui neteisingų vaizdų tarp jų, t.y. nesusijusių su tekstine užklausa, yra tikimybė, kad bus išskirti neteisingi bruožai. Tokiu atveju pastaroji situacija atitiks taip vadinamą triukšmą, kuris sąlygos klaidingas interpretacijas bei pateikiamus rezultatus subjektui.

5. Skaitmeninio vaizdo įverčio skaičiavimas

Šiame darbe įverčio paieška vykdoma naudojant panašumo grafą bei taikant jame tikrinio vektoriaus centriškumo metodą (žr. 2.1. poskyrį). Toliau bus apžvelgiami rezultatai, kurie buvo gauti pritaikius darbo metu sukurtąją taikomąją programą.

Suirnkus testinius duomenis bei suformavus panašumo grafą, dažnai susiformuoja pavienių skaitmeninių vaizdų „salelės“, t.y. grafe yra mazgų, kurie neturi incidenčių briaunų (žr. 5.1 pav). Tai reiškia, kad tiriamojoje aibėje yra skaitmeninių vaizdų, kurie nėra panašūs (turinio prasme) į bet kurį kitą aibėje esantį skaitmeninį vaizdą. Tokie skaitmeniniai vaizdai yra laikomi „išsišokeliais”. Kadangi jie yra izoliuoti elementai bei neturi sąryšio su nei vienu kitu elementu, tai prioretizuojant skaitmeninius vaizdus juos galima dėti į rezultų aibės pabaigą arba tiesiog pašalinti iš rezultatų aibės bei neįtraukti į tolimesnius skaičiavimus. Daroma prielaida, kad šie vaizdai subjektui yra neaktualūs.

Taikomojoje programoje įverčių skaičiavimui buvo taikoma (14) formulė (žr. 2.1. poskyrį). Panaudojus ją, gauname, kad visų „išsišokelių“ įverčiai yra vienodi. Taip yra dėl to, kad izoliuotos grafo viršūnės neturi incidenčių briaunų. Todėl gretimumo matricos atitinkamas stulpelis yra sudarytas iš nulinių elementų. Atlikus matricų sandaugą, rezultatas taip pat lygus nuliniam vektoriui. Todėl tikrinis vektorius yra lygus nuliui. Taigi gauname, kad „išsišokelių“ įverčiai yra lygūs

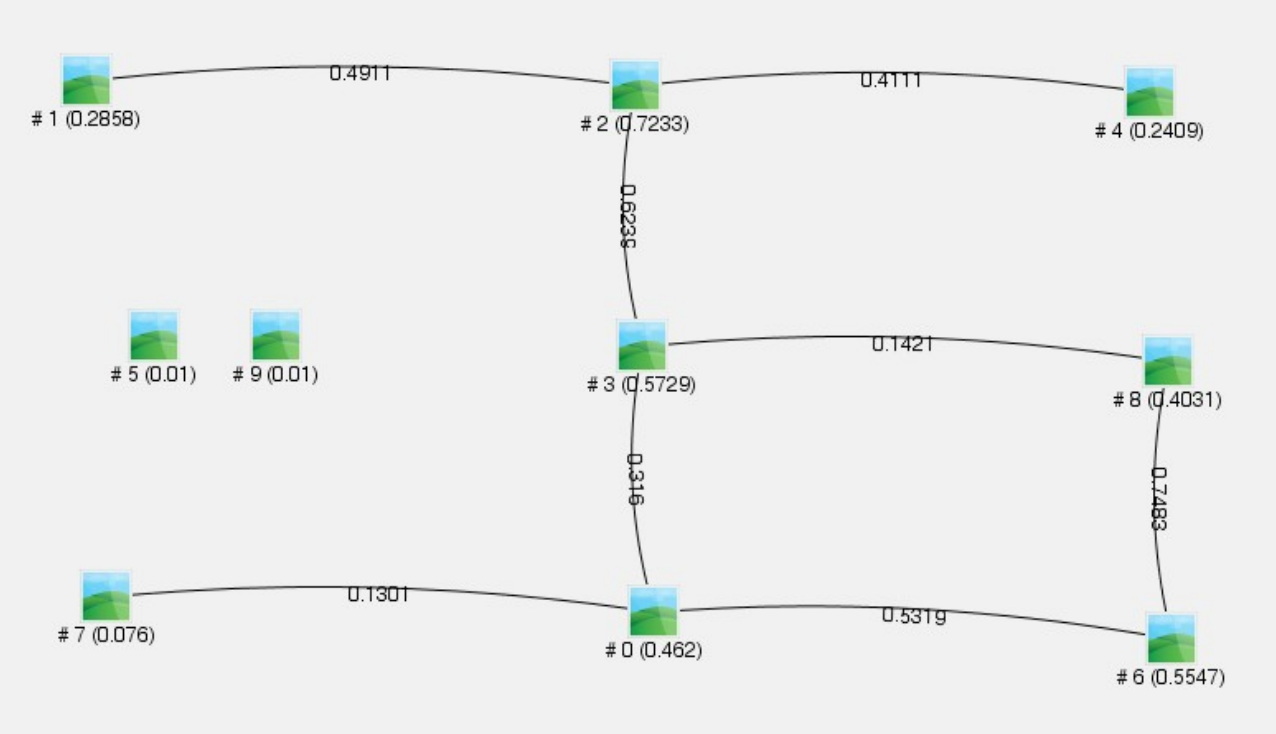
konstantai (nenormalizavus):

$$(1-d)\,p\,, \tag{21}$$

kai

$$p=[\frac{1}{n}], \tag{22}$$

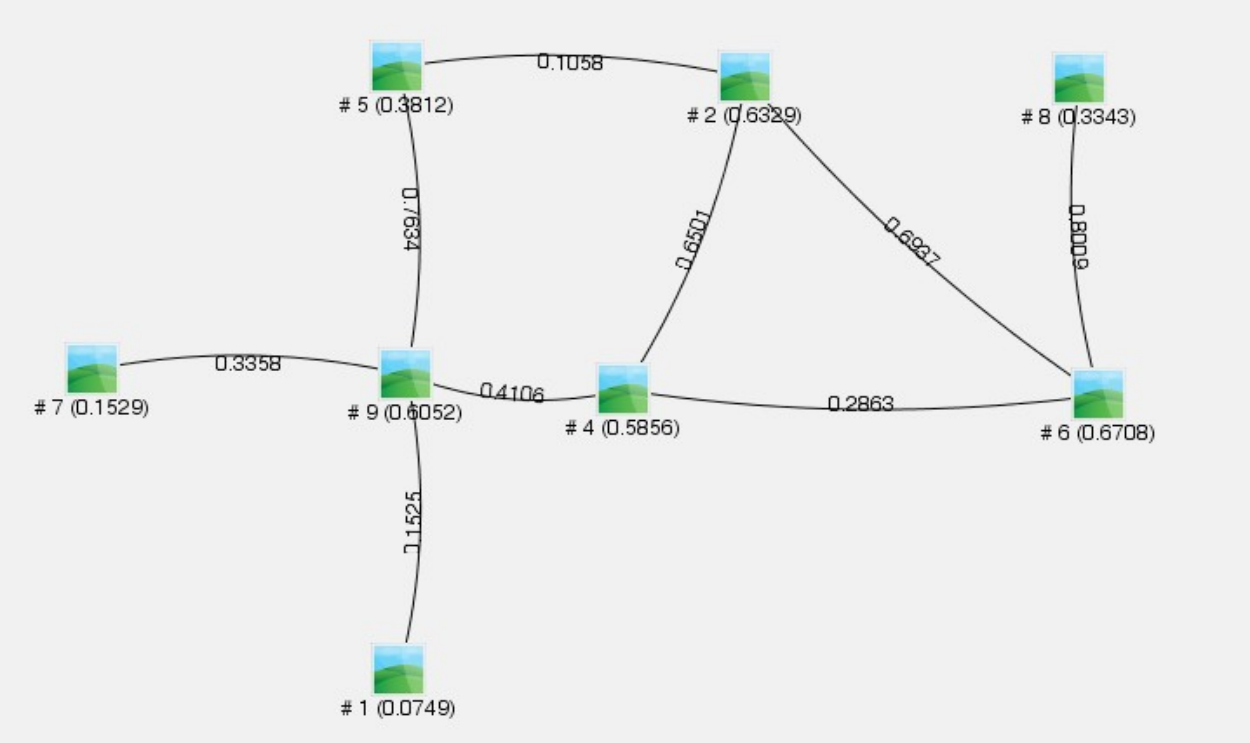
čia d – parinkta konstanta, n – skaitmeninių vaizdų skaičius (grafo viršūnių skaičius).



5.1. „Išsišokeliai“ (skaitmeninis vaizdas „#5” ir skaitmeninis vaizdas „#9”), kurių kiekvieno įvertis lygus (1-d)p ≈ 0,01
(d = 0,8; |V| = 10; |E| = 8)

Kiekvienoje skaitmeninio vaizdo įverčio skaičiavimo iteracijoje (žr. (14) formulę) gretimumo matrica yra sudauginama su tikriniu vektoriumi – kaimyninių viršūnių įverčiai dauginami su briaunų, jungiančių šias viršūnes, įverčiais. Tai reiškia, kad skaitmeninių vaizdų įverčiai (nurodyti skliausteliuose) priklauso nuo incidenčių briaunų įverčių bei „svarbių“ kamyninių viršūnių įverčių. Sąryšiai su aukštesnį įvertį turinčiomis viršūnėmis daro didesnę įtaką nagrinėjamos viršūnės įverčiui nei sąryšiai su mažesnį įvertį turinčiomis viršūnėmis. Todėl viršūnės įvertis yra didesnis kuo

pastaroji yra arčiau „svarbios“ viršūnės. Pavyzdžiui, pažvelgus į 5.2 paveikslėlį, matome, kad svarbiausia (viršūnė turinti didžiausią įvertį) viršune yra laikoma viršūnė, kurios numeris yra 6, nepaisant to, kad daugiausia kaimynų turi viršūnė, kurios numeris yra 9. Taip yra todėl, kad skaitmeninis vaizdas (viršūnė „#6”) yra panašus į kitą skaitmeninį vaizdą, kuris yra įvertintas auštu įverčiu (viršūnė „#2”,), t.y. laikomas svarbiu užklausos rezultatų aibėje.



5.2. Panašumo grafas, kuris buvo sugeneruotas darbo metu sukurtos taikomosios programos (d = 0,8; |V| = 8; |E| = 9)

Pateiksime pavyzdį, kaip apskaičiuojamas grafo viršūnės įvertis. Tarkime turime grafą, kuris yra pavaizduotas 5.3. pav. Šio grafo gretimumo matrica yra:

$$A = \begin{vmatrix} 0 & 0.8721 & 0 & 0 & 0 & 0.8761 & 0 & 0 & 0.8316 & 0 \\ 0.8721 & 0 & 0 & 0.2283 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.9099 & 0 \\ 0 & 0.2283 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.5889 & 0 & 0 & 0.5064 & 0.6826 \\ 0.8761 & 0 & 0 & 0 & 0.5889 & 0 & 0.3306 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0.3306 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0.4017 \\ 0.8316 & 0 & 0.9099 & 0 & 0.5064 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0.6826 & 0 & 0 & 0.4017 & 0 & 0 \end{vmatrix}$$

Tarkime, kad norime apskaičiuoti skaitmeninio vaizdo „#0“ įvertį. Skaičiavimams naudosime (14) formulę. Parenkame konstantą *d*, tarkim *d* = 0,8. Apskaičiuojame grafo eilę: |V| = 10. Apskaičiuojame *p*:

$$p_i = \frac{1}{10}.$$

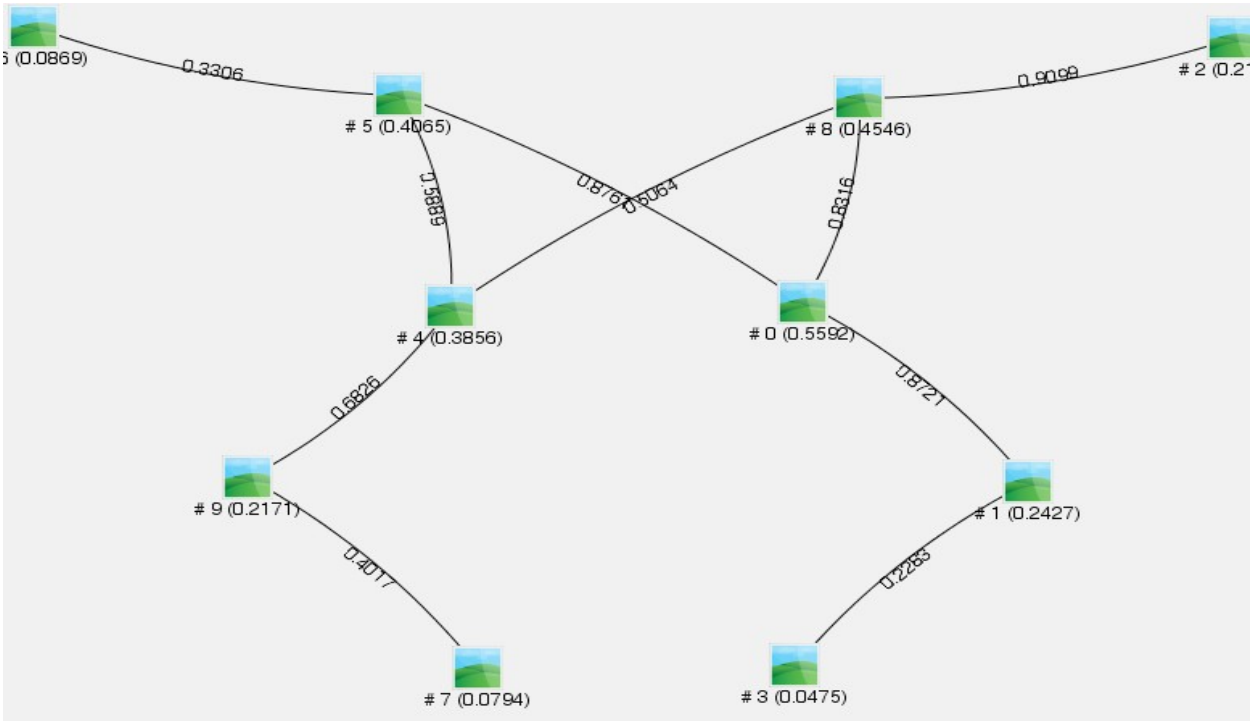
Parenkame pradinį nenulinį vektorių *I*:

$$I = (1;1;1;1;1;1;1;1;1;1),$$

Taikome formulę. Po pirmos iteracijos gauname:

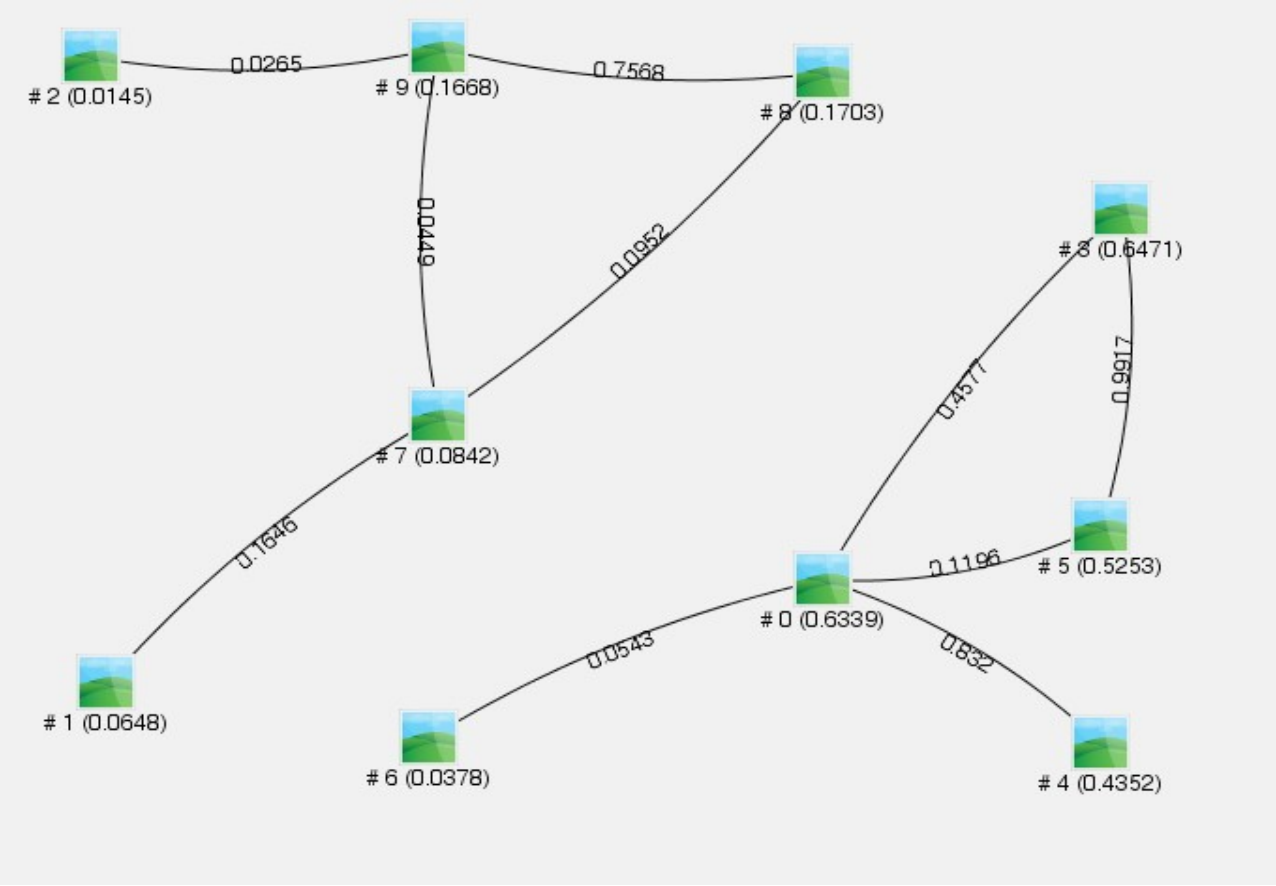
$$I = (0.4875;0.3250;0.1412;0.0538;0.3635;0.4415;0.0631;0.1055;0.4192;0.3402),$$

Tęsiame (14) formulės taikymą. Iteratyviai skaičiuojame vektoriaus *I* reikšmę. Kiekvienos iteracijos metu normalizuojame vektorių. Formulę taikome tol, kol reikšmė pradeda konverguoti. Iteravimas baigiamas priklausomai nuo siekiamo tikslumo. Skaitmeninio vaizdo „#0“ įvertį atitiks vektoriaus *I* pirmoji koordinatė (bendruoju atvjeu – *i*-ojo skaitmeninio vaizdo įvertis lygus *i*-ajai tikrinio vektoriaus koordinatei). Analogiškai apskaičiuojami kitų viršūnių įverčiai.



5.3. Panašumo grafas, kuris buvo sugeneruotas darbo metu sukurtos taikomosios programos (d = 0,8; |V| = 10; |E| = 10)

Analizuojant skaitmeninius vaizdus gali būti aptiktos dvi ar daugiau dominuojančių objektų aibės. Šių objektų gausa suformuos taip vadinamas „tematikas“ (semantiškai panašių vaizdų grupes). Objektų dominavimas bus pastebimas tik tam tikruose poaibiuose. Dėl to, sukonstruojamas ne vienas, o keli panašumo grafai (žr. 5.4. pav.). Šis reiškinys yra vadinamas viršūnių grupavimu (angl. clustering). Esant tokiai situacijai reikia nuspręsti, kurios grupės elementus grąžinti subjektui, t.y. kuris poaibis yra svarbesnis. Yra įvairių pasiūlymų, kaip spręsti šią problemą. Vienas iš jų, tai siūlymas atrinkti grafą remiantis jo dydžiu, t.y. didžiausias grafas laikomas svarbiausiu grafu [JBR07]. Galima taip pat skaičiuoti visų briaunų įverčių vidurkį ir pagal tai parinkti duomenų struktūrą. Taip pat galima tiesiog imti visus elementus bei prioretizuoti nekreipiant dėmesio į esamas grupes.



5.4. Grupių (angl. cluster) susidarymas
(d = 0,8; $|V_1| = 5$; $|E_1| = 5$;$|V_2| = 5$; $|E_2| = 5$)

Skaičiavimuose naudojama konstanta *d.* Pasak [JBR07] šaltinio, įverčio skaičiavimo algoritmas

pateikia geriausius rezultatus kai $d \geq 0,8$. Šio tiriamojo darbo metu buvo nustatyta, kad net ir $d \geq 0,4$ pateikia beveik tokio pat tikslumo rezultatus. Iš esmės šio dydžio parinkimas neturi didelės įtakos galutiniam rezultatui, t.y. prioretizavimui.

Kaip ir minėta, kiekvienos grafo viršūnės įvertis apskaičiuojamas iteraciškai taikant (14) formulę. Įvertis konverguoja į matricos tikrinį vektorių. Praktiškai pritaikius įverčio skaičiavimo algoritmą, buvo pastebėta, kad reikšmės konverguoti pradeda vidutiniškai nuo 15 – 50 iteracijos:

...
0.5926064175522527
0.6758526514021614
...
0.7543321631431149
…
0.7554709978900356
...
0.7554709978900731

5.5. pav. Grafo vršūnės įverčio konvergavimas po 1000 iteracijų

6. Panašumo grafo konstravimas

Esminė pasiūlytojo paieškos algoritmo dalis – skaitmeninių vaizdų įverčio skaičiavimas. Remiantis šiuo dydžiu įvertinamas rezultatų aibės elementų aktualumas subjektui. Įverčio paieška vykdoma naudojant panašumo grafą bei taikant jame tikrinio vektoriaus centriškumo metodą. Kiekviena grafo viršūnė atitinka skaitmeninį vaizdą, o briauna sąryšį tarp vaizdų, kuris išreiškiamas priskiriant briaunai skaliarą, t.y. įvertį. Skaitmeniniai vaizdai jungiami tarpusavyje remiantis jų panašumu. Kuo vaizdai turės daugiau bendrų bruožų, tuo jie bus laikomi panašesni. Atitinkamai briaunos, jungiančios šiuos vaizdus, įvertis bus didesnis.

Darbo metu buvo nutarta skaitmeninių vaizdų analizės procesui, kuriuo remiantis bus formuojamas panašumo grafas, naudoti SIFT algoritmą (žr. 1.8 poskyrį). Tačiau šis algoritmas naudojamas tik dalinai. Jis naudojamas tik įdomumo sričių aprašams generuoti (SIFT vektorių deskriptoriams generuoti). Skaitmeninių vaizdų panašumui įvertinti naudojamos kitos duomenų struktūros.

6.1. Optimizacija

Aprašytojo proceso metu (žr. 4.1. poskyrį) teigiama, kad panašumą tarp dviejų skaitmeninių vaizdų atitinka skaičius bruožų, kurie aptinkami abiejuose vaizduose. Šio žingsnio metu ieškoma ne

tik vienodų (identiškai) deskriptorių, bet ir tarpusavyje panašių. Kitaip tariant, net ir jei nerandama visiškas atitikmuo, į panašumo grafą įtraukiami ir panašius deskriptorius turintys skaitmeniniai vaizdai. Šiam tikslui pasiekti naudojama artimiausių kaimynų paieška. Panašumo kriterijui kontroliuoti naudojamas sutartinis slenkstis.

Algoritmo autoriai [JB08] ir [JBR07] darbuose artimiausio kaimyno paieškai pasiūlė naudoti kamuolinius medžus (žr. 3.1. poskyrį). Manau, kad pakankamai gerų rezultatų galima tikėtis panaudojus kur kas paprastesnę daugiamačių duomenų struktūrą – KD medį (1.10. poskyrį).

6.1.1. Prielaidos

Vienas esminių skaitmeninių vaizdų įverčio skaičiavimo algoritmo neigiamų bruožų – tai algorimo sparta. Skaitmeninių vaizdų apdorojimas bei gautųjų rezultatų analizė yra „brangi“ procedūra. Suformavus skaitmeniniui vaizdui požymių deskriptorus, reikia surasti jiems artimus variantus. Tai atliekama detaliai analizuojant pirmajame žingsnyje (atlikus teksto analize paremtą paiešką) surastą skaitmeninių vaizdų aibę. Kuo didesnė pradinė aibė, tuo didesnės yra apkrovos.

Formaliai aprašysime šią probleminę situaciją. Tarkime, kad po pirmojo skaitmeninių vaizdų įverčio skaičiavimo procedūros žingsnio, gauname skaitmeninių vaizdų aibę *I*:

$$I = i_1, i_2, \ldots, i_n \qquad (23)$$

čia n – skaitmeninių vaizdų kiekis, i_j, $j \in \mathbb{N}$ - skaitmeninis vaizdas.

Kuriant panašumo grafą, keikvienam nagrinėjamam skaitmeniniui vaizdui reikia įvertinti, kiek jis panašus į likusius aibės elementus. Jei turime n skaitmeninių vaizdų, tai reikės n kartų analizuoti n-1 skaitmeninį vaizdą. Iš viso reikės atlikti $n \cdot (n-1)$ iteraciją.

Kaip ir minėta, skaitmeniniai vaizdai yra panašūs, jei juose randama panašių deskriptorių. Panašiems deskriptoriams aptikti konstruojamos medžio tipo duomenų struktūros. Viena iteracija atitinka vienos duomenų struktūros kūrimą. Taigi analizuojant skaitmeninius vaizdus teks sukurti $n \cdot (n-1)$ medį.

Analizuojant pradinės aibės skaitmeninius vaizdus, kiekvienam iš jų sugeneruojami savybinių sričių deskriptoriai. Jei vienai bylai sugeneruojama m deskriptorių, tai daugiamatę duomenų struktūrą sudarys m elementų. Tai reiškia, kad reikės atlikti m įterpimo operacijų. Taigi kuo daugiau deskriptorų tuo daugiau įterpimo operacijų. Deskriptorių skaičius kiekvienam skaitmeniniui vaizdui

gali būti skirtingas.

Kaip ir minėta, nagrinėjamo paieškos proceso autoriai, deskriptorių indeksavimui, pasiūlė naudoti kamuolinius medžius (žr. 1.9. poskyrį). Šio medžio konstravimo metu naudojama pakankamai daug „brangių“ operacijų. Brangiausiai kainuoja kriterijaus, kuriuo remiantis erdvės taškai talpinami į kairįjį arba dešinįjį pomedžius, skaičiavimas. Kuo didesnė deskriptoriaus dimensija, tuo daugiau atliekama skaičiavimų.

KD medžiai pasižymi kur kas paprastesne struktūra (žr. 1.10. poskyrį). Norint alikti įterpimo operaciją į KD medį, reikia atlikti „pigias“ operacijas. Duomenų įterpimui tereikia atlikti palyginimo operaciją bei, jei reikia skaldyti viršūnę, skaitliuko, kuris žymi skaldomą dimensiją, pasukimą. Dažniausiai skaldymo dimensija parenkama ciklišKai.

Centriniame procesoriuje aritmetines bei logines operacijas atlieka aritmetinis loginis įrenginys (angl. ALU - Arithmetic Logic Unit). Loginės operacjos atliekamos daug greičiau nei aritmetinės. Gauname, kad įterpimo operacija KD medžiuose yra kur kas greitesnė nei kamuoliniuose medžiuose. Todėl sukonstruoti $n \cdot (n-1)$ KD medį galima kur kas greičiau nei atitinkamą kiekį kamuolinių medžių. Pasinaudojus šia savybe, galima daryti prielaidą, kad aptartasis skaitmeninių vaizdų algoritmas, panaudojus KD medžius taip pat veiks greičiau. Žinoma, tam kad KD medžiai paspartintų algoritmo veikimą, reikia įvertinti artimiausio kaimyno paieškos spartą abejose duomenų struktūrose.

6.1.2. Techninės detalės

Šiame darbe buvo naudotos Java pagrįstos technologijos. Operacinė sistema, kurioje buvo atliktas bylų perrinkimas – Mac OS X 10.6.8. Įgyvendinus aptariamą paieškos procedūrą bei siūlomą modifikaciją, panaudojant kitokias technologijas, metrikos gali skirtis nuo tų, kurios yra pateikiamos sekančiame poskyryje. Nagrinėjamo proceso efektyvumas gali būti gerinamas keičiant minėtąsias technologijas.

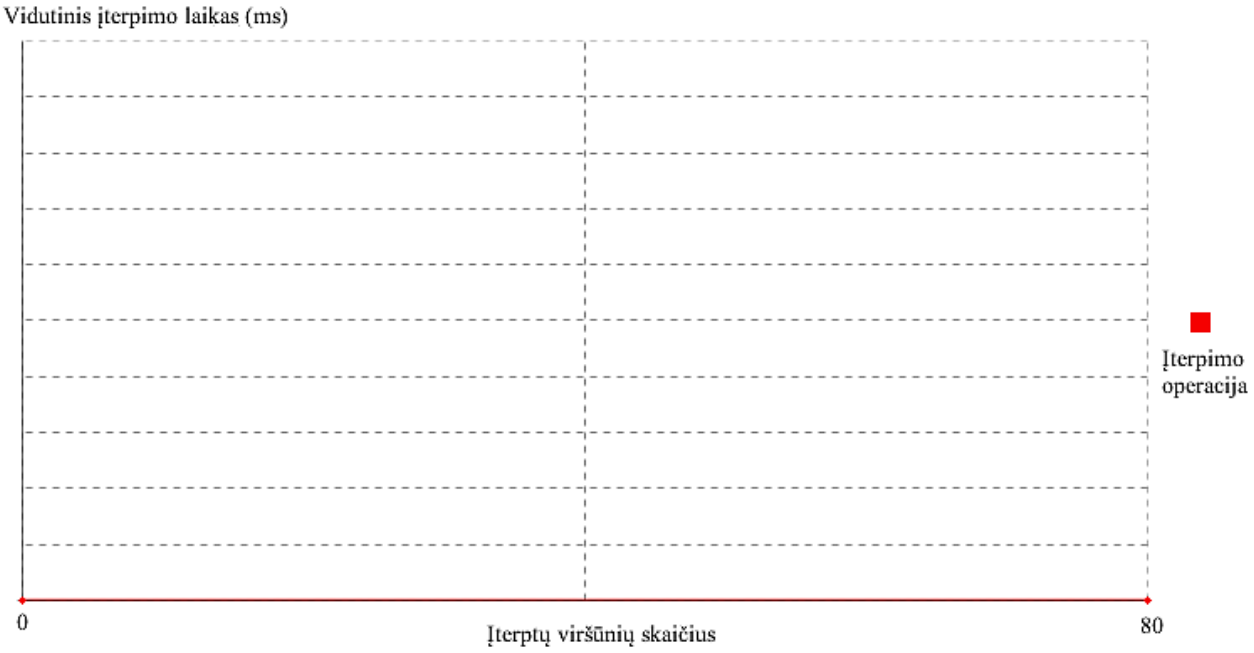
Baigiamojo magistrinio darbo metu buvo suprogramuotas prototipas, kurio pagalba galima atlikti teksto analize paremtą paiešką, panašumo grafo konstravimą, skaitmeninių vaizdų įverčio skaičiavimą bei rezultatų prioretizavimą. Darbo metu taip pat suprogramuota aptariama algoritmo modifikacija. Pastaroji buvo integruota į minėtąją programą. Siekiant atlikti algoritmo lyginamąją analizę, buvo pasinaudota „Weka” duomenų analizės įrankyje (sukurtas Waikato universitete) pateiktu kamuolinio medžio įgyvendinimu (įrankis yra atvirojo kodo). Visa tai buvo taip pat

integruota į darbo metu kuriamą taikomąją programą. Šiuo žingsniu buvo siekiama užtikrinti rezultatų nešališkumą (vykdant lyginamąją analizę).

6.1.3. Analizė

Panagrinėkime, kiek užtrunka sukurti kamuolinį medį (žr. 1.9. poskyrį). Kamuolinių medžių viršūnėse saugomi ne pavieniai taškai, bet taškų aibės. Jei, įterpus tašką, viršūnė persipildo, tai atliekama skaldymo (angl. split) operacija. Vienos skaldymo operacijos metu atliekamas Euklidinio atstumo skaičiavimas (6), skaičiuojamos projekcijos (7), randamas slenkstis bei atliekamos loginės operacijos (visi taškai, kurie yra mažesni už rastąjį slenkstį, talpinami į kairįjį vaiką, o visi taškai, kurie yra didesni už jį – į dešininį vaiką). Žvelgiant iš techninės pusės, šis žingsnis gali būti įgyvendinamas įvairiai. Pavyzdžiui, visi minėtieji skaičiavimai gali būti atlikti kiekviename lygyje. Alternatyva – atlikti didžiąją skaičiavimų dalį pirmajame lygyje. Šio žingsnio metu, kiekvienai viršūnei gali būti priskiriama reikšmė, kuri nurodytų atstumą iki pirmajame lygyje apskaičiuotojo slenksčio. Ši reikšmė būtų naudojama vaikščiojimo po medį metu. Sekančiuose lygiuose nereikėtų perskaičiuoti slenksčio. Pakaktų panaudoti saugomąją reikšmę. Pastarasis variantas pateikia kur kas geresnius rezultatus. Praktiškai atliekant analizę, erdvės skaldymui buvo naudojama pastaroji strategija.

KD medis konstruojamas daug greičiau. Konstruojant šį medį nereikia skaičiuoti euklidinių atstumų bei atlikti kitokių aritmetinių operacijų. Įterpimo metu atliekamos tik loginės operacijos. Duomenų įterpimo laikas yra artimas nuliui (žr. 6.1 pav.)



6.1 Pav. Duomenų itėrpimo į KD medį greitis. Naudojami 128 matmenų duomenys.

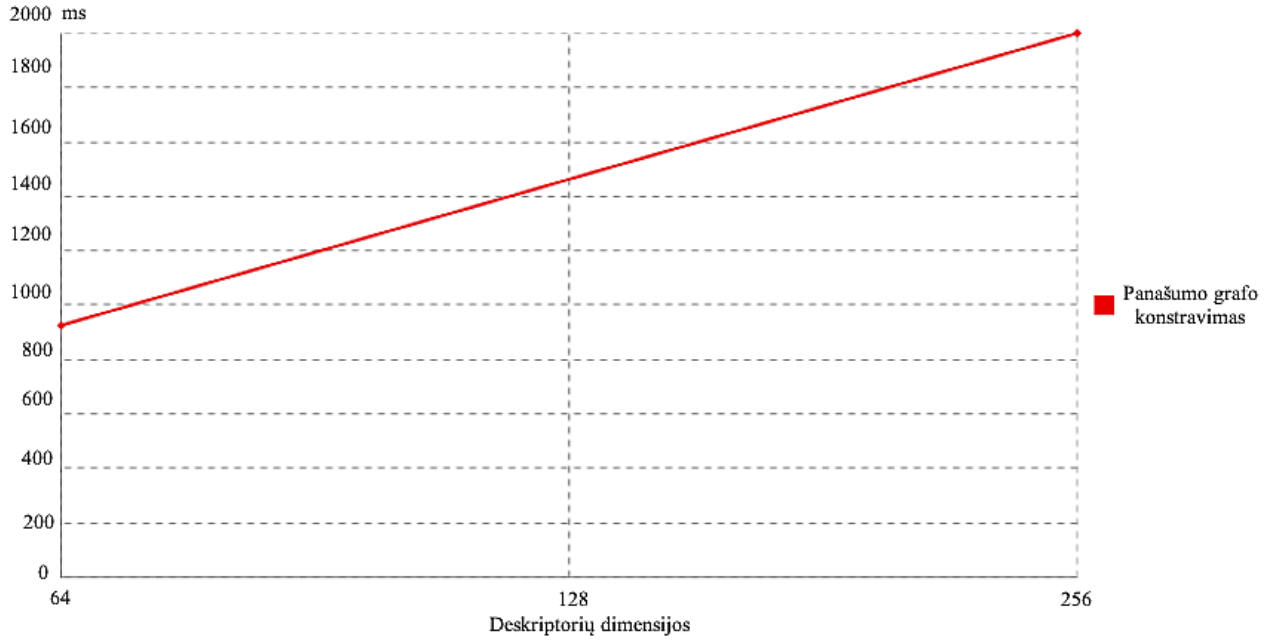
6.1 pateiktoje schemoje pavaizduota įterpimo operacija. KD medis buvo konstruojamas iš 100 deskriptorių. Kiekvienas deskriptorius yra 128 matmenų.

KD medžiai konstruojami daug greičiau nei kamuoliniai medžiai. Kuo daugiau deskriptorių, tuo KD medis darosi pranašesnis. Pradinės darytos prielaidos yra teisingos. Norint įsitikinti, kad skaitmeninių vaizdų paieškos procesas taip pat paspartėja, reikia įvertinti artimiausio kaimyno paieškos spartą (panaudojus minėtąsias duomenų struktūras). Šio žingsnio įvertinimas yra labai svarbus. Konstruojant kamuolinius medžius, erdvės skaldymas užtrunka sąlyginai ilgai. Tačiau šios duomenų struktūros pasižymi ypač naudinga (žvelgiant iš aptariamojo paieškos proceso perspektyvos) savybe – artimiausio kaimyno paieška atliekama pakankamai greitai.

Artimiausio kaimyno paieška KD medyje blogiausiu atveju tampa tiesine – apeinamos beveik visos medžio viršūnės. Norint optimizuoti skaitmeninių vaizdų algoritmą, reikia, kad konstravimo sparta atsvertų artimiausio kaimyno paieškos nuostolius. Jeigu įmanoma tokia situacija, tai KD medžiai yra puiki alternatyva kamuoliniams medžiams. Tokiu atveju galima naudoti duomenų struktūrą, kurios įgyvendinimas yra daug paprastesnis bei, žinoma, spartesnis. Kadangi beveik išvengiama aritmetinių operacijų (euklidinis atstumas naudojamas pakankamai retai bei tik artimiausio kaimyno paieškos metu), tai galima tikėtis atlikti nagrinėjamo skaitmeninių vaizdų

paieškos proceso įgyvendinimą panaudojus kur kas paprastesnes technologijas. Atsiveria platesnės galimybės panašumo grafų realizacijai.

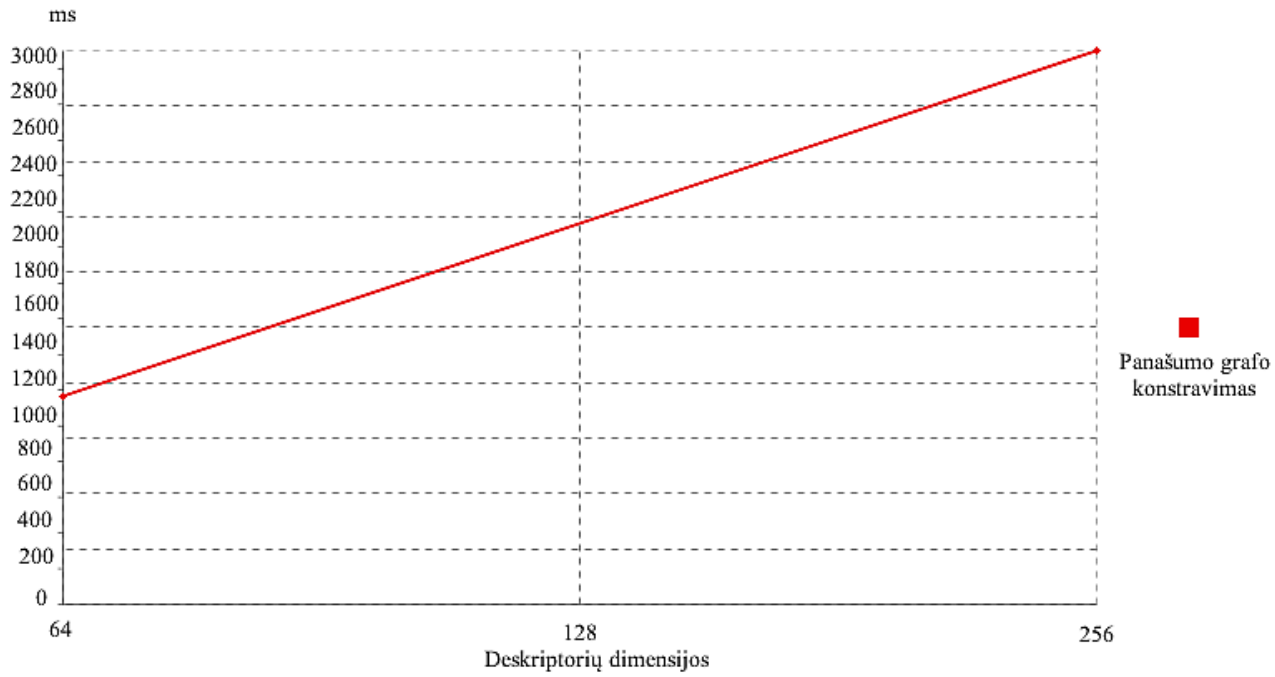
Norint galutinai patvirtinti, kad 6.1.1 poskyryje visos darytos prielaidos yra teisingos, reikia atlikti panašumo grafo konstravimo spartos įvertinimą.



6.2 Pav. Panašumo grafo konstravimo sparta panaudojus KD medį.

Aukščiau pateiktojoje schemoje pavaizduotas grafikas, kuris atspindi panašumo grafo spartą, kuomet daugiamačių duomenų indeksavimui naudojamas KD medis. Testiniai duomenys buvo suskirstyti į tris kategorijas. Kiekvienoje iš jų deskriptorių dimensijos skyrėsi. Vienas deskriptorius žymi vieną įdomumo taško aprašą. Priklausomai nuo SIFT algoritmo metu parenkamos aprašui genereuoti kaimyninės srities (įdomumo taškas yra šios srities centrinis elementas), deskriptoriaus dydis (duomenų indeksavime reiškiantis matmenis) kinta. Tyrimo metu buvo naudojami 64, 128 ir 256 matmenų vektoriai.

Žemiau pateiktajame grafike pavaizduoti analogiško tyrimo rezultatai. Šiuo atveju duomenų indeksavimui buvo naudojami kamuoliniai medžiai. Palyginus gautuosius rezultatus, matome, kad užsibrėžtas tikslas įvykdytas. Nepriklausomai nuo dimensijų kitimų, skaitmeninių vaizdų paieškos procesas atliekamas sparčiau pirmuoju atveju.



6.3 Pav. Panašumo grafo konstravimo sparta panaudojus Kamuolinį medį.

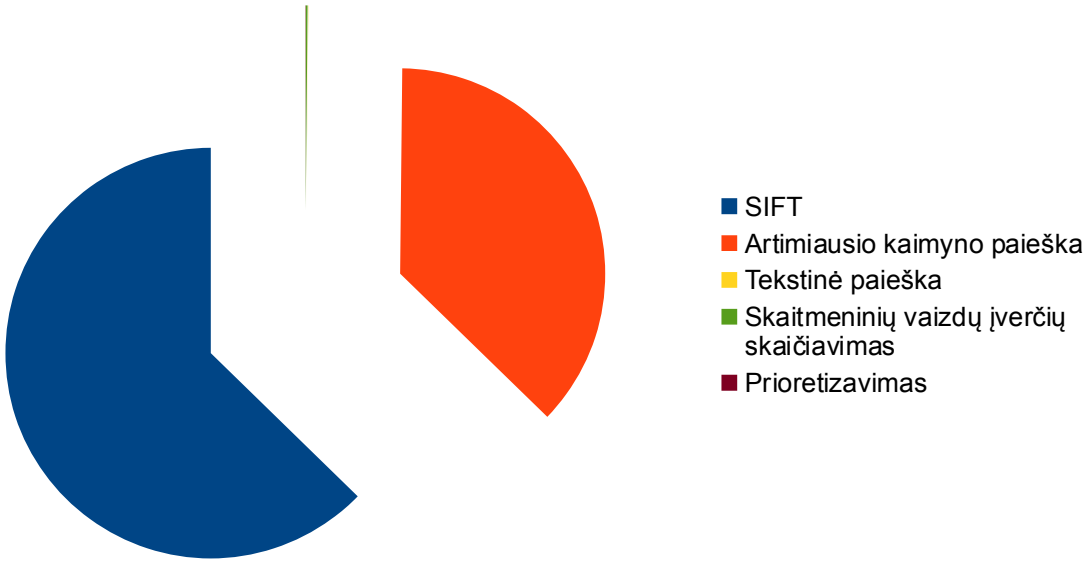
7. Prototipas

Šio darbo rezultatas – sistema, kuri yra pritaikyta skaitmeninių vaizdų, esančių viename kompiuteryje, paieškai atlikti. Sukurtasis prototipas gali veikti dviem režimais:

1. Autonominis režimas.
2. Prototipas susietas su antrine sistema, surenkančia bei išsaugojančia informaciją, reikalingą efektyviam prototipo veikimui.

Autonominis režimas – tai sukurtosios programos veikimas neisnaudojant jokiais papildomais resursais. Visos darbo apkrovos tenka pačiai programai. Tekstinė paieška, skaitmeninių vaizdų analizė, deskriptorių generavimas bei indeksavimas, panašumo grafo konstravimas ir skaitmeninų vaizdų įverčių skaičiavimas vykdomas vartotojui įvedus raktažodžius bei aktyvavus paiešką. Esminis trūkumas – kiekvieną kartą, vartotojui įvedus paieškos kriterijų, nuosekliai vykdomi visi algoritmo žingsniai. Net ir atlikus algoritmo optimizaciją (žr. 6.1 skyrių), skaitmeninių vaizdų paieška vykdoma sąlyginai lėtai. Daugiausiai veikimo laiko užima skaitmeninių vaizdų apdorojimas (žr. antrąjį skaitmeninių vaizdų įverčio skaičiavimo procedūros žingsnį). Vykdant paiešką autonominiu režimu, dažnai apdorojami tie patys duomenys. Vietoj to, kad pastarieji būtų panaudoti

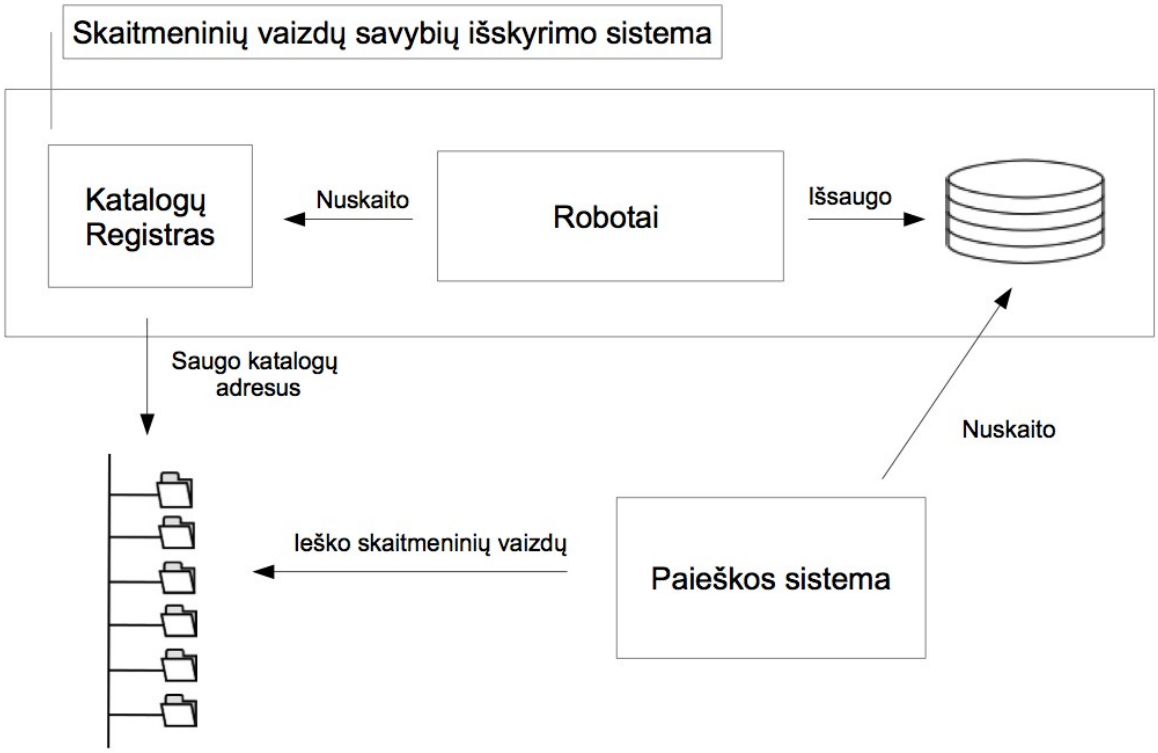
kitiems skaičiavimams atlikti, viskas vykdoma iš naujo.



7.1. Pav. Algoritmų veikimo trukmė

Alternatyva autonominiui režimui yra pagalbinės sistemos naudojimas. Toks prototipo veikimas yra paremtas saityno paieškos variklių veikimu. Saityne, vartotojui įvedus paieškos kriterijų, rezultatai grąžinami analizuojant puslapius, kurie buvo apdoroti, suindeksuoti bei įvertinti gerokai anksčiau nei sistemos naudotojas nutarė atlikti paiešką. Tokiose sistemose puslapių apdorojimas vykdomas naudojant robotus (angl. bot), kurie „šliaužia“ per saityno puslapius bei analizuoja kiekvieną iš jų. Apdoroti duomenys talpinami į specialias saugyklas (nutolusius serverius) bei sistemos naudotojui vykdant paiešką panaudojami sparčiai paieškai atlikti.

Darbo metu paieškos spartai užtikrinti (išvengiant kokybės sumažėjimo) nutarta pasiskolinti aukščiau aprašytąją idėją bei pritaikyti ją viename kompiuteryje esantiems skaitmeniniams vaizdams aptikti. Šiuo atveju norint atlikti efektyvią paiešką, pirmiausia reikia išskirti deskriptorius iš sistemoje esančių skaitmeninių vaizdų. Kadangi šių vektorių apskaičiavimas užtrunka ilgiausiai (žr. 7.1. pav.), tai jų išankstinis apdorojimas ir panaudojimas tiek kartų, kiek reikia tolimesniems skaičiavimams, žymiai paspartina skaitmeninių vaizdų paiešką. Prototipo veikimo schema pavaizduota 7.2 paveikslėlyje.

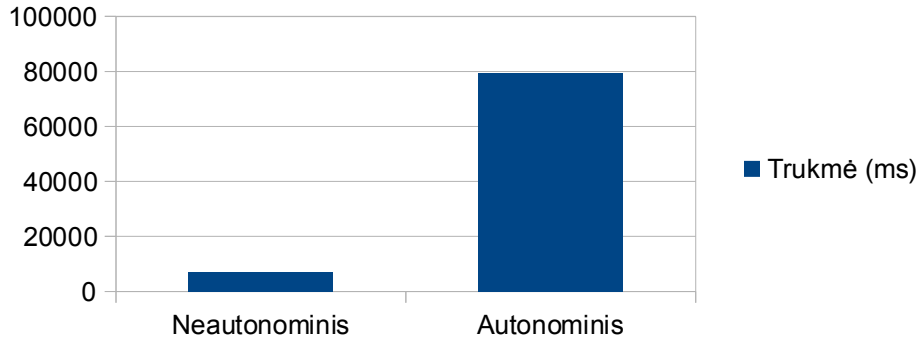


7.2. Pav. Prototipo veikimo schema (neautonominis režimas)

Norint išskirti skaitmeninių vaizdų savybes, pirmiausiai reikia žinoti, kuriuose katalogose leidžiama (subjektas galės vykdyti) paieška. Visi šie adresai yra saugomi katalogų registre (žr. 7.2. pav.). Turint šią informaciją pradedama skanuoti sistemą. Šį darbą atlieka specialūs suprogramuoti robotai (programos). Jie rekursyviai vaikšto po katalogų registre pateiktus adresus bei, suradę skaitmeninį vaizdą, išskiria jame esančias savybes (deskriptorius). Išskirtieji deskriptoriai saugomi specialioje saugykloje. Apdorojus visus skaitmeninius vaizdus, esančius užregistruotuose kataloguose, sistema tampa paruošta.

Norint surasti skaitmeninius vaizdus, vykdoma skaitmeninių vaizdų įverčių skaičiavimo procedūra (žr. 4.1 poskyrį). Remiantis gautaisiais rezultatais vykdomas skaitmeninių vaizdų prioretizavimas bei pateikimas subjektui. Šias operacijas galima atlikti naudojant darbo metu suprogramuotą paieškos programą. Naudojantis šia programa tereikia nurodyti raktažodžius bei katalogą, kuriame (bei rekursyviai ieškant pakatalogiuose) siekiama surasti skaitmeninį vaizdą (arba vaizdus). Tiesa, skirtingai nei autonominiu režimu, šiuo atveju skaitmeninių vaizdų savybės yra paimamos iš saugyklos, kurią užpildė suprogramuotieji robotai. Spartos palyginimas pateiktas 7.3

paveikslėlyje.



7.3. Pav. Paieškos sparta panaudojus autonominį ir neautonominį (naudojama antrinė sistema) režimą. Paieškai panaudoti 25 skaitmeniniai vaizdai.

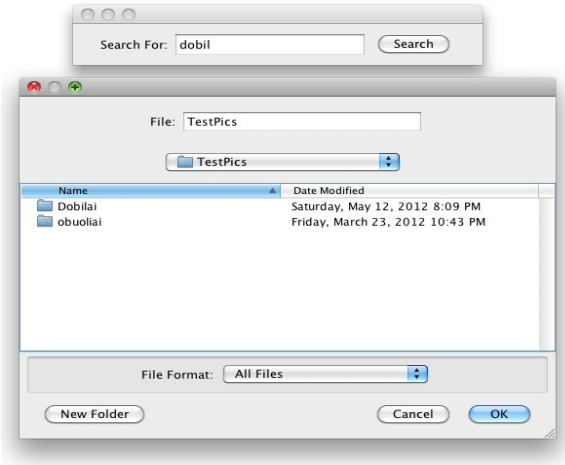
Skaitmeniniai vaizdai pateikiami varotojui analizuojant tiek tekstinę informaciją, tiek ir grafinę. Pradinė skaitmeninių vaizdų aibė randama atliekant standartinę teksto analize paremtą paiešką. Vėliau, gavus rezultatų aibę, atliekama skaitmeninių vaizdų analizė bei įvertinamas kiekvienas pradinės aibė elementas, t.y. skaitmeninis vaizdas. Tada atliekamas rūšiavimas pagal skaitmeninių vaizdų įverčius. Sistemos naudotojui pateikiami vaizdai nuo labiausiai užklausą tenkinančio skaitmeninio vaizdo, iki mažiausiai tenkinančio. Mažiausiai susiję su pateiktąja užklausa vaizdai pateikiami paskutiniai. Naudojant darbe aptartąjį algoritmą galima objektyviai įvertinti, subjektyvią sistemos savininko požiūrį į tai, kas yra svarbu.

Žinoma, pasiūlytąją paiešką galima dar patobulinti. Galima pašalinti mažiausią įvertį turinčius skaitmeninius vaizdus iš rezultatų aibės interpretuojant pastaruosius kaip netinkamus rezultatus. Tačiau tokiu atveju susiduriama su problema. Reikia parinkti slenkstį, kuris būtų naudojamas nustatyti ribą, kuri skirtų tinkamus skaitmeninų vaizdų įverčius nuo netinkamų.

7.1. Prototipo panaudojimas

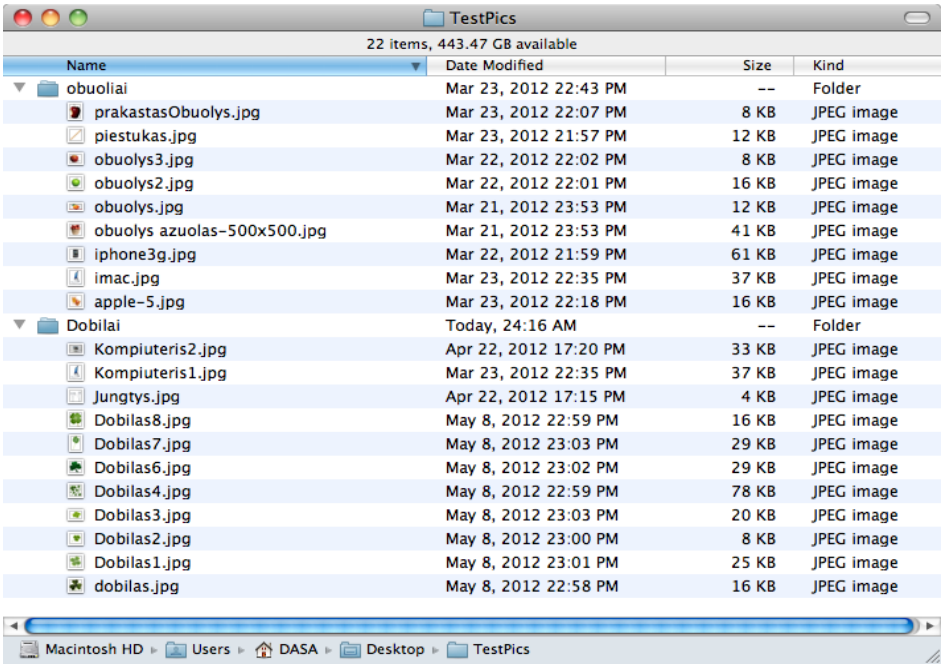
Aprašysime prototipo panaudojimą realioje sistemoje skaitmeninių vaizdų paieškai atlikti bei pateiksime gautuosius rezultatus. Naudojama operacinė sistema: Mac OS X 10.6.8.

Skaitmeninių vaizdų paieška pradedama aktyvavus programą, įvedus paieškos žodį bei pasirinkus katalogą, kuriame ieškoma skaitmeninių vaizdų.



7.4 Pav. Paieškos aktyvavimas. Įvedamas raktinis žodis bei parenkamas šakninis katalogas.

Sistemos naudotojas įveda pilną paieškos žodį arba dalį jo bei nurodo šakninį katalogą, kuriame (bei rekursyviai ieškant jame esančiuose pakatalogiuose) ieškoma užklausą tenkinančių skaitmeninių vaizdų. Paieška vykdoma dviem etapais. Pirmiausiai atliekama teksto analize paremta paieška, o vėliau apdorojami skaitmeniniai vaizdai. Žemiau pateiktoje iliustracijoje pavaizduoti skaitmeniniai vaizdai, esantys paieškos metu nurodytame kataloge.



7.5 Pav. Skaitmeniniai vaizdai, esantys nurodytame šakniniame kataloge

Įvedus raktinį žodį „dobil“ pradedama paieška. Atlikus tekstinę analizę, nustatoma, kad subjektui turi būti grąžinti visi skaitmeniniai vaizdai, kurie yra patalpinti kataloge pavadinimu „Dobilai“. Nustatyta, kad kiekvienos bylos absoliučiame adrese aptinkamas žodis „dobilai“. Dalis jo sutampa su paieškos žodžiu „dobil“. Paieškos sistema interpretuoja šį faktą taip, kad skaitmeniniai vaizdai yra susiję su paieškos žodžiu.

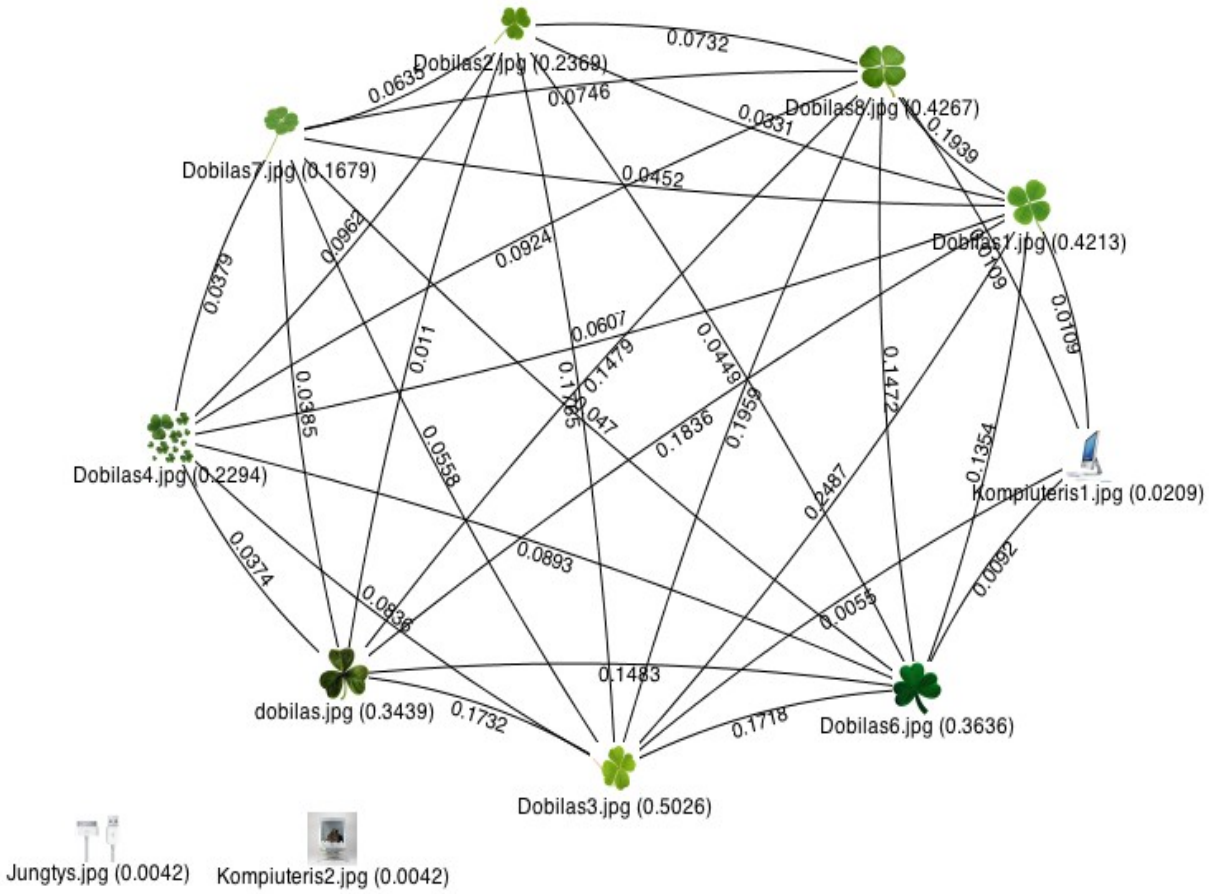
Pabaigus paiešką grąžinami 7.6 paveikslėlyje pateikti skaitmeniniai vaizdai. Matome, kad skaitmeniniai vaizdai, kuriuose yra pavaizduotas dobilas, pateikiami kartu su skaitmeniniais vaizdais, kuriuose yra pateikta kompiuterinė technika. Skaitmeniniai vaizdai neturi tvarkos. Rezultatai pateikiami atsitiktine tvarka. Taip pat jie nėra semantiškai susiję.

7.6 Pav. Skaitmeniniai vaizdai gauti atlikus tekstinę paiešką

Norint prioretizuoti skaitmeninius vaizdus reikia atlikti skaitmeninių vaizdų apdorojimą. Prioretizacija leidžia nustatyti, kurie rezultatų aibės elementai yra labiausiai tenkinantys užklausą ir kurie mažiausiai tenkina užklausą. Turint šią informaciją galima arba surūšiuoti skaitmeninius vaizdus pagal jų svarbą, arba pašalinti iš rezultatų aibės skaitmeninius vaizdus, kurie nesusiję su vyraujančia tematika.

Skaitmeninių vaizdų panašumui bei vyraujančiai tematikai nustatyti surandamas panašumo grafas. 7.7 paveikslėlyje pateiktas panašumo grafas, atitinkantis 7.6 paveikdlėlyje pateiktus skaitmeninius vaizdus. Pastebime, kad prototipas indentifikavo, kad skaitmeniniai vaizdai „Jungtys“ ir „Kompiuteris2“ yra išsišokeliai. Jie visiškai nesusiję su kitais aibės elementais. Dar vienas

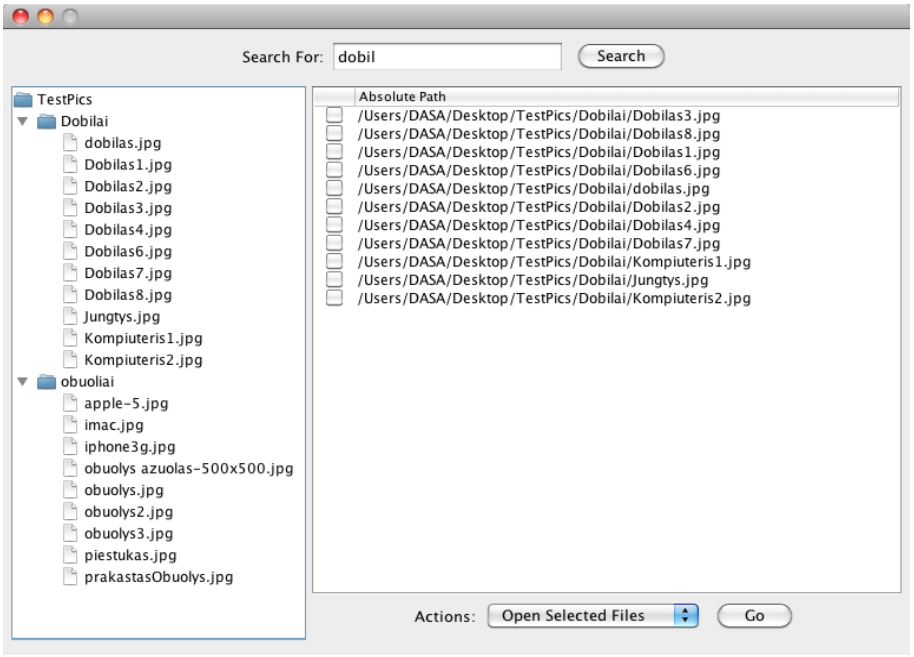
skaitmeninis vaizda („Kompiuteris1“) yra identifikuojamas kaip beveik nesusijęs su kitais aibės elementais. Pastarojo įvertis yra labai mažas. Prototipas nustatė, kad svarbiausias skaitmeninis vaizdas yra „Dobilas3“.



7.7 Pav. Panašumo grafas

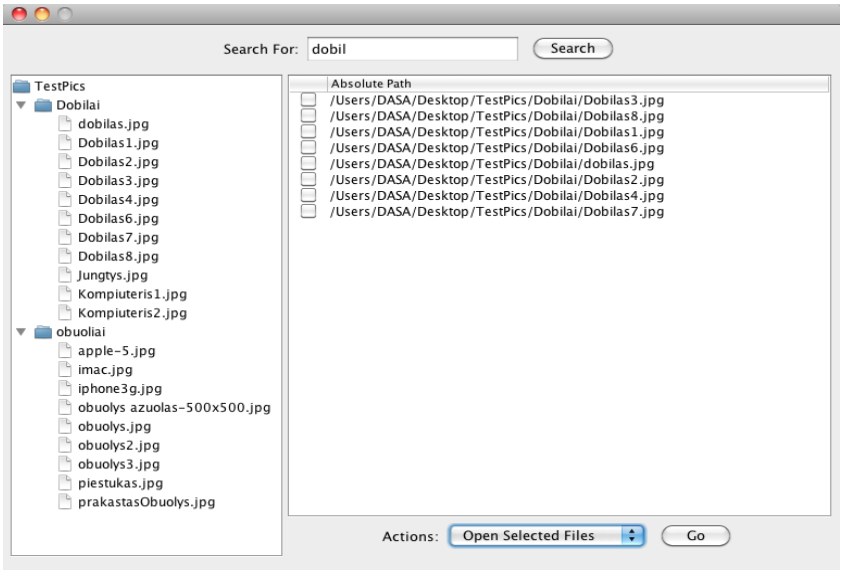
Lengva pastebėti, kad jei ne skaitmeninis vaizdas „Kompiuteris1“, panašumo grafas būtų pilnasis. Ši savybė atsiranda nustačius, kad likusiuose skaitmeniniuose vaizduose yra pavaizduotas labai panašus skatmeninis objektas. Kadangi šis objektas aptinkamas visuose skaitmeniniuose vaizduose, t.y., jie yra panašūs, reiškia jie priklauso tai pačiai tematikai. Prototipas apskaičiuoja skaitmeninių vaizdų tarpusavio panašumą. Panaudojus tikrinio vektoriaus centriškumo algoritmą bei skaitmeninių vaizdų panašumus, apskaičiuojamas įvertis. Išsišokelių įvertis yra konstanta.

7.8 paveikslėlyje pavaizduoti galutiniai paieškos rezultatai. Jie yra prioretizuoti. Pirmiausia pateikiami skaitmeniniai vaizdai, kuriuose yra pavaizduotas dobilas, o vėliau skaitmeniniai vaizdai, kuriuose yra pavaizduota kompiuterinė technika.



7.8 Pav. Paieškos rezultatai. Skaitmeniniai vaizdai prioretizuoti pagal kiekvieno iš jų įvertį.

Jeigu norima, galima išfiltruoti (panaudojus sutartinį slenkstį) mažiausiai užklausą tenkinančius rezultatus iš rezultatų aibės (žr. 7.9 Pav.).



7.9 Pav. Paieškos rezultatai. Naudojamas slenkstis išfiltruoti mažiausiai užklausą atitinkančius rezultatus.

8. Darbo rezultatai

Darbe buvo išanalizuotas skaitmeninių vaizdų įverčio skaičiavimo algoritmas, aptarti pastarojo privalumai bei trūkumai. Taip pat išanalizuotas skaitmeninių vaizdų prioretizavimas bei rezultatų pateikimas subjektui. Gilinamasi į neefektyvų rezultatų pateikimą sistemos naudotojui. Praktiškai įgyvendintas skaitmeninių vaizdų įverčio skaičiavimas panaudojant panašumo grafą bei tikrinio vektoriaus centriškumo algoritmą. Pakeista algoritmo taikymo sritis. Pasiūlyta daugiamačių duomenų indeksavimui naudoti KD medžius. Taip pat pasiūlytas techninis sprendimas leidžiantis efektyviai atlikti skaitmeninių vaizdų paiešką bei rezultatų pateikimą panaudojant skaitmeninių vaizdų įverčių skaičiavimo algoritmą. Realizuotas prototipas, kuris pateikia patobulintą skaitmeninių vaizdų paieškos funkciją. Skirtingai nuo egzistuojančių operacinėse sistemose pateikiamų paieškos funkcijų, prototipas leidžia atlikti paiešką analizuojant skaitmeninių vaizdų turinį.

Darbo metu altiktas praktinis prototipo panaudojimas. Aprašyti gautieji rezultatai. Naudojant sukurtąjį prototipą skaitmeninių vaizdų paieška atliekama greitai bei leidžiai atlikti paiešką panaudojant ne tik tekstinę informaciją, bet ir skaitmeninų vaizdų grafinę informaciją. Skaitmeniniai vaizdai pateikiami prioretizuotai, priklausomai nuo apskaičiuotųjų skaitmeninių vaizdų įverčių.

Išvados

Skaitmeninių vaizdų išrinkimo bei prioretizavimo metodas leidžia atlikti skaitmeninių vaizdų paiešką bei efektyvų rezultatų pateikimą sistemos naudotojui. Prototipas pateikia tikslesnius rezultatus nei standartinė operacinėse sistemose įdiegta paieškos funkcija. Kiekvienam skaitmeniniui vaizdui apskaičiuojamas įvertis, kuris objektyviai nusako tai, ką sistemos naudotojai subjektyviai laiko svarbiu turiniu. Rezultatai formuojami ne tik analizuojant globalias skaitmeninių vaizdų savybes, bet ir lokaliąsias.

Nepaisant pastebimų privalų šis metodas pasižymi ir esminiai trūkumais. Vykdant pateiktąjį įverčio skaičiavimo metodą susiduriama su dideliu kompiuterinių resursų poreikiu. Skaičiuojant skaitmeninių vaizdų įverčius, atliekama daug bei sudėtingų skaitmeninių vaizdų apdorojimo operacijų. Šios operacijos yra „brangios“ resursų atžvilgiu. Šiai problemai spręsti darbe buvo pasiūlyti du sprendimo būdai (algoritmo patobulinimas bei techninis sprendimas), kurie leido paspartinti prototipo veikimą. Taip pat nėra aišku, kokia yra riba, kai vis dar galima taikyti šį metodą. Skaitmeniniuose vaizduose aptinkami bendri bruožai – tai skaitmeninių vaizdų fragmentai, kurie duominuoja rezultatų aibėje. Tai reiškia, kad esant dideliems duomenų kiekiams, t.y. skaitmeninių vaizdų skaičius didelis, bei esant pakankamai dideliam kiekiui neteisingų vaizdų tarp jų, t.y. nesusijusių su tekstine užklausa, yra tikimybė, kad bus išskirti neteisingi bruožai. Tokiu atveju pastaroji situacija atitiks taip vadinamą triukšmą, kuris sąlygos klaidingas interpretacijas bei pateikiamus rezultatus subjektui. Panaudojus prototipą šiuo atveju gaunamas surūšiuotas skaitmeninių vaizdų sąrašas, kurio prioretizacija nebūtinai atitinka pateiktąją paieškos užklausą. Tačiau net ir dabar rezultatai pateikiami ne chaotiškai, bet prioretizuojant pagal nustatytasias vyraujančias tematikas.

Darbo metu buvo pasiūlyta ne tik pakeisti algoritmo taikymo sritį, bet ir panaudoti KD medžius daugiamačių duomenų indeksavimui. Tyrimo metu nustatyta, kad KD medžių panaudojimas ne tik supaprastina algoritmą, bet ir leidžia įvykdyti viename kompiuteryje esančių skaitmeninių vaizdų paiešką greičiau nei naudojant kamuolinius medžius.

Literatūros sąrašas

[AWM+09] Mohamed Aly, Peter Welinder, Mario Munich, Pietro Perona, Automatic Discovery of Image Families: Global vs. Local Features, Computational Vision Lab, Caltech, Pasadena, CA, USA, 2009, pp. 1-3.

[Bra01] Ulrik Brandes, A Faster Algorithm for Betweenness Centrality, University of Konstanz Department of Computer & Information Science, Konstanz, Germany, 2001, pp. 3-9.

[Cho05] Seungjin Choi, On Variations of Power Iteration, Department of Computer Science Pohang University of Science and Technology, Korea, 2005, pp. 1-2.

[CLR90] Th.H. Cormen, Ch.E. Leiserson, R.L. Rivest, Introduction to Algorithms, MIT Press, Mc Graw-Hill, 1990, pp. 308-309.

[CM09] Kino Coursey, Rada Mihalcea, Topic Identification Using Wikipedia Graph Centrality, 2009, pp. 1-2.

[JB08] Yushi Jing, Shumeet Baluja, PageRank for Product Image Search, College Of Computing, Georgia Institute of Technology, Atlanta GA, 2008, pp. 1-9.

[JBR07] Yushi Jing, Shumeet Baluja, Henry Rowley, Canonical Image Selection from the Web, College of Computing, Georgia Institute of Technology, Atlanta, GA , 2007, pp. 1-8.

[LMB+11] Dimitri A. Lisin, Marwan A. Mattar, Matthew B. Blaschko, Mark C. Benfield, Erik G. Learned-Miller, Combining Local and Global Image Features for Object Class Recognition, Computer Vision Laboratory, Dept. of Computer Science, University of Massachusetts, 2011, pp. 1-3

[LMG+11] Ting Liu, Andrew W. Moore, Alexander Gray and Ke Yang, An Investigation of Practical Approximate Nearest Neighbor Algorithms, School of Computer Science, Carnegie-Mellon University, 2011, pp. 2-3.

[Nor07] Stanislovas Norgėla, Logika ir Dirbtinis Intelektas, TEV, Vilnius, 2007, pp. 16-18.

[San05] Torsten Sander, An Introduction to Graph Eigenvalues and Eigenvectors, 2005, pp. 18-20.

[Man98] Eugenijus Manstavičius, Grafų teorija informatikos magistrantams, URL: http://www.mif.vu.lt/katedros/ttsk/bylos/man/files/grmag.pdf, 316 Kb, 1998 m.

[BE06] Stephen P. Borgatti, Martin G. Everett, A Graph-theoretic perspective on centrality, žurnalas Social Networks, Spalis 2006, pp. 466–484.

[BP98] Sergey Brin, Lawrence Page, The Anatomy of a Large-Scale Hypertextual Web Search Engine, Computer Science Department, Stanford University, Stanford, 1998 m., pp. 1-6.

[BPM+98] Sergey Brin, Lawrence Page, Rajeev Motwani, Terry Winograd, The PageRank Citation Rankin: Bringing Order to the web, Computer Science Department, Stanford University, Stanford, 1998 m., pp. 1-6.

[BCH+06] Catherine Benincasa, Adena Calden, Emily Hanlon, Matthew Kindzerske, Kody Law, Eddery Lam, John Rhoades, Ishani Roy, Michael Satz, Eric Valentine and Nathaniel Whitaker, Page Rank Algorithm, Department of Mathematics and Statistics University of Massachusetts, Amherst, 2006 m., pp. 1-7.

[JUO07] Algimantas Juozapavičius, Duomenų struktūros ir efektyvūs algoritmai, Leidykla TEV, 2007 m, pp. 198-203.

[LOW04] David G. Lowe, Distinctive Image Features from Scale-Invariant Keypoints (SIFT), Computer Science Department University of British Columbia Vancouver, B.C., Canada, 2004 m., pp. 1-8 ir 14-16.

[LOW99] David G. Lowe, Object Recognition from Local Scale-Invariant Features, Computer

Science Department University of British Columbia Vancouver, B.C., Canada, 1999 m., pp. 1-4

[KZN08] Neeraj Kumar, Li Zhang, Shree Nayar,What is a Good Nearest Neighbors Algorithm for Finding Similar Patches in Images?, Columbia University, University of Wisconsin-Madison, 2008 m., pp. 4-10.