Sveučilište u Zagrebu
Prirodoslovno-matematički fakultet

Ivan Stojić

Vizualizacija kompleksnih mreža u kontekstu sljedivosti evolucije informacija u razvoju proizvoda

Zagreb, 2012.

Ovaj rad izrađen je na Matematičkom odsjeku Prirodoslovno–matematičkog fakulteta u Zagrebu, pod mentorstvom doc. dr. sc. Maria Štorge s Fakulteta strojarstva i brodogradnje i komentorstvom prof. dr. sc. Saše Singera s Matematičkog odsjeka Prirodoslovno–matematičkog fakulteta, te je predan na natječaj za dodjelu Rektorove nagrade u akademskoj godini 2011./2012.

Sadržaj

1 Uvod

Riječ *informacija* ima mnogo različitih značenja. Neka od najvažnijih otkrića u povijesti čovjeka vezana su uz načine bilježenja, prenošenja i transformiranja informacija. U modernije doba, mnoge tehnologije razvijene su isključivo u svrhu manipuliranja informacijama, a u mnogim granama ljudske djelatnosti informacija se pojavljuje kao centralni koncept.

1.1 Sljedivost evolucije informacija u razvoju proizvoda

U procesu razvoja proizvoda, informacija je jedan od glavnih resursa i krajnji rezultat. Oblici u kojima se ona pojavljuje u ovom kontekstu su razni tekstualni i grafički dokumenti. Razvoj proizvoda može se promatrati kao iterativni proces transformacije inženjerskih informacija [10], s ciljem konačnog definiranja karakteristika i svojstava proizvoda (oblik, materijal, proizvodni postupak, zadovoljavanje zakonske regulative, utjecaj na okoliš, itd).

Sljedivost evolucije informacija u razvoju proizvoda može se definirati kao faktor kvalitete procesa razvoja prozvoda [15]. Cilj međunarodnog EUREKA projekta TRENIN (Traceability of Engineering Information, `http://www.trenin.org/`) bio je stvaranje okvira za praćenje inženjerskih informacija tokom razvoja proizvoda. Ovaj projekt rezultirao je razvojem ontologije sljedivosti [16], temeljem koje je definiran zapis sljedivosti (TR: Traceability Record) kao mreža elemenata i objekata sljedivosti (TE: Traceability Element, TO: Traceability Object) povezanih semantičkim vezama različitih vrsta i jačina [14, 17, 18].

Pri razvoju kompleksnih proizvoda, broj instanci elemenata ontologije sljedivosti može biti velik, a njihova međusobna povezanost može proizvesti

semantičku mrežu s vrlo kompleksnom strukturom. Alat za vizualizaciju opisan u ovom radu razvijen je s ciljem olakšavanja rada s rezultirajućom kompleksnom mrežom, upotpunjujući na taj način korištenje zapisa sljedivosti informacija u razvoju proizvoda nakon njegove pohrane.

Zbog fleksibilnosti razvijene vizualizacije, alat je moguće primijeniti i na mreže koje opisuju semantičku kompleksnost informacija u drugim područjima, kao što je vidljivo iz jednog od testnih slučajeva u ovom radu — vizualizacije znanstvene evolucije područja Znanosti o konstruiranju.

2 Ciljevi

Osnovni ciljevi rada su vizualizacija kompleksnih mreža nastalih temeljem ontologije sljedivosti informacija, prikaz dinamike evolucije informacija u razvoju proizvoda, te otkrivanje i vizualno prikazivanje bitnih značajki struktura promatranih mreža.

2.1 Vizualizacija evolucije informacija

Vizualizacija evolucije informacija koja je predmet ovog rada je zamišljena kao interaktivan, organski, prikaz kompleksnih mreža koji je dovoljno adaptivan da može prikazati evoluciju i rast kompleksnih ontoloških mreža kroz vrijeme, kako bi pružio dobro razumijevanje strukture mreže, te dinamike i svojstava njenog rasta. Organska vizualizacija informacija je metoda koja koristi simulirana svojstva organskih struktura za interaktivno vizualno sučelje kako bi pružila uvid u kvalitativne značajke koje proizlaze iz kvantitativnih podataka generiranih dinamikom razvoja informacije [8].

Neka od organskih svojstava vizualizacije realizirane u ovom radu su:

- struktura — iz agregacije elemenata nastaju kompleksne strukture,
- metabolizam — procesiranje inženjerske informacije je grafički prikazano na odgovarajućem objektu ontološke mreže,
- rast — porast broja elemenata i reorganizacija strukture mreže prati evoluciju ontološke mreže kroz vrijeme,
- homeostaza — održavanje uravnoteženog stanja raspoređivanjem elemenata mreže ovisno o promjeni strukture i vanjskim podražajima,
- responzivnost — reakcija na podražaje i promjene u okolini,

- kretanje — elementi mreže se kreću prema stabilnom rasporedu,
- reprodukcija — novi elementi ontološke mreže pozicioniraju se ovisno o ontologiji sljedivosti i postojećoj strukturi mreže u vizualizaciji.

Za realizaciju vizualizacije s ovakvim svojstvima, bilo je potrebno odabrati algoritam za raspoređivanje elemenata mreže u ravnini. Algoritam za raspoređivanje vođen silom (force directed layout algorithm) izabran je zbog svoje adaptivnosti i fluidnosti. Uz ove bitne karakteristike, algoritam za velike mreže rezultira estetski prihvatljivim rasporedom s visokom razinom simetrije i ravnomjerno distribuiranim elementima u prostoru, a moguće je i utjecati na konačni raspored modifikacijom jačina pojedinih sila te tako prilagoditi prikaz potrebama [3].

Također, zbog toga što se odabrani algoritam zasniva na osnovnim elementima dinamike kao što su sila koja opada s kvadratom udaljenosti i sila opruge, dinamičko ponašanje vizualizirane mreže trebalo bi biti blisko intuiciji korisnika vizualizacije, oslanjajući se na iskustvo s fizikalnim principima koje je korisnik stekao kroz izloženost fizikalnoj stvarnosti kojom upravljaju upravo ovakve sile.

2.2 Olakšavanje rada s kompleksnim mrežama

Vizualna reprezentacija kompleksne mreže, posebno ako ističe njezine osnovne karakteristike, vrlo je koristan alat pri razumijevanju strukture i svojstava mreže; mnogi sustavi za obradu i prikaz informacija uspješno koriste vizualizaciju u prezentaciji kompleksnih informacijskih struktura. Prije razvoja računala, mogućnosti vizualiziranja bile su znatno ograničene, pogotovo pri radu s kompleksnim mrežama koje se sastoje od velikog broja elemenata.

Kompleksnost ontoloških mreža koje se pojavljuju u kontekstu razvoja proizvoda reflektira kompleksnost samih proizvoda, koja je posljednjih desetljeća u porastu, s naznakama da će se ovaj trend nastaviti, pa i ubrzati. Pojavljuje se sve veća potreba za upravljanjem ovom kompleksnošću, u cilju efikasnijeg razvoja proizvoda, povećanja raspona varijanti proizvoda, te uspješnijeg pokrivanja što šireg segmenta tržišta [11].

Jedan od ciljeva ovog rada je olakšavanje rada s kompleksnim ontološkim mrežama kroz grafičko sučelje koje bitno unaprjeđuje njihovo proučavanje i razumijevanje u odnosu na tablični prikaz njihovih zapisa.

2.3 Otkrivanje strukture kompleksnih mreža

Osim novih mogućnosti vizualiziranja složenih struktura, razvoj računala rezultirao je i novom dinamikom u području proučavanja strukture kompleksnih mreža. U prošlosti, uz matematičku teoriju grafova, proučavanjem mreža posebno su se bavile i društvene znanosti. U odnosu na ove mreže, koje su se sastojale od najviše nekoliko desetaka ili stotina elemenata, razvoj računala omogućio je uvid u statistička svojstva i strukturu vrlo velikih mreža s milijunima ili čak milijardama elemenata, otvorivši potpuno novi pristup proučavanju kompleksnih mreža [12].

Bitan cilj vizualizacije je i rasvijetljavanje strukture kompleksnih mreža, posebno identifikacija jako povezanih grozdova elemenata uloženih u mrežu, te računanje s tim usko povezane mjere modularnosti mreže. U ovu svrhu odabran je Newmanov brzi algoritam za detekciju grozdova [13] kako bi analiza strukture glatko upotpunila vizualizaciju evolucije ontološke mreže.

3 Modeliranje kompleksnih mreža

Kompleksne mreže pojavljuju se i u mnogim drugim kontekstima, kao što su World Wide Web, hranidbeni lanci, mreže neurona, razne distribucijske mreže i slično. Eulerovo rješenje problema Königsberških mostova 1735. godine smatra se početkom teorije grafova i formalnog proučavanja mreža [12].

Matematička teorija grafova služi kao teoretska podloga za modeliranje i proučavanje kompleksnih mreža; elementi mreže predstavljeni su vrhovima, a veze u mreži bridovima grafa. U nastavku poglavlja slijede definicije nekih osnovnih pojmova iz teorije grafova i konvencije korištene u daljnjem tekstu.

3.1 Osnovni pojmovi i konvencije

Definicija 1. ***Graf*** *je uređeni par $G = (V, E)$ skupa V čije elemente nazivamo* ***vrhovi*** *i familije $E \subseteq \mathcal{P}(V)$ dvočlanih podskupova od V, čije elemente nazivamo* ***bridovi****. Graf također nazivamo i* ***neusmjereni graf****.*

Definicija 2. ***Usmjereni graf*** *je uređeni par $G = (V, E)$ skupa V čije elemente nazivamo* ***vrhovi*** *i skupa $E \subseteq V \times V$ uređenih parova elemenata iz V, čije elemente nazivamo* ***bridovi****.* ***Pripadni graf*** *usmjerenog grafa G je neusmjereni graf nastao zanemarivanjem orijentacije svih bridova iz G.*

Definicija 3. ***Neusmjereni (usmjereni) multigraf*** *je neusmjereni (usmjereni) graf čiji bridovi čine multiskup.*

Definicija 4. *Kažemo da su vrh v i brid e grafa* ***incidentni*** *ako je $v \in e$.* ***Stupanj vrha*** *v, u oznaci $k(v)$ je broj bridova s kojima je taj vrh incidentan.*

Vrhovi v i w su ***susjedni*** *ako su incidentni s nekim zajedničkim bridom. Analogno se definiraju ovi pojmovi za vrhove i bridove usmjerenog grafa.*

Sa z označavamo prosječan stupanj vrha u grafu:

$$z = \frac{1}{|V|} \sum_{v \in V} k(v).$$

Definicija 5. *Neka je $G = (V, E)$, $V = \{v_1, v_2, \ldots, v_n\}$, $n \in \mathbb{N}$, (usmjereni) (multi)graf.* ***Matrica susjedstva*** *od G je matrica $\boldsymbol{A} \in \mathbb{M}^{n \times n}$ čiji je element na mjestu (i, j), $\boldsymbol{A}_{ij}$ jednak broju zajedničkih bridova s kojima su v_i i v_j incidentni, za sve $i, j \in \{1, \ldots, n\}$.*

Za vrhove grafa $v = v_i$ i $w = v_j$, element $\boldsymbol{A}_{ij}$ matrice susjedstva označavamo i sa $\boldsymbol{A}_{vw}$.

Definicija 6. *Kažemo da je graf $G' = (V', E')$* ***podgraf*** *grafa G ako je $V' \subseteq V$ i $E' \subseteq E$.*

Definicija 7. ***Graf generiran skupom vrhova*** *$V' \subseteq V$ je maksimalni (u smislu relacije* biti podgraf*) podgraf od G takav da mu je skup vrhova jednak V'.*

Definicija 8. ***Staza*** *u (usmjerenom) (multi)grafu je alternirajući niz vrhova i bridova $v_1, e_1, v_2, e_2, \ldots, e_{k-1}, v_k$ takav da su za sve $i \in 1, 2, \ldots, k-1$ vrhovi v_i i v_{i+1} incidentni s bridom e_i, te nema ponavljanja bridova u nizu.* ***Duljina staze*** *je broj bridova u stazi.*

Put *u (usmjerenom) (multi)grafu je staza u kojoj nema ponavljanja vrhova, osim eventualno na početku i kraju. Put u kojem su prvi i posljednji vrh jednaki zovemo* ***ciklus****.*

Definicija 9. *Kažemo da je (usmjereni) (multi)graf* ***povezan*** *ukoliko postoji put između svaka dva vrha toga grafa, a inače kažemo da je* ***nepovezan****.*

Komponenta povezanosti *(usmjerenog) (multi)grafa je maksimalni* (*u smislu relacije* biti podgraf) *povezani podgraf grafa.*

Kompleksna mreža je u ovom radu modelirana kao usmjereni multigraf jer je pomoću ove strukture moguće obuhvatiti sve prethodno definirane vrste grafova — zanemarivanjem orijentacija bridova obuhvaćeni su neusmjereni multigrafovi, a zabranom višestrukih bridova između dva vrha grafovi. U implementaciji je izostavljena mogućnost veze vrha grafa sa samim sobom jer se za tim nije ukazala potreba.

U daljnjem tekstu se riječ "graf" koristi za sve ove vrste grafova, ako nije specifirano o kakvom se grafu radi. Broj vrhova grafa $G = (V, E)$ označen je s $n = |V|$, a broj bridova s $m = |E|$, s time da se za broj vrhova koristi i oznaka $|G|$, a za pripadnost vrha grafu, $v \in V$ i oznaka $v \in G$.

U kontekstu kompleksnih mreža važno je svojstvo i prosječna udaljenost između vrhova grafa.

Definicija 10. *Označimo sa d_{ij} duljinu najkraćeg puta između vrhova v_i i v_j (smatramo da je $d_{ij} = +\infty$ ako ne postoji takav put). Definiramo l kao aritmetičku sredinu duljina najkraćih puteva:*

$$l = \frac{1}{n(n-1)} \sum_{i \neq j} d_{ij}.$$

U slučaju kad ova vrijednost nije konačna (primjerice, kod nepovezanih grafova), korisno je definirati sličnu vrijednost. Definiramo l' kao harmonijsku

sredinu duljina najkraćih puteva:

$$l' = \left(\frac{1}{n(n-1)} \sum_{i \neq j} d_{ij}^{-1} \right)^{-1}.$$

Pokazuje se da je u većini mreža koje se pojavljuju u stvarnosti prosječna udaljenost elemenata mreže relativno mala u odnosu na broj vrhova mreže — takozvani small-world effect. Ovaj naziv nastao je zbog poznatih eksperimenata Stanleya Milgrama u kojima je slanjem pisama putem poznanika nastojao izmjeriti prosječnu udaljenost ljudi u društvenoj mreži. Za dva proizvoljno odabrana vrha mreže može se očekivati da je potrebno samo nekoliko koraka kako bi se došlo od jednog do drugog vrha. Vrijednost l za stvarne mreže je tipično manja od 10, čak i za vrlo velike mreže od nekoliko stotina tisuća elemenata [12]. Ovo ima snažne implikacije za mnoge kompleksne mreže, posebno pri proučavanju toka informacija u informacijskim i društvenim mrežama.

Definicija 11. ***Stablo*** *T je povezani neusmjereni graf u kojem nema ciklusa. Stablo se sastoji od jednog istaknutog vrha R kojeg nazivamo* ***korijen stabla****, i 0 ili više podgrafova $T_1, T_2, \ldots, T_k$ povezanih s korijenom koji su i sami stabla. Pritom, vrh $R_i \in T_i$ koji je susjedan korijenu R stabla T je korijen stabla T_i, te za R kažemo da je* ***roditelj*** *od R_i, a za R_i da je* ***dijete*** *od R. Vrhove stabla također nazivamo i* ***čvorovi****.*

Čvorove stabla koji imaju dijete nazivamo ***unutrašnji čvorovi****, a ostale čvorove nazivamo* ***listovi****.*

U daljnjem tekstu se za brojeve, te vrhove i bridove grafa koriste mala slova latiničnog i grčkog pisma (n, k, v, e, θ), uz iznimku mjere modularnosti

Q (koja je definirana kasnije). Za grafove, skupove, stabla i čvorove stabla koriste se velika slova latiničnog pisma (G, V, T, R), a za vektore i matrice masno otisnuta slova ($\mathbf{c}$, $\mathbf{P}$).

4 Metode i algoritmi

4.1 Algoritam za raspoređivanje vrhova grafa vođen silom

Graf se prikazuje u ravnini, te je svakom vrhu grafa v_i pridružen vektor položaja $\mathbf{P}_i$ i vektor brzine $\mathbf{V}_i$.

Za prikaz grafa korišten je algoritam za raspoređivanje vrhova vođen silom [5]. U implementiranoj verziji algoritma vrhovi se tretiraju kao čestice istovrsnih naboja, te se odbijaju silom koja opada s kvadratom udaljenosti, pri čemu se kao količina naboja pridružena pojedinom vrhu uzima stupanj vrha uvećan za 1. Ova vrijednost također služi i kao masa pridružena vrhu. Bridovi se tretiraju kao opruge koje povezuju susjedne vrhove.

Između svaka dva vrha v i w djeluje odbojna sila intenziteta

$$f_r(v, w) = \frac{k_r(k(v) + 1)(k(w) + 1)}{\text{dist}(v, w)^2},$$

gdje je $k_r > 0$ konstanta jakosti odbojne sile, a $\text{dist}(v, w)$ euklidska udaljenost vrhova v i w. Između susjednih vrhova djeluje i sila opruge intenziteta

$$f_a(v, w) = k_a \ (\text{dist}(v, w) - l_e),$$

gdje je $k_a > 0$ konstanta opruge, a $l_e > 0$ duljina brida pri kojoj je pripadna opruga u ekvilibriju. Ova sila može djelovati kao privlačna ili kao odbojna sila, ovisno o predznaku izraza u zagradi.

Bez prigušenja, odbojne sile i sile opruga uzrokovale bi beskonačno titranje vrhova grafa. Zato se nakon svakog vremenskog koraka brzine vrhova

korigiraju prema formuli

$$\mathbf{V}_i' = (1 - k_d\ t)\ \mathbf{V}_i,$$

gdje je $k_d > 0$ konstanta prigušenja, a t trajanje u sekundama vremenskog koraka, pri čemu su ove vrijednosti odabrane tako da je $k_d\ t < 1$.

Kako ovaj algoritam nastoji maksimalno razmaknuti vrhove grafa (koliko to bridovi-opruge dopuštaju), u slučaju nepovezanog grafa on rezultira neograničenim udaljavanjem komponenata povezanosti grafa. Da bi se ovo spriječilo, algoritam se koristi samo unutar komponenata povezanosti.

Različite komponente povezanosti međusobno se odbijaju ukoliko su im centri na udaljenosti manjoj od $\frac{3}{2}$ zbroja njihovih radijusa, pri čemu je centar komponente povezanosti K definiran sa

$$\mathbf{c}(K) = \frac{1}{|K|} \sum_{v_i \in K} \mathbf{P}_i,$$

a radijus sa

$$r(K) = \frac{1}{2} \max_{v,w \in K} \mathrm{dist}(v, w).$$

Intenzitet odbojne sile između dvije komponente povezanosti K_i i K_j je

$$f(K_i, K_j) = \frac{k_r^K\ |K_i|\ |K_j|}{\mathrm{dist}(\mathbf{c}(K_i), \mathbf{c}(K_j))^2},$$

gdje je $k_r^K > 0$ konstanta odbojne sile za komponente povezanosti.

Konstante k_r, k_a, l_e, k_d i k_r^K pružaju visok stupanj kontrole nad geometrijom i dinamikom grafa. One se mogu odrediti empirijski, pri čemu su estetika rezultirajućeg rasporeda vrhova i prirodnost dinamike grafa glavni kriteriji

za njihov odabir, a moguće je i automatski ih odrediti pomoću osnovnih karakteristika grafa [5].

Osnovni koraci algoritma s navedenim modifikacijama prikazani su pseudokodom:

Algoritam FORCE_DIRECT($G, \mathbf{P}, \mathbf{V}, t$)
{ G je graf, $\mathbf{P}$ matrica s vektorima položaja vrhova grafa, $\mathbf{V}$ matrica s vektorima brzina vrhova grafa, a t trajanje vremenskog koraka. }
{ Računanje sila. }
Inicijaliziraj matricu sila na vrhove $\mathbf{F}$ nul-vektorima.
Inicijaliziraj matricu sila na komponente povezanosti $\mathbf{F}^K$ nul-vektorima.
Odredi komponente povezanosti $K_1, \ldots, K_k$ grafa G.
Za svaku komponentu povezanosti K_i:
    Izračunaj centar $\mathbf{c}(K_i)$ i radijus $r(K_i)$.
    Izračunaj odbojne sile među vrhovima u K_i i ažuriraj $\mathbf{F}$.
    Izračunaj sile opruga među vrhovima u K_i i ažuriraj $\mathbf{F}$.
Za svake dvije komponente povezanosti K_i, K_j takve da je $i < j$:
    Ako je $\text{dist}(\mathbf{c}(K_i),\ \mathbf{c}(K_j)) < \frac{3}{2}(r(K_i) + r(K_j))$:
        Izračunaj odbojnu silu između K_i i K_j i ažuriraj $\mathbf{F}^K$.
{ Računanje brzina i položaja vrhova. }
Za svaki vrh v_i grafa G:
    S K_j označimo komponentu povezanosti koja sadrži v_i.
    $\mathbf{V}_i \leftarrow (1 - k_d\ t)\ \mathbf{V}_i + \frac{t}{k(v_i)+1}\left(\mathbf{F}_i + \frac{\mathbf{F}_j^K}{|K_j|}\right)$
    $\mathbf{P}_i \leftarrow \mathbf{P}_i + t\ \mathbf{V}_i$

Kako bi se vizualizacija mogla izvoditi u realnom vremenu za veće grafove, potrebno je odabrati brze algoritme. Računski najzahtjevniji dio gornjeg algoritma je računanje odbojnih sila između svih parova vrhova pojedine komponente povezanosti grafa — problem n tijela. Ovaj problem i njemu

srodni problemi pojavljuju se u mnogim kontekstima u matematičkom modeliranju fizikalnih sustava, primjerice u simulaciji nastajanja planetarnih sustava i njihove dinamike, modeliranju sudara galaksija, razvoja strukture svemira, modeliranju ponašanja plazme i slično.

Naivni algoritam za rješenje problema n tijela je složenosti $O(n^2)$ koja, uz potreban vremenski korak vizualizacije u trajanju od nekoliko stotinki sekunde (animacija s 25 sličica po sekundi odgovara vremenskom koraku od 4 stotinke sekunde), ne dopušta rad s grafovima s više od nekoliko stotina vrhova. Na sreću, zbog česte pojave problema n tijela, razvijeni su mnogi brzi algoritmi za njegovo rješavanje. U ovom radu korišten je Barnes–Hutov algoritam, složenosti $O(n \log(n))$ [2].

4.2 Barnes–Hutov algoritam za brzo sumiranje sila

Barnes–Hutov algoritam služi za brzo sumiranje sila u problemu n tijela. Algoritam je ovdje opisan u kontekstu računanja sila između vrhova komponente povezanosti K.

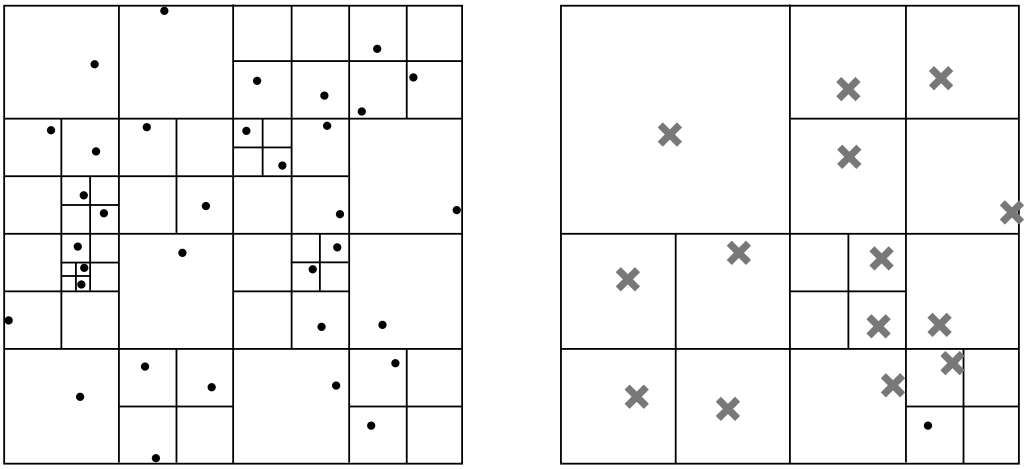
Računski zahtjevno računanje sila između svih parova vrhova izbjegnuto je podjelom dijela ravnine kojeg zauzimaju vrhovi iz K u stablastu strukturu, te korištenjem ove strukture kako bi se brzo izračunala sila na pojedini vrh. Prva faza algoritma je izgradnja stabla u kojem pojedini čvor predstavlja dio ravnine (točnije, kvadrat u ravnini) te sadrži informacije o nabojima koji se nalaze unutar tog dijela ravnine. Svaki čvor $N^{(j)}$ stabla označen je uređenom četvorkom $(A^{(j)}, V^{(j)}, \mathbf{c}_q^{(j)}, q^{(j)})$, gdje je $A^{(j)}$ kvadrat, $V^{(j)}$ skup vrhova unutar tog kvadrata, $\mathbf{c}_q^{(j)}$ centar naboja vrhova iz $V^{(j)}$, a $q^{(j)}$ ukupan naboj vrhova iz $V^{(j)}$.

Pri tome je kvadrat iz oznake korijena stabla najmanji kvadrat koji pokriva sve vrhove komponente povezanosti K. Svaki unutrašnji čvor stabla ima točno četvero djece, te su kvadrati iz oznaka djece dobiveni podjelom kvadrata iz oznake roditelja na četiri jednaka kvadrata. Na kraju, za svaki list $N^{(j)}$ stabla vrijedi da je $|V^{(j)}| \leq b$, gdje je $b \in \mathbb{N}$ parametar algoritma.

Izgradnja stabla počinje stablom koje se sastoji samo od korijena označenog s $(A, \emptyset, \mathbf{0}, 0)$, gdje je A najmanji kvadrat koji pokriva sve vrhove iz komponente povezanosti K. Zatim se jedan po jedan vrh ubacuje u stablo:

Algoritam BUILD_TREE(K)
Pronađi najmanji kvadrat A koji pokriva K.
$T \leftarrow$ stablo s korijenom R označenim s $(A, \emptyset, \mathbf{0}, 0)$
Za svaki vrh $v \in K$:
    INSERT_VERTEX(v, R)
Vrati T.

Algoritam INSERT_VERTEX(v_i, $N^{(j)}$)
$\mathbf{c}_q^{(j)} \leftarrow \left(q^{(j)} + k(v_i) + 1\right)^{-1} \left(q^{(j)}\mathbf{c}_q^{(j)} + (k(v_i)+1)\,\mathbf{P}_i\right)$
$q^{(j)} \leftarrow q^{(j)} + k(v_i) + 1$
$V^{(j)} \leftarrow V^{(j)} \cup \{v_i\}$
Ako je $N^{(j)}$ unutrašnji čvor stabla:
    Za dijete $N^{(k)}$ od $N^{(j)}$ takvo da $A^{(k)}$ sadrži v_i:
        INSERT_VERTEX(v_i, $N^{(k)}$)
Inače:
    Ako je $|V^{(j)}| > b$:
        Podijeli $A^{(j)}$ na četiri jednaka kvadrata $A_1, \ldots, A_4$.
        Za $k = 1$ do 4:
            Dodaj dijete N_k označeno s $(A_k, \emptyset, \mathbf{0}, 0)$ u čvor $N^{(j)}$.
            Za svaki vrh $v \in V^{(j)}$ kojeg A_k sadrži:
                INSERT_VERTEX(v, N_k)



Slika 1: Lijevo: izgrađeno stablo, desno: računanje sile na jedan vrh.

Nakon što je stablo izgrađeno, računanje ukupne sile na pojedini vrh v_i može se efikasno izvesti silaskom niz stablo, počevši od korijena. Neka je $N^{(j)}$ promatrani čvor stabla u ovom postupku. Ako je $N^{(j)}$ dovoljno udaljen od v_i, sila na v_i uzrokovana od vrhova u $V^{(j)}$ se aproksimira silom koju bi uzrokovao vrh s nabojem $q^{(j)}$ i pozicijom $\mathbf{c}_q^{(j)}$. Ako $N^{(j)}$ nije dovoljno udaljen, rekurzivno se obilaze njegova djeca (osim ako je $N^{(j)}$ list stabla — tada se zasebno računaju sile na v_i za svaki vrh iz $V^{(j)} \setminus \{v_i\}$).

Pritom, čvor $N^{(j)}$ je *dovoljno udaljen* od v_i ako je

$$\mathrm{dist}(\mathbf{c}_q^{(j)}, \mathbf{P}_i) > \frac{a^{(j)}}{\theta} \ ,$$

gdje je $a^{(j)}$ duljina stranice kvadrata $A^{(j)}$, a $\theta \in \langle 0, 1]$ parametar algoritma. Ovim postupkom se mnoge skupine naboja aproksimiraju pripadnim centrima naboja, s time da su ove skupine veće što su udaljenije, kao što je prikazano na slici 1 desno, gdje su križićima predstavljeni centri naboja korišteni za računanje sile na vrh koji se nalazi u donjem desnom dijelu slike.

Algoritam FORCE_VERTEX(v_i, $N^{(j)}$)

Ako je $N^{(j)}$ dovoljno udaljen od v_i:

Ažuriraj $\mathbf{F}_i$ silom uzrokovanom nabojem $q^{(j)}$ s pozicijom $\mathbf{c}_q^{(j)}$.

Inače:

Ako je $N^{(j)}$ list:

Za svaki $w \in V^{(j)} \setminus \{v_i\}$:

Ažuriraj $\mathbf{F}_i$ silom uzrokovanom vrhom w.

Inače:

Za svako dijete N_k od $N^{(j)}$:

FORCE_VERTEX(v_i, N_k)

Pomoću parametra θ moguće je podesiti kompromis između preciznosti simulacije i brzine izračunavanja. Kako se u okviru ovog rada Barnes–Hutov algoritam koristi samo pri računanju rasporeda vrhova grafa, a ne u okviru fizikalne simulacije kretanja tijela, koristan raspon vrijednosti parametra θ znatno je veći nego u primjenama koje zahtijevaju veću preciznost simulacije. Fizikalna točnost u simulaciji u svrhu vizualizacije zauzima daleko manje bitno mjesto, te je izbor parametra θ ograničen samo stabilnošću simulacije (pri vrlo agresivnim vrijednostima ovog parametra i maloj točnosti dolazi do nepoželjnih efekata u kretanju vrhova grafa).

Kako se računanje sila na pojedine vrhove izvodi potpuno neovisno, ovaj dio algoritma moguće je efikasno paralelizirati i time bitno ubrzati izvođenje algoritma, što je i iskorišteno u implementaciji alata za vizualizaciju opisanog u ovom radu.

4.3 Vizualizacija evolucije grafa

Vizualizacija evolucije grafa omogućena je kontrolom pomoću koje korisnik može odabrati željeni vremenski trenutak prikaza mreže. Prilikom promjene vremenskog trenutka, vrhovi i bridovi grafa se pojavljuju ili nestaju, pri čemu se vizualizirani graf, vođen silama, prilagođava novoj strukturi u potrazi za stabilnim stanjem.

Konačna geometrija kompletnog grafa računa se prije prvog prikazivanja djelomičnog grafa, te su time određeni položaji vrhova koji se pojavljuju kada korisnik odabere neki kasniji vremenski trenutak u vizualizaciji. Ovime je postignuto pojavljivanje novih vrhova blizu vrhova s kojima su oni povezani, što rezultira poboljšanom fluidnošću prikaza evolucije grafa.

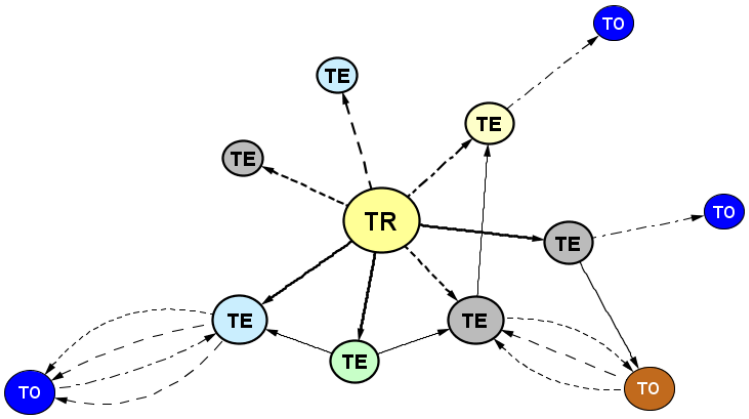
Ovo inicijalno računanje geometrije kompletnog grafa također se vrši prethodno opisanim algoritmom, s tom razlikom da se konstanta prigušenja k_d na početku postavlja na malu vrijednost, te polako raste prema konačnoj vrijednosti. Ovaj postupak sličan je simuliranom kaljenju: na početku je moguća brza promjena stanja grafa zbog niske razine prigušenja, a tokom računanja povećavanje konstante prigušenja postupno "zamrzava" graf u konačnom položaju.

4.4 Crtanje grafa

Uz vizualiziranje strukture ontološke mreže kroz raspored vrhova grafa u ravnini, korištene su i boje vrhova za označavanje različitih klasa elemenata ontološke mreže, veličine vrhova ovisne o stupnju vrha, i različiti tipovi linija za različite klase relacija. Tekstualne oznake unutar vrhova označavaju klasu elementa ontologije sljedivosti kojeg je vrh instanca.

Jednostruki bridovi prikazuju se kao ravne linije u grafu, a višestruki bridovi kao zakrivljene linije, pomoću Bézierovih krivulja drugog stupnja [4], dok je usmjerenost brida prikazana strelicom na odgovarajućem kraju brida.

Pojavljivanje novih vrhova i bridova popraćeno je animacijama, a procesiranje inženjerske informacije prikazano je pomoću pulsirajućeg animiranog efekta na odgovarajućem vrhu grafa. Tokom rasta mreže, rast vrhova u skladu s rastom njihovih stupnjeva i animacija bridova pri pojavi višestrukih bridova između vrhova, realizirana manipulacijom kontrolnih točaka Bézierovih krivulja, dodano pospješuju organsku vizualizaciju mreže.



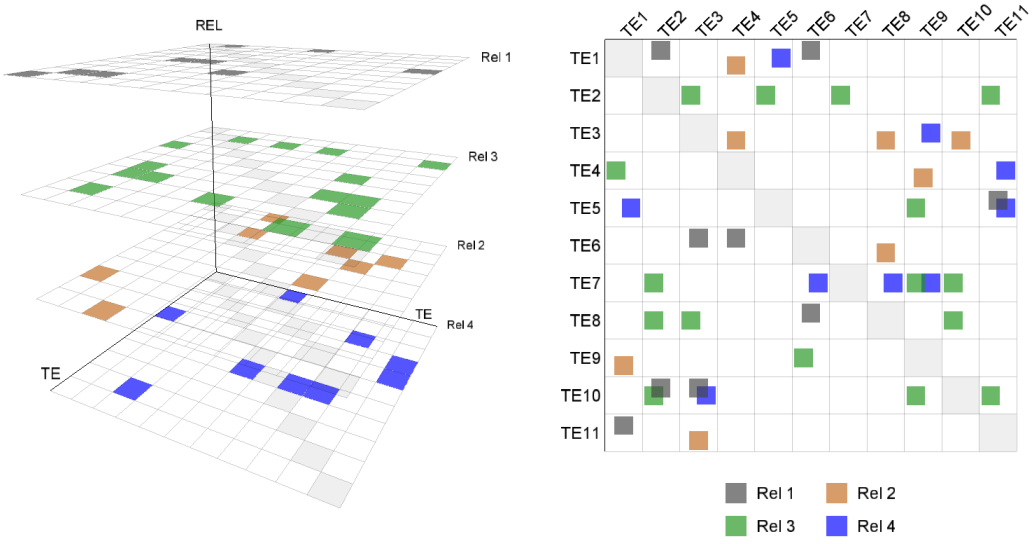
Slika 2: Prikaz vrhova i bridova mreže u razvijenom alatu

Zbog lakšeg snalaženja pri proučavanju kompleksnih mreža implementirana je i mogućnost filtriranja elemenata i relacija prema klasi kojoj oni pripadaju. Odgovarajući vrhovi i bridovi koji su filtrirani prikazuju se transparentno, čime su istaknuti željeni elementi i relacije mreže a istovremeno struktura mreže ostaje vidljiva. Kontrola za mijenjanje veličine prikaza grafa i translatiranje prikaza pomoću povlačenja pozadine mišem dodatno olakšavaju navigaciju kompleksnom mrežom.

4.5 Prikaz klasa relacija u ontologiji sljedivosti

U ontologiji sljedivosti, od posebne su važnosti elementi sljedivosti (Traceability Elements) i semantičke relacije među njima. Istovremena vizualizacija matričnih prikaza ovih relacija u trodimenzionalnom prostoru pruža alternativni način prikaza ovog aspekta strukture ontološke mreže.

Matrice relacija grafički su prikazani na nivoima kao binarne matrice u kojima je prisutnošću boje naznačena prisutnost jedinica na odgovarajućim mjestima u matrici, a moguće je promatrati i agregaciju ovih relacija u jednoj matrici. Pomakom gledišta i istovremenim glatkim prijelazom iz perspektivne u ortografsku projekciju na intuitivan način je naznačena agregacija matričnih prikaza relacija.



Slika 3: Alternativni prikaz relacija

4.6 Pronalaženje strukture grozdova mreže

Kažemo da kompleksna mreža posjeduje strukturu grozdova (ili modula) ako postoje gusto povezani podskupovi elemenata mreže, sa znatno manjom gustoćom veza između različitih podskupova. Pojava strukture grozdova u mreži ukazuje na postojanje podjele elemenata mreže na disjunktne podskupove. Za elemente u istom podskupu postoji veća vjerojatnost da, osim pripadnosti istom grozdu, imaju i neke druge zajedničke karakteristike, koje su i rezultirale njihovom većom povezanošću. Zbog toga je identifikacija strukture grozdova bitna metoda pri proučavanju strukture kompleksne mreže.

Struktura grozdova je pronađena u mnogim kompleksnim mrežama, u kojima ona reflektira stvarnu strukturu sustava koje kompleksna mreža predstavlja. Primjerice, u kompleksnoj mreži World Wide Weba grozdovi odgovaraju tematskim cjelinama, u društvenim mrežama skupovima blisko povezanih ljudi, dok u biokemijskim i neuronskim mrežama struktura grozdova može predstavljati podjelu na funkcionalne grupe i time pridonijeti analizi funkcioniranja promatranog sustava [7].

Iako se analiza grozdova najviše primjenjuje kod običnih "neontoloških" mreža, u kontekstu ontoloških mreža koje se pojavljuju u procesu razvoja proizvoda struktura grozdova mreže može se koristiti pri identifikaciji različitih konteksta razvoja inženjerskih informacija i njihove međusobne povezanosti i međuzavisnosti. Razumijevanje ove strukture i dinamike njene promjene kroz vrijeme pruža uvid u strukturu procesa razvoja proizvoda i evoluciju inženjerskih informacija, te se može iskoristiti u svrhu optimizacije samog procesa razvoja proizvoda.

Od mnogih algoritama za identifikaciju strukture grozdova odabran je brzi

algoritam za detekciju strukture grozdova [13]. Pregled algoritama za rješavanje ovog problema može se vidjeti u [7], a posebno treba istaknuti vrlo uspješan algoritam od Guimerà i suradnika temeljen na metodi simuliranog kaljenja, složenosti ovisne o parametrima algoritma, i algoritam od Ducha i Arenasa složenosti $O(n^2 \log(n))$. Odabrani algoritam je, za grafove u kojima je broj bridova proporcionalan broju vrhova, te koji posjeduju hijerarhijsku strukturu grozdova, prosječne složenosti $O(n \log^2(n))$ [6], a u usporedbi s uspješnijim algoritmima (ali veće složenosti), prilično dobro detektira strukturu grozdova [7]. Za primjenu u ovom radu, algoritam je modificiran kako bi radio na multigrafovima.

Algoritam se bazira na mjeri modularnosti. Ova je mjera pridružena podjeli grafa na grozdove, i određuje koliko je značajna struktura grozdova u danoj podjeli. Mjera modularnosti se često koristi u svrhu evaluacije uspješnosti algoritama za identifikaciju strukture grozdova.

Neka je $G = (V, E)$ graf, $|V| = n \in \mathbb{N}$, $|E| = m \in \mathbb{N}$ i $\mathbf{A} \in \mathbb{N}^{n \times n}$ matrica susjedstva grafa G. Nadalje, neka je $\{C_1, \ldots, C_k\}$, $k \in \mathbb{N}$ neka particija skupa vrhova V. Pod pojmom *grozd određen skupom* C_i podrazumijeva se podgraf generiran skupom C_i. Za $v \in V$, označimo sa c_v indeks j takav da je $v \in C_j$.

S e_{ij} označimo udio bridova u grafu koji povezuju C_i i C_j:

$$e_{ij} = \frac{1}{2m} \sum_{v,w \in V} \delta(c_v, i)\delta(c_w, j)\mathbf{A}_{vw} \quad i, j \in \{1, \ldots, k\},$$

a s a_i označimo udio krajeva bridova koji su incidentni s nekim vrhom u C_i:

$$a_i = \frac{1}{2m} \sum_{v \in C_i} k(v), \quad i \in \{1, \ldots, k\}.$$

Definicija 12. *Uz gornje oznake,* ***modularnost*** *(u oznaci Q) pridružena grafu $G = (V, E)$ i pripadnoj particiji skupa vrhova $\{C_1, \ldots, C_k\}$, $k \in \mathbb{N}$ je broj*

$$Q = \sum_{i=1}^{k} \left(e_{ii} - a_i^2\right).$$

U izrazu za Q, e_{ii} je udio bridova unutar grozda određenog s C_i, a a_i^2 je procjena očekivanog udjela bridova unutar istog grozda, ako bi se bridovi u grafu pojavljivali na slučajan način ali uzimajući u obzir stupnjeve vrhova.

Ako se za neku podjelu na grozdove udio bridova unutar grozdova ne razlikuje bitno od udjela koji se može očekivati pri slučajnom postavljanju bridova, vrijednost od Q će biti blizu nule. Naprotiv, ukoliko podjela na grozdove rezultira većim udjelom bridova unutar grozdova, vrijednost od Q bit će veća od nule. U praksi, vrijednosti modularnosti Q veće od 0.3 su indikacija značajne strukture grozdova [6].

Osnovna ideja algoritma je maksimizacija modularnosti iterativnim spajanjem grozdova, počevši od particije skupa vrhova V na jednočlane skupove $P^{(0)} = \{\{v_1\}, \{v_2\}, \ldots, \{v_n\}\}$. Početne vrijednosti od a_i su

$$a_i^{(0)} = \frac{k(v_i)}{2m}, \quad i = 1, \ldots, n,$$

a početna modularnost

$$Q^{(0)} = \sum_{i=1}^{n} \left[0 - \left(a_i^{(0)}\right)^2\right] = -\frac{1}{4m^2} \sum_{i=0}^{n} k(v_i)^2.$$

Algoritam kroz rad održava matricu $\Delta\mathbf{Q}$, u kojoj se na presjeku i-tog retka i j-tog stupca nalazi razlika između modularnosti podjele na grozdove koja bi rezultirala spajanjem i-tog i j-tog skupa trenutne particije skupa vrhova V, i

trenutne modularnosti. Ova matrica je simetrična, ima nule na dijagonali, a početna vrijednost na mjestu $(i, j), i \neq j$ je vrijednost promjene modularnosti koja bi rezultirala spajanjem jednočlanih skupova $\{v_i\}, \{v_j\} \in P^{(0)}$:

$$\Delta\mathbf{Q}_{ij}^{(0)} = e_{ij}^{(0)} - 2a_i^{(0)}a_j^{(0)} = e_{ij}^{(0)} - \frac{k(v_i)k(v_j)}{2m^2}, \quad i, j \in \{1, \ldots, n\}, \quad i \neq j.$$

U svakom koraku spaja se par grozdova za koje je pripadna vrijednost u matrici $\Delta\mathbf{Q}$ pozitivna i maksimalna, te se ažuriraju Q, matrica $\Delta\mathbf{Q}$ i vrijednosti a_i. Ako ne postoji par grozdova čije bi spajanje dovelo do povećanja modularnosti, algoritam završava.

Neka je s trenutni korak algoritma i, bez smanjenja općenitosti, pretpostavimo da se spajaju posljednja dva skupa u

$$P^{(s)} = \left\{C_1^{(s)}, C_2^{(s)}, \ldots, C_{n-s-1}^{(s)}, C_{n-s}^{(s)}\right\},$$

dakle da je $\Delta\mathbf{Q}_{n-s-1,n-s}^{(s)}$ maksimalni element u matrici $\Delta\mathbf{Q}^{(s)}$.

Nakon spajanja $C_{n-s-1}^{(s)}$ i $C_{n-s}^{(s)}$ u uniju $C_{n-s-1}^{(s+1)}$, imamo

$$P^{(s+1)} = \left\{C_1^{(s)}, C_2^{(s)}, \ldots, C_{n-s-2}^{(s)}, C_{n-s-1}^{(s+1)}\right\},$$

dakle prvih $n - s - 2$ skupova u particiji ostalo je nepromijenjeno. Iz ovoga i definicije modularnosti Q lako se vidi da će gornji lijevi kvadratni blok dimenzije $n-s-2$ matrice $\Delta\mathbf{Q}^{(s+1)}$ biti jednak odgovarajućem bloku matrice $\Delta\mathbf{Q}^{(s)}$, preciznije:

$$\Delta\mathbf{Q}_{ij}^{(s+1)} = \Delta\mathbf{Q}_{ij}^{(s)}, \quad i, j = 1, \ldots, n - s - 2.$$

Dakle, potrebno je izračunati samo posljednji redak i stupac matrice $\Delta\mathbf{Q}^{(s+1)}$. Označimo $t := n - s - 1$. Iz definicije modularnosti lako se pokaže da vrijedi:

$$\Delta\mathbf{Q}_{kt}^{(s+1)} = \Delta\mathbf{Q}_{tk}^{(s+1)} = e_{tk}^{(s)} - 2a_t^{(s)}a_k^{(s)} + e_{t+1,k}^{(s)} - 2a_{t+1}^{(s)}a_k^{(s)}, \quad k = 1, \ldots, t-1.$$

Kako za proizvoljni korak algoritma s i $i, j \in \{1, \ldots, n-s\}$, $i \neq j$ vrijedi

$$\Delta\mathbf{Q}_{ij}^{(s)} = e_{ij}^{(s)} - 2a_i^{(s)}a_j^{(s)},$$

dobivamo:

$$\Delta\mathbf{Q}_{kt}^{(s+1)} = \Delta\mathbf{Q}_{tk}^{(s+1)} = \Delta\mathbf{Q}_{tk}^{(s)} + \Delta\mathbf{Q}_{t+1,k}^{(s)}, \quad k = 1, \ldots, t-1.$$

Ažuriranje modularnosti Q svodi se na dodavanje odgovarajućeg elementa matrice $\Delta\mathbf{Q}^{(s)}$:

$$Q^{(s+1)} = Q^{(s)} + \Delta\mathbf{Q}_{t,t+1}^{(s)}.$$

Ovo jednostavno pravilo za ažuriranje matrice $\Delta\mathbf{Q}$ zahtijeva čuvanje cijele matrice u memoriji. Iz prethodnih razmatranja vidi se da je moguće ažurirati matricu i na sljedeći način.

Za $k \in 1, \ldots, t-1$:

ako je $C_k^{(s)}$ povezan i sa $C_t^{(s)}$, i sa $C_{t+1}^{(s)}$, stavimo

$$\Delta\mathbf{Q}_{kt}^{(s+1)} = \Delta\mathbf{Q}_{tk}^{(s+1)} = \Delta\mathbf{Q}_{tk}^{(s)} + \Delta\mathbf{Q}_{t+1,k}^{(s)},$$

ako je $C_k^{(s)}$ povezan samo sa $C_t^{(s)}$, stavimo

$$\Delta\mathbf{Q}_{kt}^{(s+1)} = \Delta\mathbf{Q}_{tk}^{(s+1)} = \Delta\mathbf{Q}_{tk}^{(s)} - 2a_{t+1}^{(s)}a_k^{(s)},$$

ako je $C_k^{(s)}$ povezan samo sa $C_{t+1}^{(s)}$, stavimo

$$\Delta\mathbf{Q}_{kt}^{(s+1)} = \Delta\mathbf{Q}_{tk}^{(s+1)} = \Delta\mathbf{Q}_{t+1,k}^{(s)} - 2a_t^{(s)}a_k^{(s)}.$$

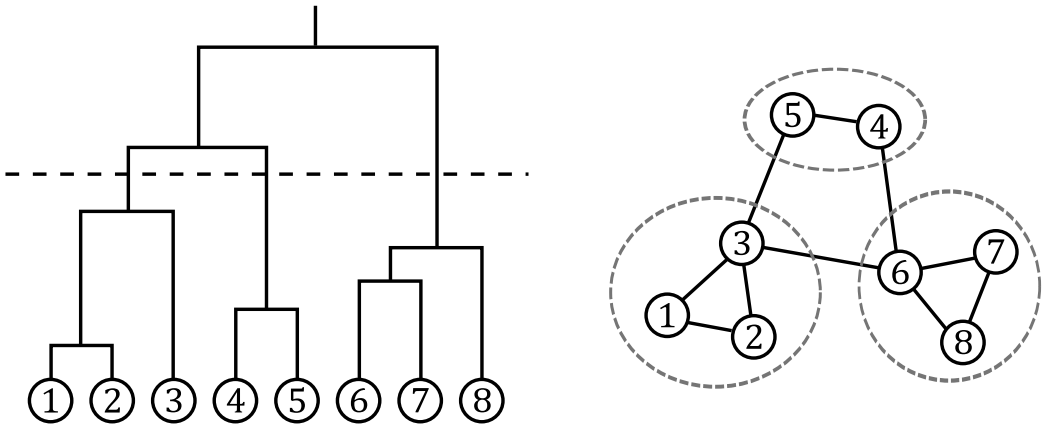
Uz ovaj način ažuriranja, potrebno je u matrici $\Delta\mathbf{Q}$ čuvati samo vrijednosti za međusobno povezane članove particije skupa vrhova. Iako je potrebno čuvati i vrijednosti a_i, prostorna složenost algoritma se uvelike smanjuje za grafove koji su rijetko povezani, a takvi grafovi se uglavnom pojavljuju u praksi. Pri spajanju $C_t^{(s)}$ i $C_{t+1}^{(s)}$ kao u prethodnom opisu, vrijednosti od $a_k^{(s+1)}$, $k = 1, \ldots, t-1$ ostaju jednake kao u prošlom koraku algoritma; potrebno je samo ažurirati:

$$a_t^{(s+1)} = a_t^{(s)} + a_{t+1}^{(s)}.$$

Ažuriranje matrice $\Delta\mathbf{Q}$ može se dodatno ubrzati čuvanjem redaka matrice u obliku uravnoteženih binarnih stabala, a pronalaženje maksimalnog elementa matrice održavanjem max-hrpe koja sadrži maksimalne elemente redaka. U ovome radu korišten je ovaj pristup, pri čemu su uravnotežena binarna stabla implementirana kao AA-stabla [1].

Stablasta struktura koja nastaje spajanjem grozdova naziva se dendrogram. Svakim horizontalnim rezom dendrograma dobiva se jedna moguća struktura grozdova, pri čemu je rezultat opisanog algoritma rez dendrograma s maksimalnom modularnošću. Na slici 4 prikazan je primjer dendrograma za graf sa osam vrhova. Algoritam počinje od dna lijeve slike, iterativno spajajući grozdove. Ima smisla nastaviti algoritam i kada modularnost počne opadati, te izgraditi potpuni dendrogram jer strukture grozdova koje nastaju daljnjom aglomeracijom često reprezentiraju stvarne strukture u kompleksnoj mreži.

Bitno je naglasiti da opisani algoritam ne pronalazi globalni maksimum modularnosti, po svim particijama skupa vrhova, jer u svakom koraku na *greedy* način odabire grozdove za spajanje. Pronalaženje globalnog maksimuma modularnosti je vrlo težak problem [7].



Slika 4: Dendrogram s jednim rezom i pripadnom podjelom grafa

4.7 Vizualizacija strukture grozdova

Detektirani grozdovi u kompleksnoj mreži prikazani su bojanjem pozadine elemenata i veza u mreži ovisno o pripadnosti pojedinom grozdu. Kod većih i kompleksnijih mreža ovaj pristup ne uspijeva dovoljno jasno vizualno predočiti pojedine grozdove i njihov međusobni odnos. Zato je vizualna prezentacija strukture grozdova mreže poboljšana jednostavnom modifikacijom jačina opruga u algoritmu za raspoređivanje vođenog silom — opruge koje pripadaju vezama unutar pojedinog grozda su ojačane, a opruge veza između različitih grozdova oslabljene. Ovime je postignuta automatska reorganizacija elemenata mreže u ravnini u skladu sa strukturom grozdova.

5 Implementacija

Vizualizacija razvijena u ovome radu bila je zamišljena kao samostalna aplikacija koja se može izvoditi na svim popularnijim operacijskim sustavima. Kako je planirano da, jednom dovršena, vizualizacija bude izdana pod nekom od open source licenci, važno je da licenca alata za razvoj nije ograničavajuća u tom smislu. Uz ove kriterije, poželjno je da je odabrani alat u visokom stupnju razvoja, te da je razvoj alata aktivan. Zbog podrške za interaktivan rad s velikim mrežama, potrebno je da korišteni alat za razvoj bude skalabilan, posebno u smislu hardverskog ubrzanja grafičkih operacija.

Mnogi popularni programski jezici, uz odgovarajuće biblioteke potprograma, lako bi zadovoljili navedene uvjete, ali odabirom najboljeg alata može se bitno ubrzati i olakšati razvoj aplikacije. Zbog ovoga su razmotreni razni softverski alati za vizualiziranje kompleksnih mreža. Neki od njih prikazani su u tablici 1.

Tablica 1: Alati za vizualiziranje mreža

	Protovis	**Gephi**	**MSAGL**	**Processing**
Tip	programski alat	aplikacija	biblioteka potprograma	programsko okruženje
Svrha	vizualizacija	vizualizacija i analiza mreža	reprezentacija i modeliranje mreža	vizualizacija
Open source	da	da	ne	da
Aktivan razvoj	ne	da	ne	da
Operacijski sustav	razni (web preglednik)	Windows, Linux, MacOS	Windows	Windows, Linux, MacOS
Programski jezik	JavaScript	Java	C# .NET	baziran na Javi
Hardversko ubrzanje	ovisno o web pregledniku	da	moguće kroz vanjske alate	da

Protovis (`http://vis.stanford.edu/protovis/`) je open source programski alat razvijen na sveučilištu Stanford. Kroz crtanje osnovnih grafičkih elemenata omogućuje stvaranje vizualne prezentacije podataka, pri čemu je korištenjem JavaScripta moguće ostvariti potrebne programske konstrukte, kao što su algoritmi za raspoređivanje elemenata mreže u ravnini i analizu strukture mreže. Posebnost, ali i nedostatak u kontekstu ovog rada, Protovisa je što se rezultirajuća vizualizacija izvodi u okruženju web preglednika, što bitno ograničava mogućnosti hardverskog ubrzanja crtanja mreže.

Gephi (`http://gephi.org/`) je samostalna open source aplikacija za interaktivnu vizualizaciju i analizu kompleksnih mreža i sustava primjenom mnogih ugrađenih metrika. Omogućuje i analizu dinamičkih mreža, te primjenu algoritama za raspoređivanje elemenata mreže. Gephi podržava usmjerene grafove ali nema podršku za višestruke veze između vrhova (multigrafove) i ontologije. Korišten je u mnogim projektima iako je još u relativno ranoj fazi razvoja.

MSAGL (Microsoft Automatic Graph Layout, `http://research.microsoft.com/en-us/`) je .NET biblioteka potprograma za raspoređivanje elemenata mreže, izdana pod komercijalnom licencom. Korištenjem MSAGL-a, uz dodatne biblioteke potprograma, moguće je ostvariti samostalne aplikacije s grafičkim sučeljem, namijenjene Windows okruženju. Niska portabilnost i restriktivna licenca uvelike umanjuju korisnost ove biblioteke potprograma.

5.1 Processing

Processing (`http://www.processing.org/`) je okruženje za razvoj programa bazirano na programskom jeziku Java, i posebno prilagođeno razvoju vizualno orijentiranih aplikacija s naglaskom na animaciji i interakciji. Razvoj

Processinga započeo je 2001. godine, a u međuvremenu je Processing evoluirao od jednostavnog proširenja Jave u samostalni alat za dizajniranje i izradu prototipova [9].

Za razliku od većine alata za vizualiziranje informacija, Processing ne nudi unaprijed definirane predloške za vizualni prikaz podataka, nego samo okvir za razvoj vizualizacija u obliku programskog sučelja, pružajući korisniku potpunu slobodu u izgradnji vizualne prezentacije informacija.

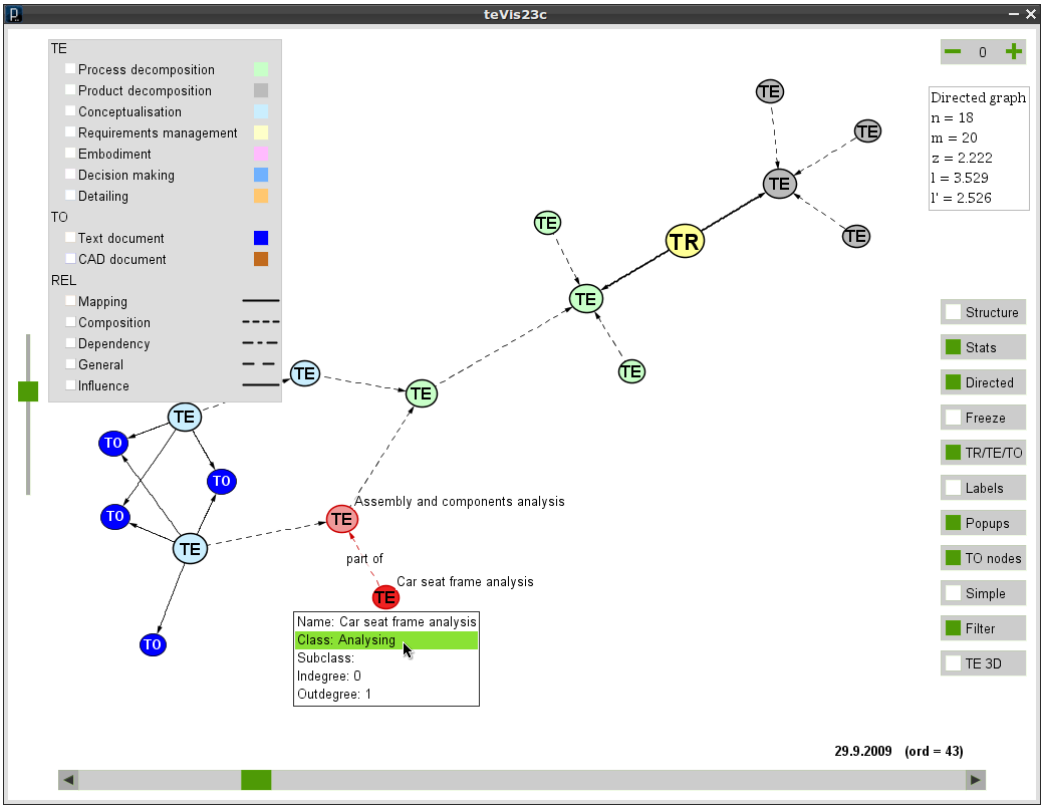
S druge strane, za razliku od "golih" programskih jezika, Processing nudi mnoge funkcije za crtanje osnovnih grafičkih likova, iscrtavanje teksta, animaciju, trodimenzionalnu grafiku i rad s ulaznim uređajima — sve u jednostavnom programskom sučelju, uz mogućnost proširivanja modulima pisanima u Javi.

Uz open source licencu, aktivan razvoj, visoku portabilnost i podršku za hardversko ubrzanje grafičkih operacija, Processing se pokazao kao izvrstan alat za implementaciju vizualizacije opisane u ovom radu.

5.2 Grafičko korisničko sučelje

Iako Processingovo programsko sučelje ne sadrži elemente i grafičke kontrole za izgradnju grafičkog korisničkog sučelja (kao što su gumbi, padajući izbornici i slično), korištenjem grafičkih funkcija i ugrađene podrške za ulazne uređaje moguće je implementirati potrebnu funkcionalnost.

Za potrebe vizualizacije implementirani su osnovni elementi grafičkog sučelja. Prednost ovakve "ručne" izgradnje grafičkog sučelja je fleksibilnost u prezentaciji i funkcionalnosti elemenata grafičkog sučelja.



Slika 5: Grafičko korisničko sučelje

Vizualizacija učitava zapise sljedivosti temeljene na ontologiji, ili druge kompleksne mreže, iz tekstualne datoteke u CSV (comma-separated values) formatu. Sam grafički prikaz mreže je glavni oblik prezentacije izlaznih podataka, a moguće je i ispisati u datoteke detektiranu strukturu grozdova mreže i neke karakteristike elemenata mreže (kao što su stupnjevi elemenata), za korištenje u daljnjoj analizi pomoću drugih alata.

6 Testni slučajevi

Vizualizacija je za vrijeme razvoja testirana na slučajno generiranim mrežama. U svrhu validacije vizualizacije na stvarnim primjerima, korištene su stvarne kompleksne mreže, a dva primjera stvarnih mreža opisana su u ovom poglavlju — ontološka mreža inženjerskih informacija nastala pri razvoju proizvoda, te mreža povezanosti ključnih riječi znanstvenih članaka. Osnovne karakteristike ovih dviju mreža prikazane su u tablici 2, pri čemu je za vrijednost modularnosti Q uzeta modularnost strukture grozdova pronađene prethodno opisanim algoritmom.

Tablica 2: Karakteristike promatranih mreža

mreža	vrsta grafa	n	m	z	l	l'	Q
ontološka mreža (aktivni naslon za glavu automobilskog sjedala)	usmjereni graf	53	86	3.245	3.445	2.844	0.549
mreža ključnih riječi (Znanost o konstruiranju)	neusmjereni graf	1263	3364	5.327	$+\infty$	5.519	0.701

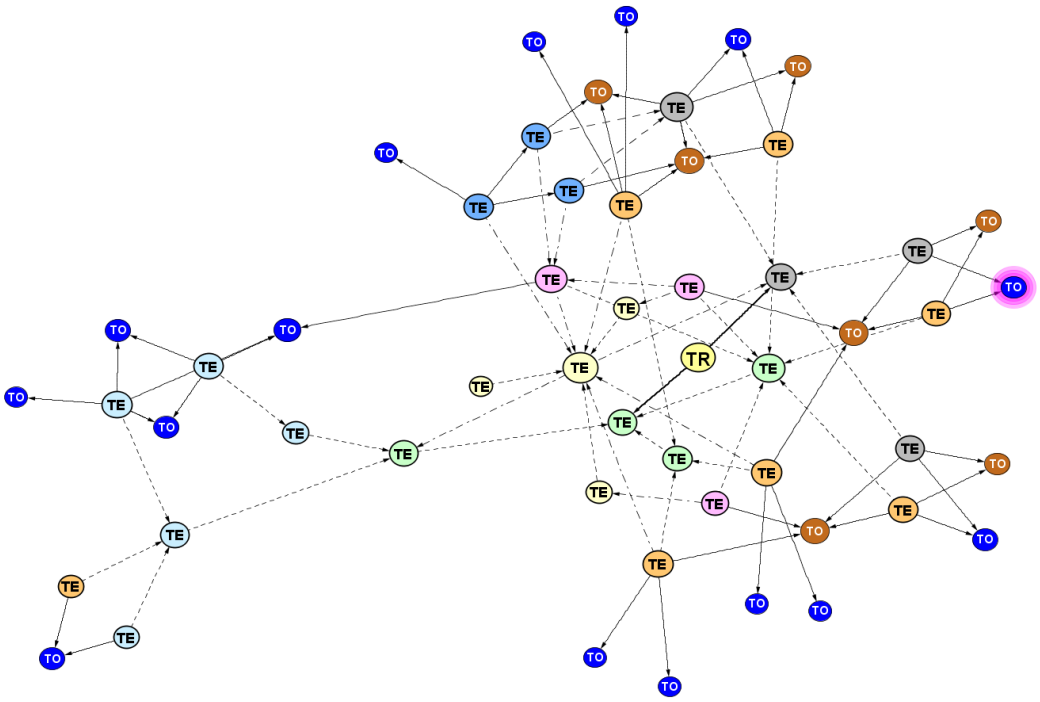
6.1 Razvoj aktivnog naslona za glavu za automobilska sjedala

Ova semantička mreža nastala je u sklopu projekta TRENIN proučavanjem evolucije inženjerskih informacija nastalih pri razvoju aktivnog naslona za glavu za automobilska sjedala. Sastoji se od 53 elementa povezanih s ukupno 86 veza. Čak i ova relativno mala mreža višestruko premašuje radni kapacitet ljudske memorije, te je shvaćanje strukture i evolucije mreže bitno olakšano njenom vizualizacijom.

Na slici 6 vidi se prikaz mreže na kraju procesa razvoja — treba naglasiti

da rast mreže slijedi i rast informacija pa i rast mreže sljedivosti informacija. Tekstualna oznaka unutar elementa mreže označava klasu elementa ontologije sljedivosti koji je instanciran tim elementom mreže, a bojom su označene različite klase elemenata mreže. Primjerice, u ovoj mreži se objekti sljedivosti (označeni s TO) nalaze u jednoj od dvije klase — tekstualni dokumenti (označeni smeđom bojom) i CAD (Computer Aided Design) dokumenti označeni tamno plavom bojom. U tablici 3 nalazi se legenda uz sliku 6.

Na krajnjem desnom dijelu slike može se vidjeti element mreže okružen ružičastim sjajem — ovime je označeno nastajanje i procesiranje inženjerske informacije, te interakcija korisnika (u točno određenom trenutku) sa točno određenom datotekom koja sadrži potrebne inženjerske informacije (CAD model, Word dokument, Excel tablica itd).



Slika 6: Ontološka mreža

Tablica 3: Oznake elemenata ontološke mreže

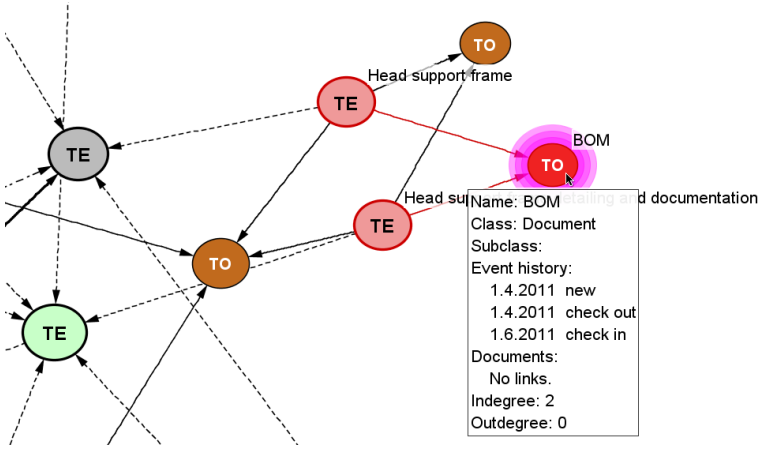
TR	Zapis sljedivosti

Elementi sljedivosti	
TE	Dekompozicija procesa razvoja
TE	Dekompozicija proizvoda
TE	Koncipiranje
TE	Upravljanje zahtjevima
TE	Razrada
TE	Donošenje odluka
TE	Detaljiranje

Objekti sljedivosti	
TO	CAD dokument
TO	Tekstualni dokument

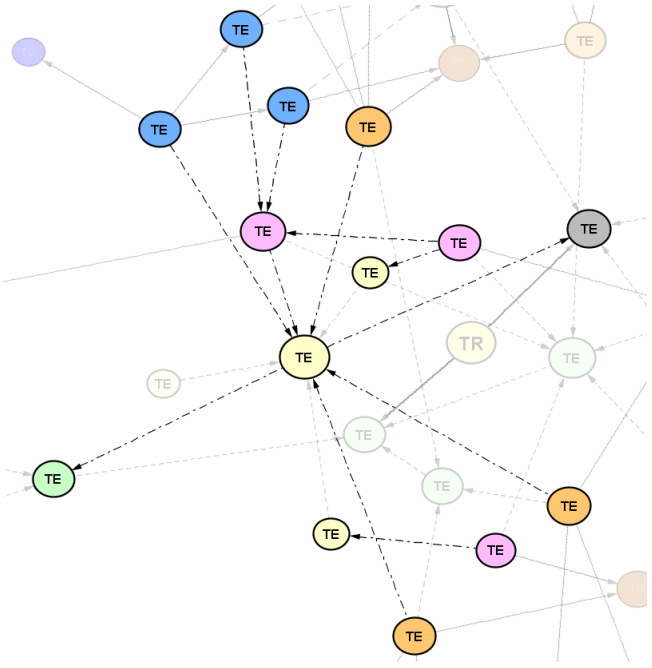
Ovakva struktura je puno lakše sljediva kroz faze konstruiranja nego uobičajena struktura direktorija i tablična struktura koje nude današnji PDM/PLM (product data management/product lifecycle management) sustavi za upravljanje informacijama u razvoju proizvoda, te je razumno očekivati da bi proces upravljanja informacijama u razvoju proizvoda bio znatno poboljšan korištenjem ovakvog alata za vizualizaciju.

Postavljanjem pokazivača miša na element mreže, pojavljuje se prozor sa informacijama o elementu, te se crvenom bojom i tekstualnim oznakama označavaju incidentni bridovi i susjedni elementi, kao što je prikazano na slici 7.



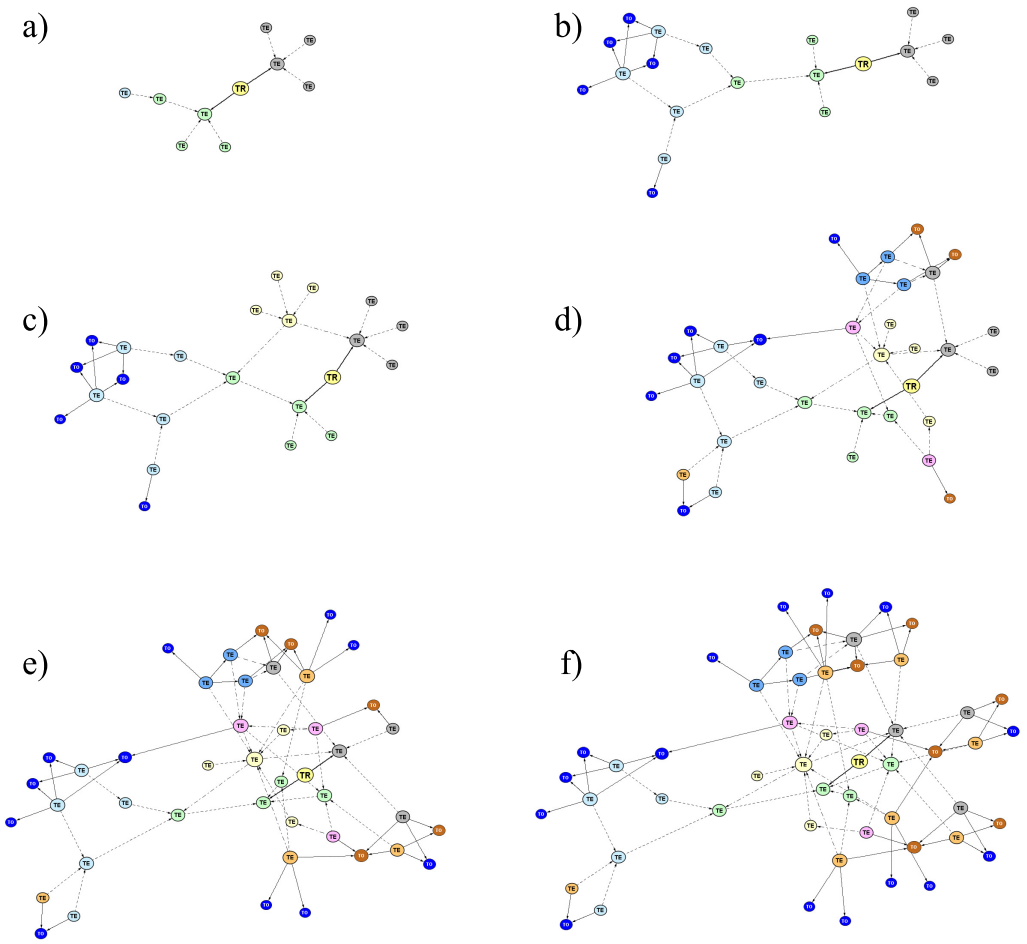
Slika 7: Prikaz informacija o objektu sljedivosti kao elementu mreže

Izdvajanje dijela mreže moguće je filtriranjem po klasi elementa ili klasi relacije u mreži. Na slici 8 prikazan je dio mreže generiran jednom od relacija.



Slika 8: Filtriranje elemenata u mreži obzirom na prije definiranu ontologiju

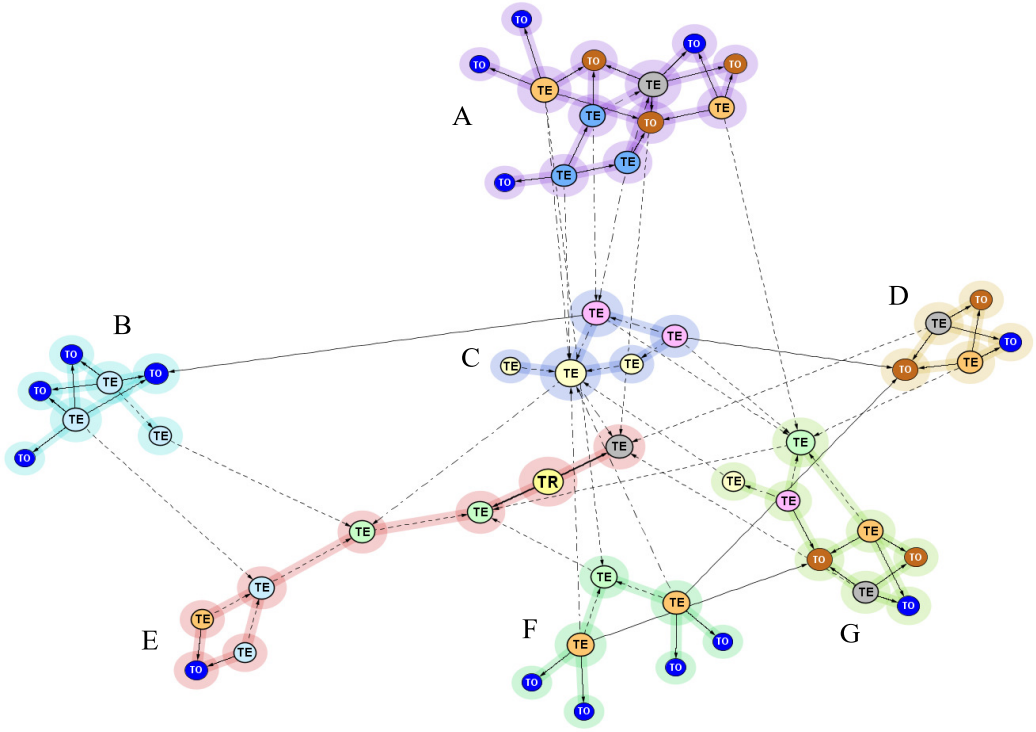
Prikaz evolucije mreže nalazi se na slici 9. Slike označene slovima a) do f) prikazuju mrežu u šest različitih vremenskih trenutaka prikaza. U vizualizaciji se, mijenjanjem vremenskog trenutka prikaza, mreža glatko prilagođava novonastaloj strukturi.



Slika 9: Prikaz evolucije mreže

Na slici 10 je prikaz mreže nakon detekcije strukture grozdova — Vidi se sedam grozdova označenih različitom bojom pozadine i automatska reorganizacija elemenata mreže u skladu sa strukturom grozdova. Tablica 4 prikazuje inženjersko značenje detektiranih grozdova i ilustrira njihovu podudarnost sa

specifičnim aspektima procesa razvoja proizvoda.



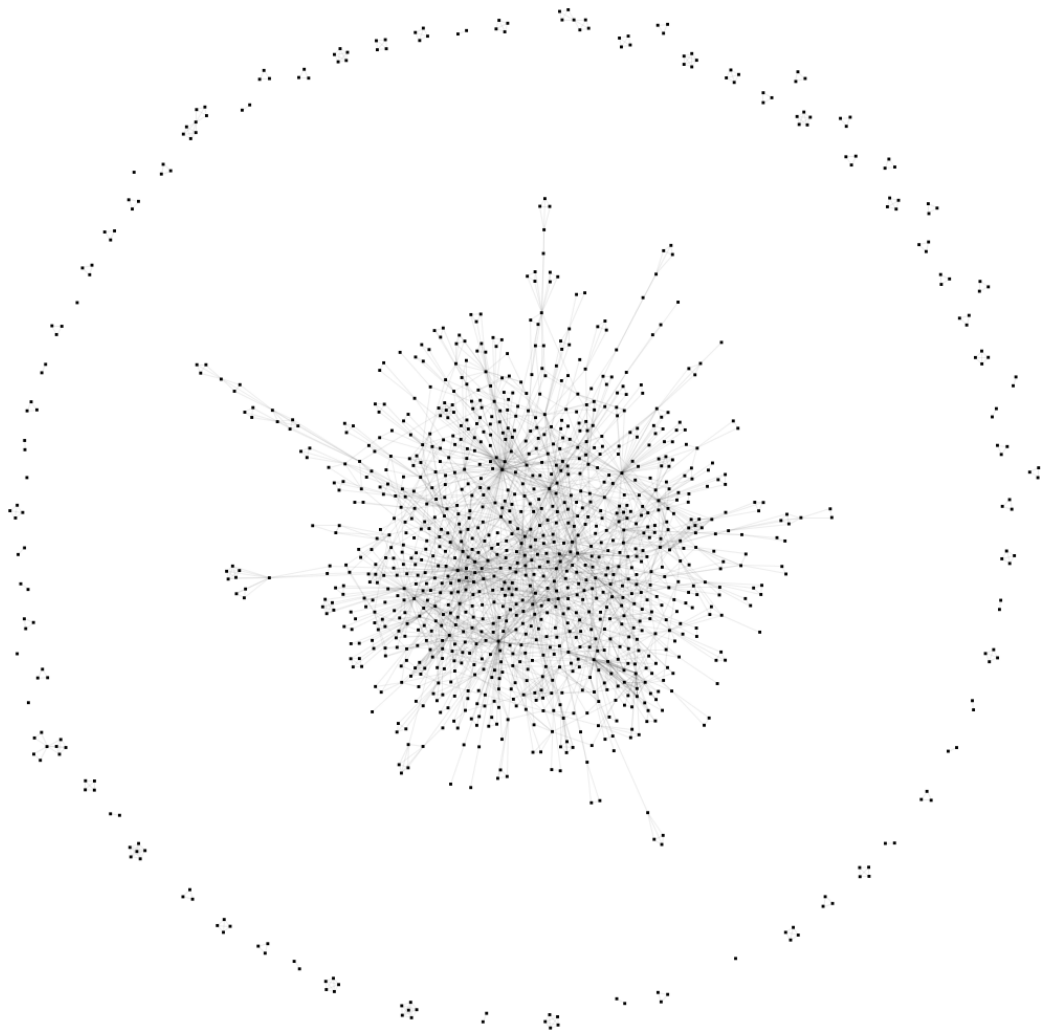
Slika 10: Struktura grozdova ontološke mreže

Tablica 4: Sadržaj grozdova

Grozd	Sadržaj
A	virtualna analiza dva koncepta mehanizma za pozicioniranje naslona za glavu; proces donošenja odluka vezan uz odabir koncepta na kojem se temelji daljnji razvoj
B	informacije vezane uz sigurnosne propise i zakonsku regulativu o sjedalima osobnih vozila
C	analiza zahtjeva i provjera za razinu detaljiranja procesa razvoja sjedala
D	pripadajući modeli i tehnička dokumentacija naslona za glavu
E	analiza postojećih rješenja sjedala obzirom na postojeće zahtjeve
F	virtualna analiza odziva konstrukcije okvira sjedala i naslona za glavu
G	pripadajući CAD modeli i tehnička dokumentacija okvira sjedala obzirom na početne zahtjeve

6.2 Znanstvena evolucija područja Znanosti o konstruiranju

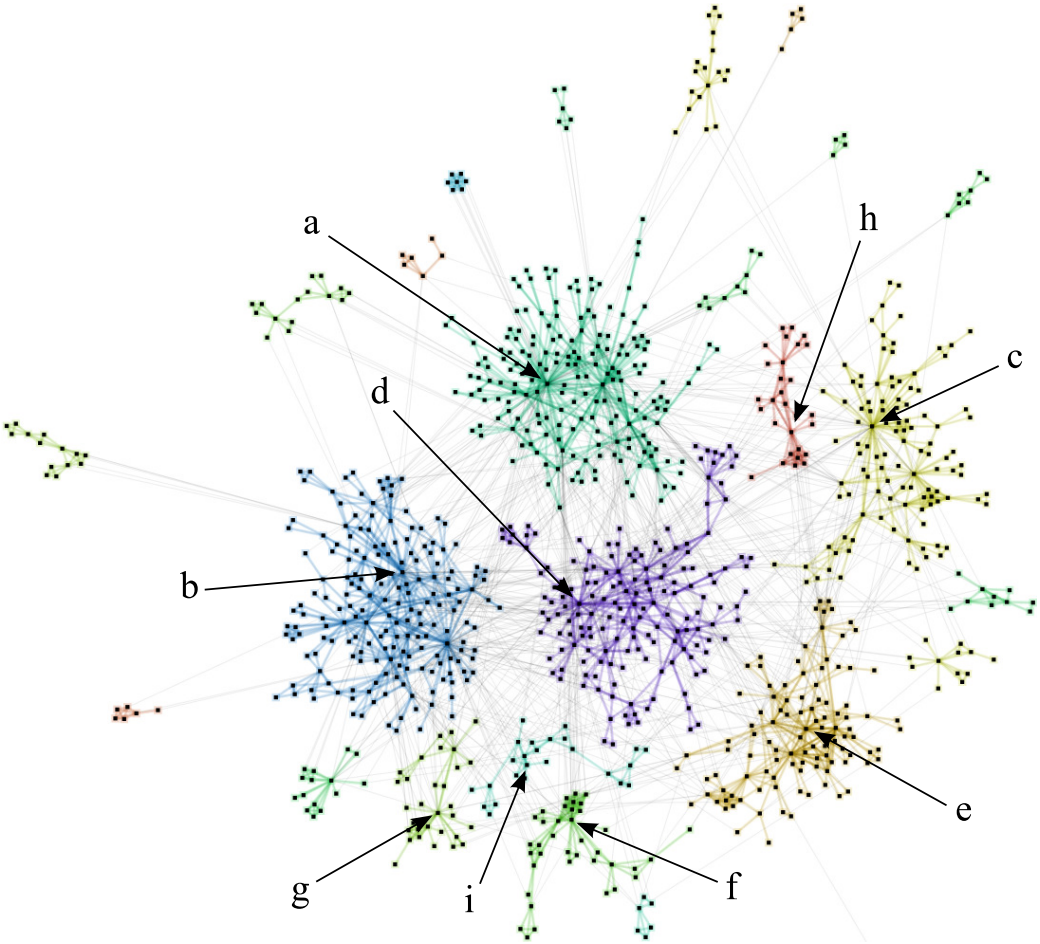
Mreža povezanosti ključnih riječi znanstvenih članaka iz područja Znanosti o konstruiranju nastala je međusobnim povezivanjem ključnih riječi ukoliko se one pojavljuju kao ključne riječi u istom radu.



Slika 11: Mreža ključnih riječi Znanosti o konstruiranju

U ovom testnom slučaju nema ontologije kao osnove za definiranje semantike,

nego su ključne riječi iskorištene onako kako su ih definirali autori, uz minimalnu normalizaciju koja je nužna za ispravljanje pravopisnih pogrešaka i jednoznačno uvrštavanje sinonima. Na slici 11 je kumulativni prikaz razvoja kroz radove objavljene na tri međunarodne znanstvene konferencije (održane 2002., 2004. i 2006. godine) iz serije DESIGN u području Znanosti o konstruiranju. Ključne riječi predstavljene su vrhovima, a veze između ključnih riječi bridovima grafa.



Slika 12: Struktura grozdova velike komponente povezanosti

Osnovna struktura mreže vidi se na slici 11 — mreža se sastoji od jedne

velike komponente povezanosti i većeg broja manjih. U manjim komponentama povezanosti nalaze se ključne riječi članaka proizašlih iz potpuno novih istraživanja ili istraživanja koja tematski ne spadaju u područje.

Na slici 12 prikazana je struktura grozdova centralne komponente povezanosti mreže. Male komponente povezanosti čine zasebne grozdove te su ispuštene sa slike. Analiza strukture grozdova pokazuje da se unutar grozdova nalaze srodne ključne riječi, te bi se ova zapažanja mogla iskoristiti primjerice pri određivanju usko povezanih područja istraživanja ili pri organiziranju znanstvenih konferencija i/ili određivanju smjera istraživanja. Neke od centralnih ključnih riječi u pojedinim grozdovima označene su na slici i popisane u tablici 5 zajedno s njihovim ukupnim brojevima veza u mreži.

Tablica 5: Centralne ključne riječi u grozdovima

oznaka	ključna riječ	broj veza (k)
a	CAD (computer aided design)	105
b	design education	80
c	FEA (finite element analysis)	63
d	knowledge management	56
e	life cycle assessment	47
f	visualisation	40
g	project management	24
h	safety analysis	20
i	innovation management	10

7 Zaključci

Vizualizacija opisana u ovom radu pokazala se kao vrlo koristan alat pri analizi kompleksnih ontoloških mreža nastalih u okviru sljedivosti evolucije inženjerskih informacija u razvoju proizvoda. Upotrebom varijante algoritma vođenog silom za raspoređivanje elemenata mreže, postignut je organski prikaz kompleksnih mreža i njihovog rasta kroz vrijeme, jasno predočavajući dinamiku evolucije informacija u razvoju proizvoda, a detektiranjem i vizualizacijom strukture grozdova mreže pružen je uvid u strukturu kompleksnih mreža — pogotovo koristan pri radu s većim mrežama.

Implementirani alat za vizualizaciju dovoljno je brz za rad s mrežama od nekoliko tisuća elemenata, a daljnja skalabilnost planirana je razvijanjem paralelnih algoritama i korištenjem snažnih paralelnih računala. Jedan od budućih smjerova razvoja je i prikaz kompleksnih mreža u trodimenzionalnom prostoru te evaluacija takve vrste korisničkog sučelja u odnosu na klasično dvodimenzionalno sučelje. Daljnji razvoj obuhvaća i analizu tranzitivnosti elemenata, analizu skalabilnosti mreže, pronalaženje glavnih pod-struktura unutar modula mreže itd.

Iako je prvenstveno namijenjena vizualiziranju vrlo specifične vrste semantičkih mreža koje se temelje na ontologiji, implementirana vizualizacija može se koristiti i za rad s drugim vrstama mreža. Kako bi se omogućila bolja analiza kompleksnih mreža, potrebno je implementirati širi raspon algoritama za analizu njihove strukture. Proučavanje ovih algoritama i njihove efikasne implementacije još je jedan od smjerova daljnjeg rada.

8 Zahvale

Zahvaljujem se mentoru, doc. dr. sc. Mariu Štorgi i komentoru prof. dr. sc. Saši Singeru, i posebno dr. sc. Tini Stankoviću na motivaciji pri izradi ovog rada, korisnim sugestijama, i ustupanju primjeraka kompleksnih mreža.

Također se zahvaljujem prof. dr. sc. Sanji Singer na pružanju početnog impulsa i prilike za rad u ovom području.

9 Popis literature

[1] A. Andersson, *Balanced search trees made simple*, Proc. 3rd Workshop on Algorithms and Data Structures, 60-71, 1993.

[2] J. Barnes, P. Hut, *A hierarchical $O(N \log N)$ force-calculation algorithm*, Nature, 324(4), 446-449, 1986.

[3] G. Di Battista, P. Eades, R. Tamassia, I.G. Tollis, *Graph Drawing: Algorithms for the Visualization of Graphs*, Prentice-Hall, Englewood Cliffs, NJ, 1998.

[4] S.R. Buss, *3-D Computer Graphics: A Mathematical Introduction with OpenGL*, Cambridge University Press, Cambridge, 2003.

[5] C. Chen, *Information Visualization – Beyond the Horizon*, Springer-Verlag, London, 2006.

[6] A. Clauset, M.E.J. Newman, C. Moore, *Finding community structure in very large networks*, Phys. Rev. E 70, 066111, 2004.

[7] L. Danon, J. Duch, A. Díaz-Guilera, A. Arenas, *Comparing community structure identification*, J. Stat. Mech. 2005 (09): P09008, 2005.

[8] B.J. Fry, *Organic Information Design*, Master thesis, Massachusetts Institute of Technology, 2000.

[9] B.J. Fry, *Visualizing Data*, O'Reilly Media, Inc., Sebastopol, CA, 2008.

[10] V. Hubka, W.E. Eder, *Theory of technical systems: a total concept theory for engineering design*, Springer-Verlag, Berlin; New York, 1988.

[11] U. Lindemann, M. Maurer, T. Braun, *Structural Complexity Management: An Approach for the Field of Product Design*, Springer-Verlag, Berlin; Heidelberg, 2009.

[12] M.E.J. Newman, *The structure and function of complex networks*, SIAM Review 45, 167-256, 2003.

[13] M.E.J. Newman, *Fast algorithm for detecting community structure in networks*, Phys. Rev. E 69, 066133, 2004.

[14] N. Pavković, et al., *Case studies to explore indexing issues in product design traceability*, Proceedings of the 18th International Conference on Engineering Design – ICED 11, Copenhagen, Denmark, 2011.

[15] M. Štorga, *Traceability in product development*, Proceedings of the DESIGN 2004 - 8th International DESIGN Conference, FSB, Zagreb, The Design Society, Glasgow, 2004.

[16] M. Štorga, M.M. Andreasen, D. Marjanović, *The Design Ontology: Foundation for the Design Knowledge Exchange and Management*, Journal of Engineering Design 21(4), 427-454, 2010.

[17] M. Štorga, et al., *Traceability of engineering information development in PLM*, 8th International Conference on Product Lifecycle Management PLM'11. Eindhoven, The Netherlands, 2011.

[18] M. Štorga, D. Marjanović, T. Savšek, *Reference model for traceability records implementation in engineering design environment*, Proceedings of the 18th International Conference on Engineering Design – ICED 11, Copenhagen, Denmark, 2011.

10 Sažetak

Ivan Stojić

Vizualizacija kompleksnih mreža u kontekstu sljedivosti evolucije informacija u razvoju proizvoda

Ključne riječi: vizualizacija, sljedivost informacija, raspoređivanje vrhova grafa vođeno silom, detekcija strukture grozdova

U ovom radu opisan je i implementiran alat za vizualizaciju ontoloških mreža nastalih u kontekstu sljedivosti evolucije informacija u razvoju proizvoda. Nakon kratkog opisa sljedivosti evolucije informacija navedeni su ciljevi razvoja alata za vizualizaciju.

Nadalje, opisana je varijanta algoritma za raspoređivanje vrhova grafa vođenog silom, Barnes–Hutov algoritam za rješavanje problema n tijela, te upotreba ovih algoritama u vizualizaciji evolucije kompleksnih mreža. Zatim je opisan brzi algoritam za detekciju strukture grozdova mreže i vizualizacija strukture grozdova uz pomoć algoritma za raspoređivanje vrhova grafa vođenog silom.

Na kraju je dan kratki pregled alata za vizualiziranje kompleksnih mreža, i prikazani su rezultati za dvije kompleksne mreže dobiveni implementiranim alatom za vizualizaciju — njihova vizualizacija i zaključci dobiveni analizom strukture grozdova.

11 Summary

Ivan Stojić

Visualisation of complex networks in the context of traceability of engineering information

Keywords: visualisation, traceability of information, force directed layout, detecting community structure

In this work I have described and developed a tool for visualisation of engineering information evolution during product development. After a short description of traceability of engineering information evolution, main goals of the visualisation tool are described.

Next, I have described a variant of the force directed graph layout algorithm and Barnes–Hut algorithm and explained how these algorithms were used in visualising the evolution of complex networks. After that, fast algorithm for detecting community structure in networks, and using force directed layout algorithm in visualising community structure are described.

In the end, I have given a short review of several graph visualisation toolkits, and shown the results obtained by using the visualisation tool that has been developed for two complex networks — their visualisation and insights obtained by analysing their community structures.