Högskolan i Gävle

Projektrapport

Datavisualization - Design and Construction, 7,5 hp

HT 19, period 1

Visualisering av källkod

av

Thomas Lundgren
900802-1835
thomaslundgren@live.com

Niklas Nordgren
901010-5758
n_nordgren@hotmail.com

Hanna Medén
960214-0965
hannameden@hotmail.com

Institutionen för matematik, natur- och datavetenskap
Högskolan i Gävle
S-801 76 Gävle, Sweden

Innehållsförteckning

1 Inledning 1
1.1 Bakgrund 1
1.2 Syfte 1
1.3 Frågeställning 1
2 Metod 2
2.1 Bidragshistorik till Git-repository 2
2.1.1 Importera genererade data till Matlab 2
2.1.2 Diagram över de bidragsgivare som bidragit mest 4
2.2 Tillämpande av LOC på olika kodbaser 5
2.3 Riktade grafer 6
3 Resultat 11
3.1 Bidragshistorik till Git-repository 11
3.1.1 Diagram över de bidragsgivare som bidragit mest 11
3.1.2 Utskrift av antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare 11
3.1.3 Visualisering av antalet bidrag över tid 12
3.2 Tillämpande av LOC på olika kodbaser 13
3.2.1 Diagram över ett ramverks beståndsdelar (Programmeringsspråk) 13
3.2.2 Diagram över ramverkens beståndsdelar (Antal filer) 14
3.2.3 Diagram över ramverkens beståndsdelar (Antal rader kod) 14
3.3 Riktade grafer 15
4 Diskussion 17
4.1 Bidragshistorik till Git-repository 17
4.2 Tillämpande av LOC på olika kodbaser 17
4.3 Riktade grafer 17
5 Referenser 18

1 Inledning

1.1 Bakgrund

Källkod bör vara enkel att upprätthålla för att minimera tidsåtgången av mjukvaruutveckling. Faktorerna som påverkar detta är bland annat kodbasens klassrelationer, komplexitet och struktur.

Visualiseringar av källkod ger utvecklare och andra intressenter en bättre överblick över källkodens skick, samt om det förekommer brister som måste åtgärdas.

1.2 Syfte

Syftet med detta projekt var att undersöka möjligheterna att visualisera källkod i dataanalysverktyget Matlab.

1.3 Frågeställning

För att avgränsa arbetet valdes tre olika aspekter av en kodbas som skulle visualiseras:

- Relationer mellan klasser i en Java-kodbas.
- Bidragen (eng. commits) till ett Git-repository över tid.
- Storleken hos en kodbas.

Avgränsningen resulterade i följande frågeställningar:

- Hur visualiseras relationer mellan klasser i Java-källkod med användning av en riktad graf på bästa sätt?
- Hur visualiseras relationer mellan klasser i Java-källkod med användning av en grannmatris (eng. adjacency matrix) på bästa sätt?
- Hur visualiseras bidragen till en kodbas över tid med hjälp av bidragshistorik framtaget med versionshanteringssystemet Git på bästa sätt?
- Hur visualiseras storleken hos en kodbas och hur jämförs den med andra kodbasers storlek på bästa sätt?

2 Metod

2.1 Bidragshistorik till Git-repository

2.1.1 Importera genererade data till Matlab

Git lagrar all bidragshistorik förknippat med ett repository i loggfiler. För att generera data över bidragshistoriken användes kommandot git log i Git Bash. Det repository som användes var CSS-ramverket Bootstrap. Resultatet skrevs till en textfil med tabbseparerade värden (TSV-fil). Figur 1 visar innehållet i textfilen.

```
1  054ee07 Thomas Lundgren Fri Oct 25 11:12:50 2019    Merge branch 'dev' of https://github.com/ThomasLundgren/datavis-projekt into dev
2  f5756e3 Thomas Lundgren Fri Oct 25 11:12:45 2019    Added code to callgraph.mlx to create adjacency matrix
3  b96a81d NiklasNordgren  Fri Oct 25 08:55:18 2019    Merge branch 'dev' of https://github.com/ThomasLundgren/datavis-projekt into dev
4  3be2106 NiklasNordgren  Fri Oct 25 08:55:04 2019    Create get_edges_output.txt
5  6c3ed80 Thomas Lundgren Fri Oct 25 08:53:13 2019    Remove irrelevant files
6  cc23a02 NiklasNordgren  Fri Oct 25 08:51:09 2019    Merge branch 'dev' of https://github.com/ThomasLundgren/datavis-projekt into dev
7  0eb4e0e NiklasNordgren  Fri Oct 25 08:51:05 2019    Loc grejjer
8  8ceb9a4 Thomas Lundgren Thu Oct 24 07:56:09 2019    Update thomas.mlx
9  e753057 Thomas Lundgren Tue Oct 22 17:51:19 2019    Fixed bug in get_edges.jar
10 1e83244 Thomas Lundgren Tue Oct 22 17:15:14 2019    Create get_edges_output.txt
11 76259ef Thomas Lundgren Tue Oct 22 17:13:36 2019    Update get_edges.jar
12 5c12082 Thomas Lundgren Tue Oct 22 13:02:28 2019    Create get_edges.jar
13 82aa752 Thomas Lundgren Tue Oct 22 10:58:59 2019    Added regex matching txt-doc, change name of mlx file
14 f6aee94 Thomas Lundgren Wed Oct 16 15:14:38 2019    Fix bootstrap path in SourceWiz
15 472e03d Thomas Lundgren Wed Oct 16 14:57:58 2019    Created SourceWiz.mlx
```

Figur 1: Resultatet av att köra kommandot `git log` i Git bash.

Innehållet i textfilen manipulerades genom Notepad++:s funktion ”hitta och ersätt”. Ett reguljärt uttryck användes som ersatte veckodag, datum, klockslag och kolontecken med ett blanksteg i kolumnen som innehåller datum, se Figur 2.

```
1  054ee   Thomas Lundgren  Oct 2019   Merge branch 'dev' of https//gib.com/ThomasLundgren/datavis-projekt into dev
2  f5756e  Thomas Lundgren  Oct 2019   Added code to callgraph.mlx to create adjacency matrix
3  bad NiklasNordgren   Oct 2019   Merge branch 'dev' of https//gib.com/ThomasLundgren/datavis-projekt into dev
4  be2106  NiklasNordgren   Oct 2019   Create get_edges_output.txt
5  ced Thomas Lundgren  Oct 2019   Remove irrelevant files
6  cca NiklasNordgren   Oct 2019   Merge branch 'dev' of https//gib.com/ThomasLundgren/datavis-projekt into dev
7  ebee    NiklasNordgren   Oct 2019   Loc grejjer
8  ceba    Thomas Lundgren  Oct 2019   Update thomas.mlx
9  e753057 Thomas Lundgren  Oct 2019   Fixed bug in get_edges.jar
10 e83244  Thomas Lundgren  Oct 2019   Create get_edges_output.txt
11 76259ef Thomas Lundgren  Oct 2019   Update get_edges.jar
12 c12082  Thomas Lundgren  Oct 2019   Create get_edges.jar
13 aa752   Thomas Lundgren  Oct 2019   Added regex matching txt-doc, change name of mlx file
14 faee    Thomas Lundgren  Oct 2019   Fix bootstrap path in SourceWiz
15 472ed   Thomas Lundgren  Oct 2019   Created SourceWiz.mlx
```

Figur 2: TSV-filen efter att funktionen ”hitta och ersätt” har använts.

TSV-filen öppnades sedan i Microsoft Excel och sparades i xlsx-format (Exceldokument-format), se Figur 3.

	A	B	C	D	E	F	G	H	I
1	be957d	XhmikosR	Oct 2019	Update devDependencies. (#29508)					
2	7327eb	Martijn Cuppens	Oct 2019	Fix top level ampersand (#29518)					
3	eb428	leshasmp	Oct 2019	Carousel variables (#29493)					
4	eafd	astrahov	Oct 2019	Group line-height variables (#29466)					
5	efcab	Jeremy Jackson	Oct 2019	Add color argument to button mixins (#29444)					
6	b3451ff	Mark Otto	Oct 2019	Add new .bg-body utility class (#29511)					
7	133ecc	XhmikosR	Oct 2019	Drop support for Node.js . (#29496)					
8	577bfb	XhmikosR	Oct 2019	Rename "js/tests/units" to "js/tests/unit". (#29503)					
9	1770691b	XhmikosR	Oct 2019	Dist (#29484)					
10	cd3579	XhmikosR	Oct 2019	CI move `CI` env variable to the root of the workflow. (#29499)					
11	622c914a	XhmikosR	Oct 2019	Update devDependencies. (#29447)					
12	60559da	astrahov	Oct 2019	Add variable for `$breadcrumb-font-size` (#29467)					
13	ebfe	Johann-S	Oct 2019	add modularity integration test					
14	db541c	Johann-S	Oct 2019	return to the original file structure to avoid breaking modularity					
15	393ddae	Martijn Cuppens	Oct 2019	Fix border for single card in accordion (#29453)					

Figur 3: TSV-filen importerad till Microsoft Excel.

Innehållet i Excel-filen importerades slutligen till Matlab genom funktionen `readtable()`. Resultatet redovisas i Figur 4.

	CommitId	Username	Date	CommitDescription
1	'be957d'	'XhmikosR'	'Oct 2019'	'Update devDependen...
2	'7327eb'	'Martijn Cup...	'Oct 2019'	'Fix top level ampersa...
3	'eb428'	'leshasmp'	'Oct 2019'	'Carousel variables (#...
4	'eafd'	'astrahov'	'Oct 2019'	'Group line-height vari...
5	'efcab'	'Jeremy Jac...	'Oct 2019'	'Add color argument t...
6	'b3451ff'	'Mark Otto'	'Oct 2019'	'Add new .bg-body util...
7	'133ecc'	'XhmikosR'	'Oct 2019'	'Drop support for Nod...
8	'577bfb'	'XhmikosR'	'Oct 2019'	'Rename "js/tests/unit...
9	'1770691b'	'XhmikosR'	'Oct 2019'	'Dist (#29484)'
10	'cd3579'	'XhmikosR'	'Oct 2019'	'CI move `CI` env vari...
11	'622c914a'	'XhmikosR'	'Oct 2019'	'Update devDependen...
12	'60559da'	'astrahov'	'Oct 2019'	'Add variable for `$bre...
13	'ebfe'	'Johann-S'	'Oct 2019'	'add modularity integr...
14	'db541c'	'Johann-S'	'Oct 2019'	'return to the original fi...
15	'393ddae'	'Martijn Cup...	'Oct 2019'	'Fix border for single c...

Figur 4: Data importerat till tabell i Matlab genom funktionen readtable().

2.1.2 Diagram över de bidragsgivare som bidragit mest

Koden för att skapa ett diagram över de bidragsgivare som gjort flest bidrag till Bootstrap:s repository redovisas i Figur 5.

```
8   [sortedCommits, newIndices] = sort(commitsPerContr, "descend");
9   sortedContributors = contributors(newIndices);
10  x = 10 ;
11  x = round(x);
12  topxUsers = sortedContributors(1:x);
13
14  sortedContributors = categorical(topxUsers, topxUsers);
15  sortedCommits = sortedCommits(1:x);
16
17  bar(sortedContributors, sortedCommits);
18  s = strcat('Top ', num2str(x), ' contributors');
19  title(s);
```

Figur 5: Koden för att skapa ett stapeldiagram över de användare som bidragit mest.

Koden grupperar antalet bidrag efter bidragsgivare med funktionen `histcounts()`, sorterar efter antalet bidrag i fallande ordning och skapar ett stapeldiagram med funktionen `bar()`.

1.1.2 Utskrift av antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare

Programmet skriver ut antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare (Figur 6).

```
20  numContributors = length(contributors);
21  numCommits = length(gitHistoryTable.CommitId);
22  avgCommits = numCommits/numContributors;
23  disp(['Number of contributors: ', num2str(numContributors)])
24  disp(['Number of commits: ', num2str(numCommits)])
25  disp(['Average commits per contributor: ', num2str(avgCommits)])
```

Figur 6: Koden för att skriva ut antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare.

1.1.3 Visualisering av antalet bidrag över tid

För att visualisera antalet bidrag till Bootstrap:s kodbas över tid skapades ett linjediagram. Figur 7 visar koden som producerar diagrammet.

```
months = categorical(gitHistoryTable.Date);
[commitsPerMonth, months] = histcounts(months);
monthDates = datetime(months);
plot(monthDates, commitsPerMonth, 'LineWidth', 2);

title('Commits over time');
grid on;
xlabel('Year');
ylabel('Number of commits');
```

Figur 7: Koden för att visualisera bidragen till kodbasen över tid.

2.2 Tillämpande av LOC på olika kodbaser

Den öppna programvaran LOC användes på sex olika kodbaser för att få ut visualiseringsdata. Samtliga kodbaser som verktyget användes på är likartade i det avseende att de alla är ramverk eller bibliotek för webbutveckling.

Datat som framställdes gjorde det möjligt att visualisera storleksskillnaden mellan kodbaserna såväl som vilka programmeringsspråk varje kodbas bestod av.

Nedan i Figur 8 visas formatet på det data som LOC producerar till en textfil, som vidare manipulerades med hjälp av reguljära uttryck i Notepad++, för att sedan föras över till ett Excel-dokument. Anledningen till detta var att förenkla hanteringen av data i Matlab.

Language	Files	Lines	Blank	Comment	Code
TypeScript	5100	598501	77716	101687	419098
JavaScript	458	175553	25750	29119	120684
Markdown	261	67580	23953	0	43627
JSON	456	18331	77	0	18254
HTML	495	13009	1576	1310	10123
CSS	199	7484	1071	180	6233
Sass	48	4507	714	129	3664
Bourne Shell	56	2240	427	420	1393
Plain Text	15	487	36	0	451
XML	2	16	0	0	16
Protobuf	1	51	9	27	15
Less	2	1	0	1	0
Total	7093	887760	131329	132873	623558

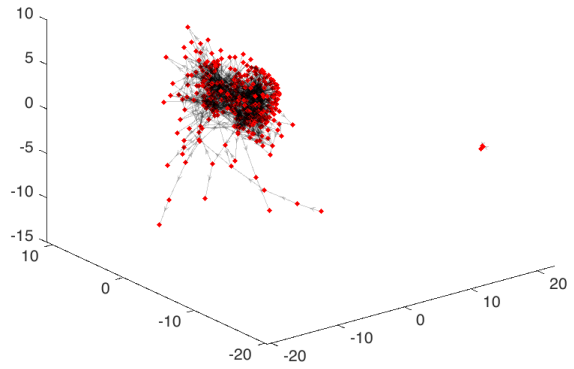
Figur 8: Texfilen som genererats med hjälp av LOC.

Vidare i Matlab lästes data från Excel-filerna in som tabeller med hjälp av funktionen `readtable()` för att på så sätt kunna använda det för visualiseringar.
För att rita stapeldiagrammen i Matlab användes funktionen `bar()`, med transponatet av den inlästa tabellens rubriker som omvandlats till kategoriska data med hjälp av funktionen `categorical()` som x-värden. Tabellens numeriska kolumner utgjorde y-värden till stapeldiagrammet. För att tydliggöra vad varje stapel i diagrammet över ett ramverks beståndsdelar translaterar till för datamängd valde vi att använda färger av olika nyanser på staplarna, samt presentera en teckenförklaring för att visa vilket värde respektive färg motsvarar.

2.3 Riktade grafer

För att få fram data för kopplingar mellan klasserna i källkoden så användes JCallGraph, ett program med öppen källkod där man matar in en JAR-fil och får ut ett textdokument. Textfilen innehöll mer information än det som behövdes för denna uppgift så genom ett egenförfattat Java-program erhölls en textfil där varje rad representerade ett unikt anrop mellan två klasser. Det var då enkelt att läsa in filen i Matlab och deklarera start- och slut-noder och skapa en riktad graf.

De första graferna gjordes i 3D då det ansågs vara ett bra sätt för att få se en översikt och kunna interagera med grafen (Figur 9). Det blev sedan tydligt att det var väldigt svårt att urskilja de olika klasserna (noderna) och vid försök till zoom så ändrades proportionerna på axlarna vilket gjorde att det var lätt att data hamnade utanför det visuella området.

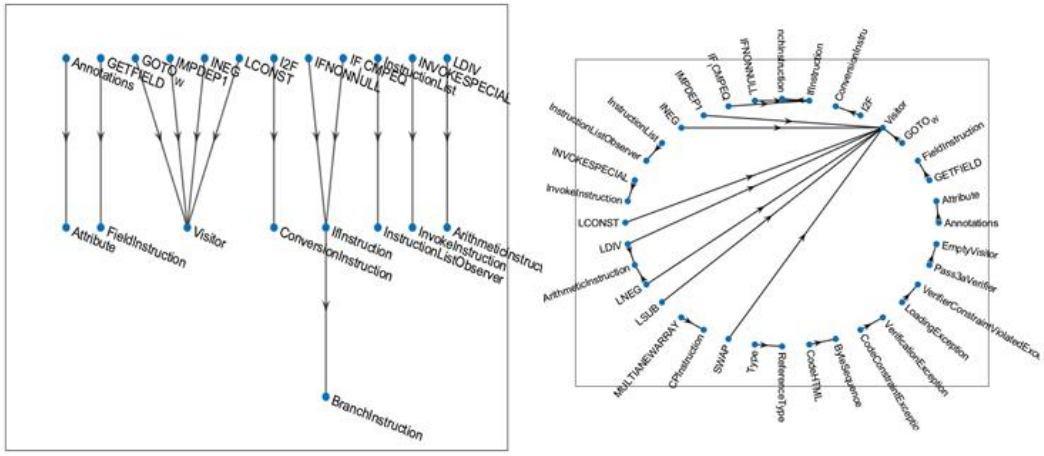


Figur 9: Ett första försök till att skapa en visualisering över relationer mellan klasser med hjälp av en riktad graf.

Då denna visualisering är över ett väldigt stort dataset ansågs en tvådimensionell representation av grafen vara mer tydlig. Detta trots att fler kantlinjer då nödvändigtvis korsar varandra. Forskning utförd av Helen Purchase visar att man bör sträva efter att ha så få kantlinjer som korsar varandra som möjligt för att öka förståelsen för det data som grafvisualiseringen ämnar förmedla [1].

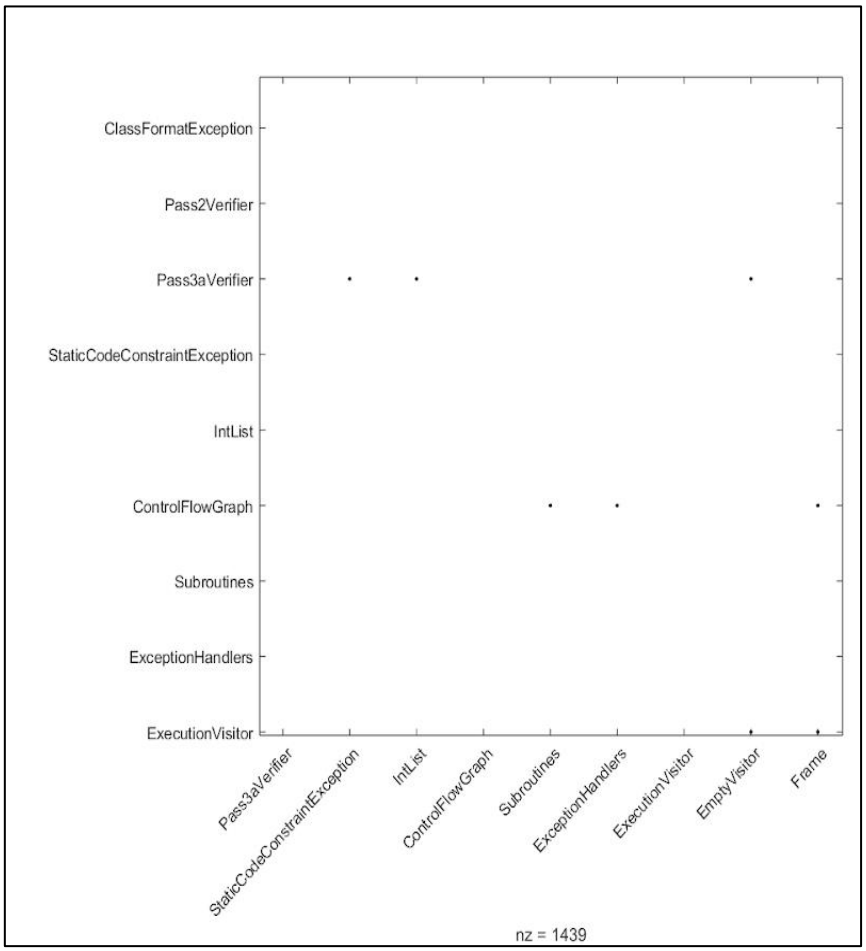
Flera olika typer av grafer producerades genom att tillämpa olika grafritningsalgoritmer genom kommandot `layout` i Matlab, men med tanke på storleken så blev det många som blev alldeles för röriga för att kunna få någon

information ut av den. Figur 10 visar exempel på olika layouts med bara en delmängd av datasetet.



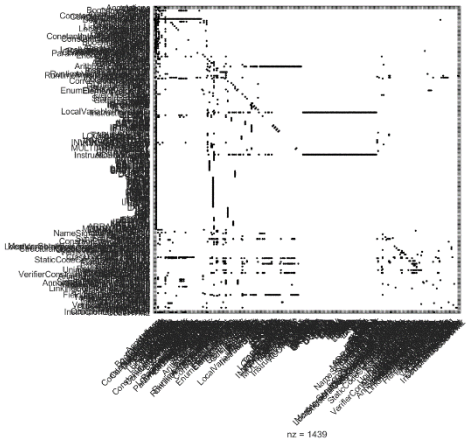
Figur 10: Två alternativa representationer av en riktad graf.

Vi försökte även visa kopplingarna med hjälp av en grannmatris, för att få en översiktlig bild över det hela, men även det blev väldigt rörigt. Figur 11 visar ett exempel på en anropsmatris med en delmängd av datasetet.

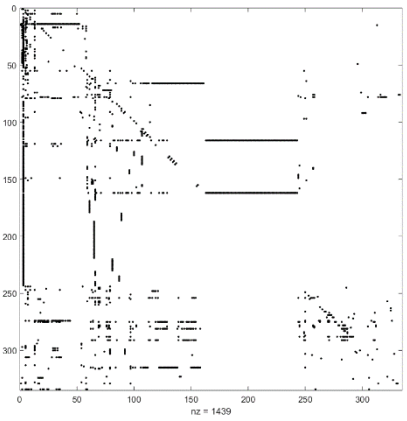


Figur 11: Grannmatris över en delmängd av datasetet.

Figur 12 visar en anropsmatris som innehåller hela datasetet. Visualiseringen är oläslig då alla klassnamnen är skrivna över varandra, men om man väljer att dölja klassnamnen som i Figur 13 så ser man kopplingarna men inte vilka som kopplar, och förlorar då en stor del av den information man vill kunna få ut av en källkods relationer.

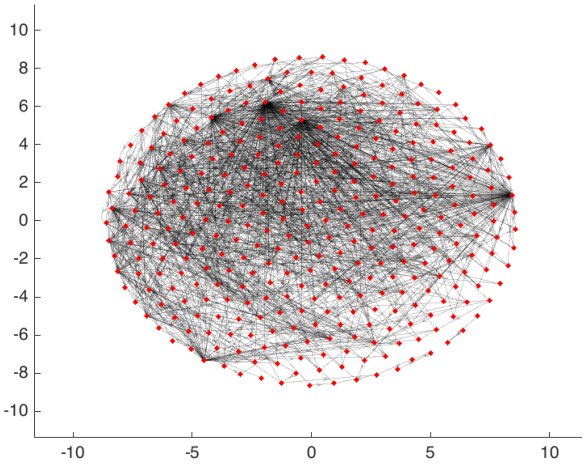


Figur 12: Grannmatris över hela datasetet med klassnamn längs x- och y-axlarna.



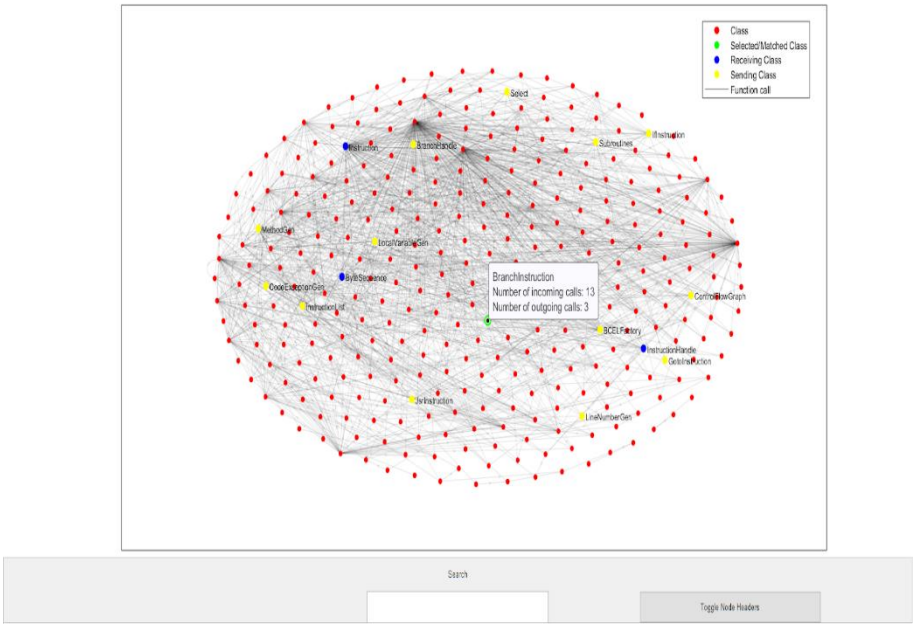
Figur 13: Grannmatris utan klassnamnen.

Trots att visualiseringen i Figur 14 nedan har många korsande kantlinjer ansågs den vara det bästa sättet att visa en stor mängd relationella data. Man ser lätt vilka klasser som har flest kopplingar, men relationernas riktningar går inte att utläsa. Vi valde att höja genomskinligheten i kantlinjerna något, då de nu blir en väldigt mörk figur och på flera ställen helsvart.



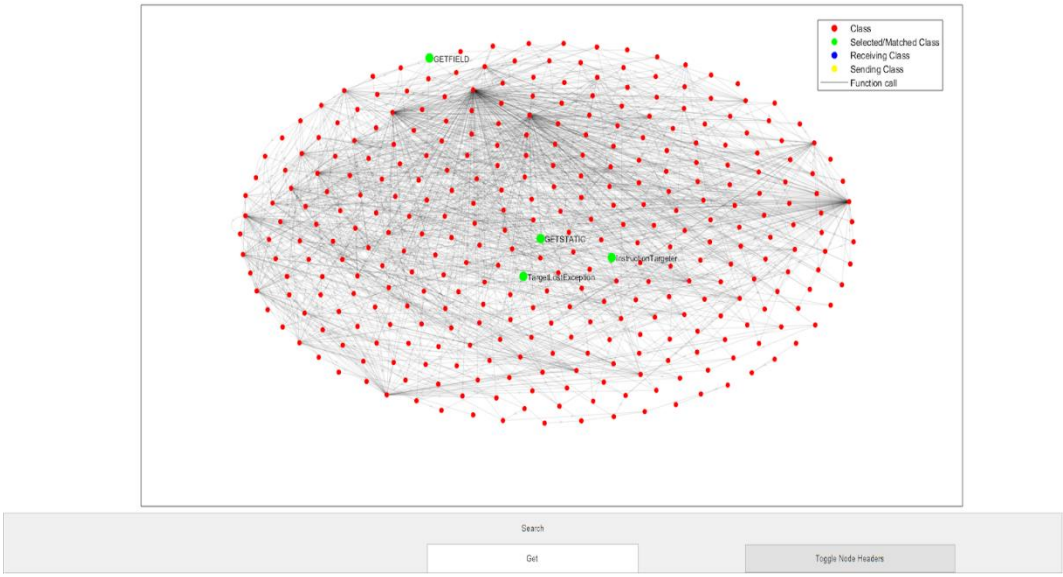
Figur 14: En lämplig visualisering av en anropsgraf.

Funktionalitet så som sökmetod samt markering av klasser som sedan visar upp dennes kopplingar implementerades även för en lättare hantering och interaktivitet. Det går att välja en nod (klass) och då kommer dess namn och alla dess kopplingar upp, deras namn samt om de är in eller utgående anrop (Figur 15).



Figur 15: En nod har klickats på. Inkommande och utgående anrop visas.

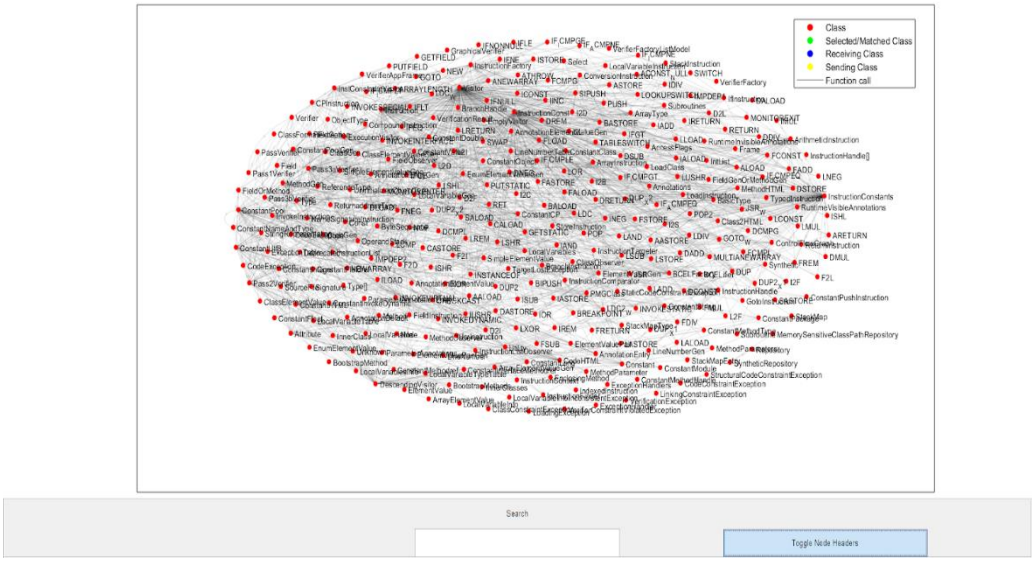
Det finns även en sökfunktion där alla klasser som innehåller sökordet markeras och klassnamnet visas (Figur 16). Värt att nämna är att sökfunktionen inte tar hänsyn till gemener och versaler, utan omvandlar både söksträngen och namnen på noderna (klasserna) till teckensträngar bestående av enbart gemener.



Figur 16: Programmets sökfunktion har använts med sökordet ”get”.

En grafs storlek är en viktig aspekt för hur tydlig visualiseringen av den är [2]. Genom att skapa ett interaktivt användargränssnitt där noder markeras på detta vis kan användaren fokusera på de delar av grafen som är relevanta.

Man kan även enkelt välja att visa alla klassers namn vilket även det är ett exempel på hur lätt det blir rörigt med ett stort dataset (Figur 17).



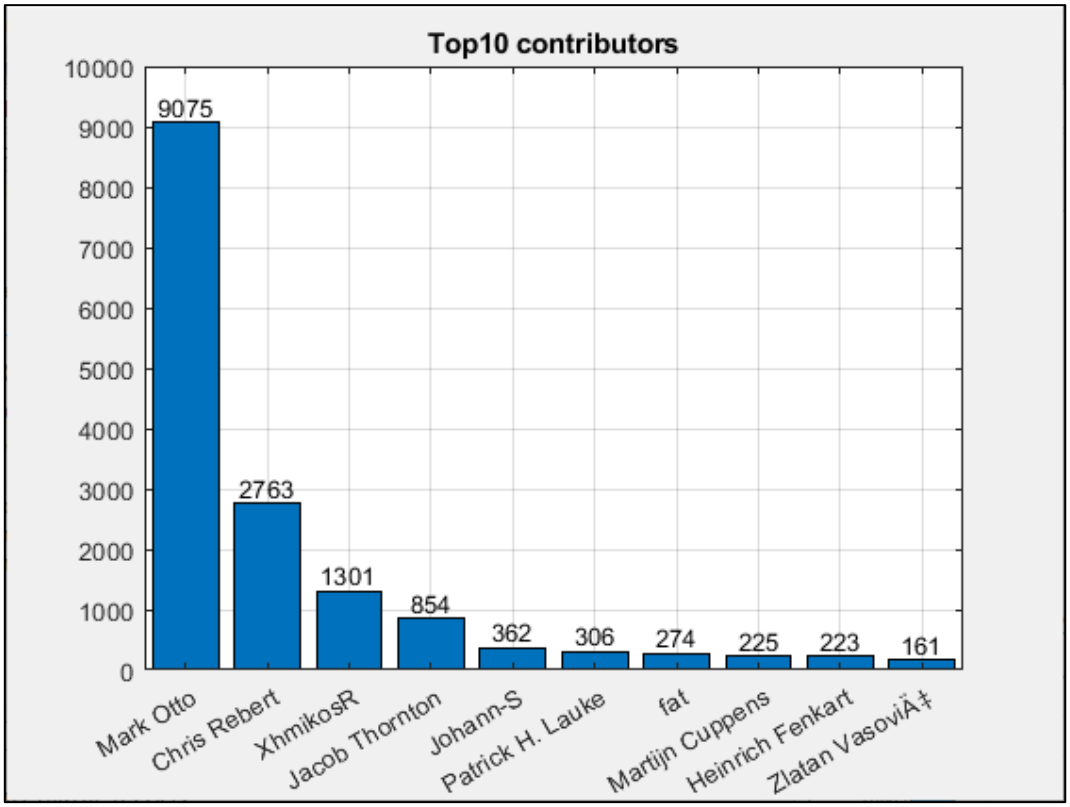
Figur 17: Alla klassnamn visas i grafen.

3 Resultat

3.1 Bidragshistorik till Git-repository

3.1.1 Diagram över de bidragsgivare som bidragit mest

Diagrammet som producerats visas i Figur 18.



Figur 18: Diagram över de bidragsgivare som bidragit mest.

Då domänen består av kategoriska data och kodomänen endast har en dimension (antalet bidrag) fanns få alternativ för val av visualisering. Ett alternativ hade varit att markera ut magnituden med punkter istället för staplar. Stapeldiagrammet valdes för att det ansågs vara tydligare.

3.1.2 Utskrift av antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare

I programmet skrivs antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare ut (Figur 19).

```
Number of contributors: 1276
Number of commits: 19057
Average commits per contributor: 14.935
```

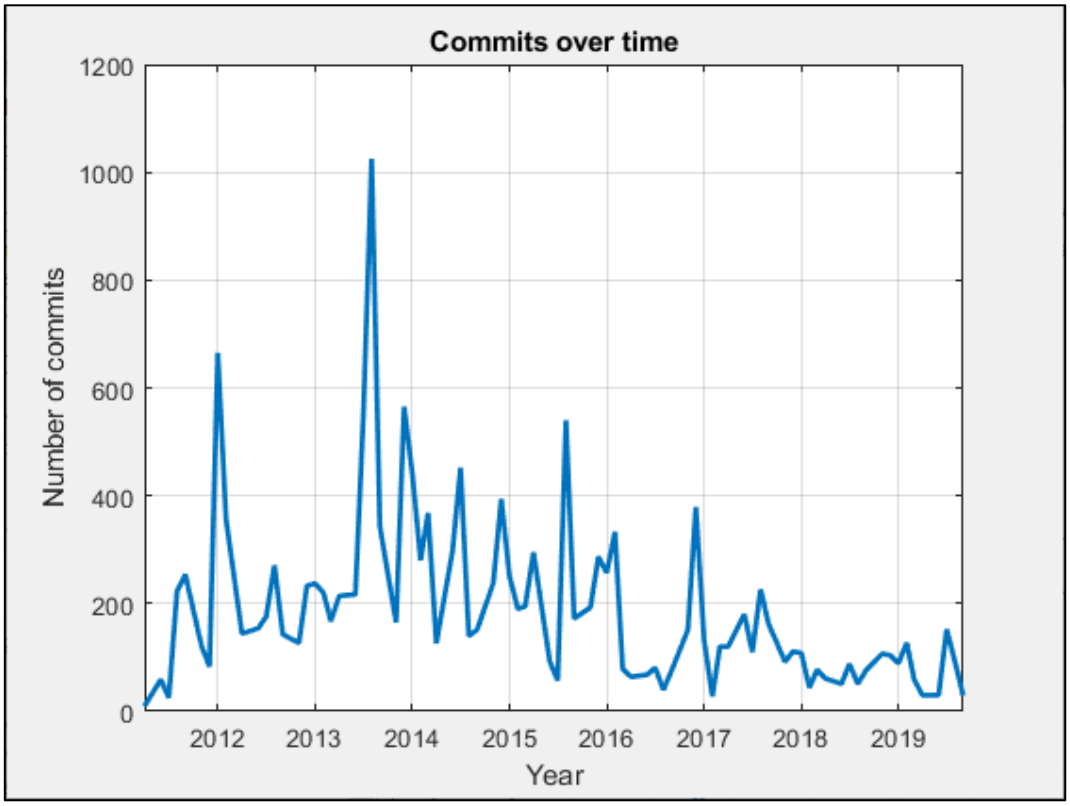
Figur 19: Utskrift av antalet bidragsgivare, antalet bidrag och bidrag per bidragsgivare.

Ett alternativ till att skriva ut medelvärdet (bidrag per bidragsgivare) hade varit att använda ett lådagram för att visa medianvärdet. Denna visualisering hade dock innehållit data utöver medianen (minimum, maximum, kvartilavstånd, etc.). Dessutom förmedlar medianvärdet och medelvärdet inte samma information. Dessa skillnader innebär att de två visualiseringarna är inte helt jämförbara.

En annan möjlig relaterad visualisering är cirkeldiagram för att visa hur stor del av bidragen som gjorts av vissa individer.

3.1.3 Visualisering av antalet bidrag över tid

För att visualisera antalet bidrag över tid användes ett linjediagram (Figur 20). Linjediagrammet var den enda visualiseringstekniken som ansågs lämplig.



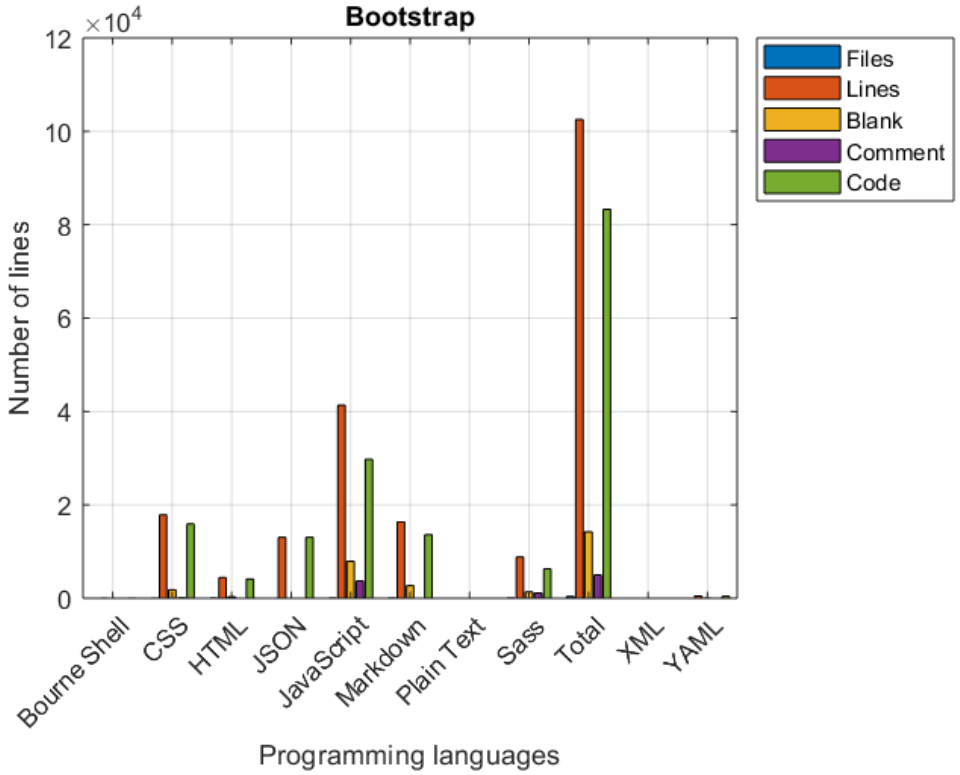
Figur 20: Linjediagram som visar antalet bidrag över tid.

3.2 Tillämpande av LOC på olika kodbaser

3.2.1 Diagram över ett ramverks beståndsdelar (Programmeringsspråk)

Ett av de framställda diagrammen visas i Figur 21. Längs x-axeln i har vi kategoriska data som anger vilket programmeringsspråk som det tillhörande värdet på y-axeln avser. Värt att notera är att "Total" inte är ett programmeringsspråk, utan en summering för den totala mängden av respektive kategori som finns i teckenförklaringen.

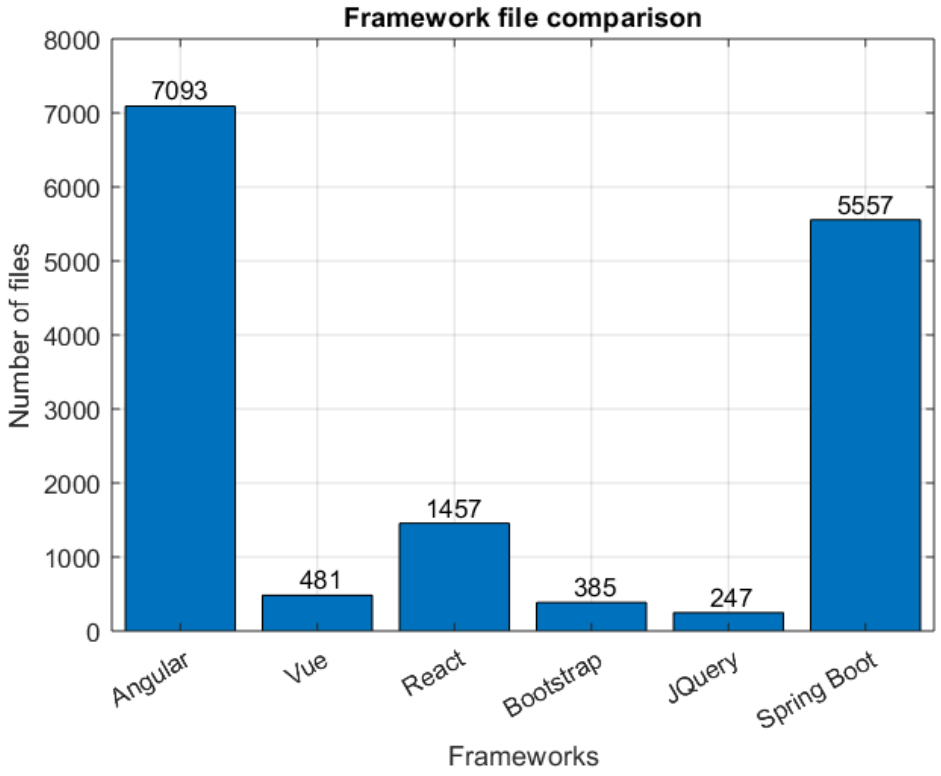
Som man ser i figuren så är storleken av en del datamängder betydligt mindre relativt de andra, detta leder till att det blir nästintill omöjligt att avläsa de mindre värdena på y-axeln. Under diskussionsdelen av denna rapport ges exempel på hur dessa mindre värden hade kunnat visualiserats på ett bättre sätt.



Figur 21: Diagram över vilka programmeringsspråk som har använts i källkoden.

3.2.2 Diagram över ramverkens beståndsdelar (Antal filer)

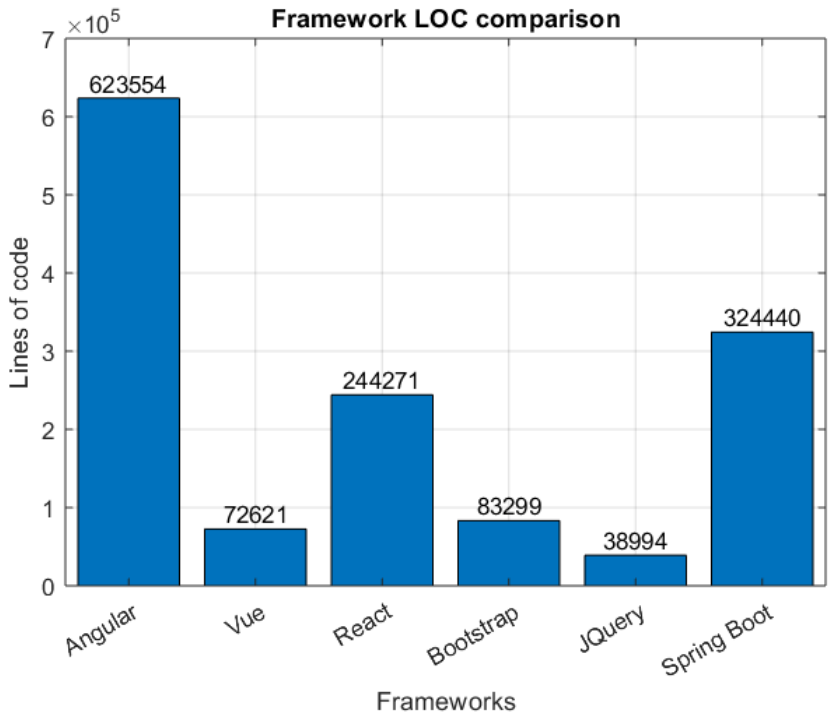
Nedan i Figur 22 har vi kategoriska data längs x-axeln som anger ramverket, längs y-axeln samt ovanför respektive stapel i diagrammet kan vi utläsa antalet filer som ramverket innehar.



Figur 22: Stapeldiagram som visar antalet filer i flera olika webbutvecklingsramverk.

3.2.3 Diagram över ramverkens beståndsdelar (Antal rader kod)

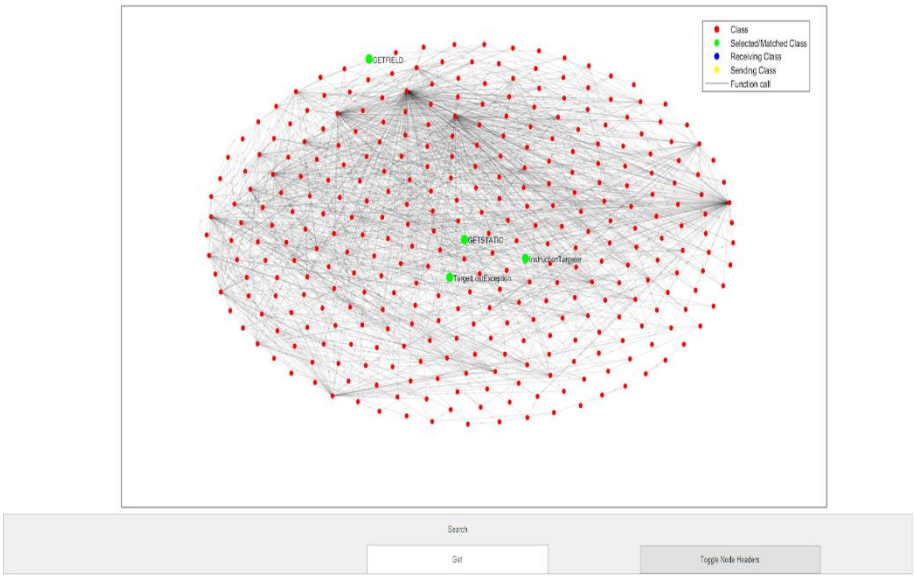
I Figur 23 har vi återigen kategoriska data längs-axeln som anger ramverket, längs y-axeln samt ovanför varje stapel i diagrammet kan vi utläsa antal rader av kod som ramverket består av.



Figur 23: Stapeldiagram som visar antalet rader kod i flera olika webbutvecklingsramverk.

3.3 Riktade grafer

Den visualisering vi ansåg bäst visade kopplingar mellan klasser är denna cirkel (Figur 24) i 2D som trots sina korsade linjer ger en bra översiktsbild och tillsammans med sök och markeringsfunktioner så är det lätt att hitta specifik information och se direkta och indirekta samband.



Figur 24: Den slutliga visualiseringen för en anropsgraf.

En anropsmatris kan vara ett kraftfullt verktyg för att snabbt se hur många och vilka kopplingar som finns, men tyvärr ej en passande visualisering för detta projekt. Något som kan diskuteras är att dela upp klasserna i olika grupper och visualisera de för sig men det kan då vara lätt att förlora förståelsen för helheten.

4 Diskussion

4.1 Bidragshistorik till Git-repository

Arbetet resulterade i en tydlig och överskådlig visualisering av bidragshistoriken. Programmet hade kunnat designats för att vara dynamiskt, så att det kan användas för att visualisera bidragshistorik från vilket Git-repository som helst. För att programmet ska fungera krävs att textfilen som genererats med Git Bash manipuleras manuellt innan inläsning till Matlab sker. Matlab-koden skulle också behöva utökas så att x-axeln (tiden) i linjediagrammet anpassas dynamiskt efter det dataset som används. Om källkod som har utvecklats i mindre än ett år hade det varit lämpligt att visa antalet bidrag per månad eller kanske till och med per dag.

4.2 Tillämpande av LOC på olika kodbaser

För de kodbaser som består av flera olika programmeringsspråk, och att storleken av ingående språken har stor differens finns det stor förbättringspotential. Den intuitiva avläsning av diagrammet blir näst intill omöjlig då diagrammets staplar har stor variation i deras värden. Ett bra alternativ för att bibehålla en intuitiv avläsning av diagrammet, är att separera mycket stora och små värden i olika diagram.

Ett annat alternativ vore att kompensera med någon form av interaktivitet, till exempel zoomning eller visning av stapelns värde när man håller muspekaren över den.

4.3 Riktade grafer

Det är ett stort problem att få en tydlig visualisering på kopplingar mellan en stor mängd klasser, men med hjälp av funktioner som gör att användaren kan söka på specifika klasser och se antal in- och utanrop samt se vilka klasser som den kopplar till gör det till en bra visualisering.

5 Referenser

[1] H. Purchase, “Which Aesthetic Has the Greatest Effect on Human Understanding?,” presented at the 5th International Symposium on Graph Drawing, Rome, Italy, Sep. 28-20 1997, pp. 248–261.

[2] I. Herman, G. Melançon, and M. S. Marshall, “Graph visualization and navigation in information visualization: a survey,” *IEEE Trans. Vis. Comput. Graph.*, vol. 6, no. 1, pp. 24–43, 2000.