DBL HTI + Webtech (2IOA0)
2019 Q4 - Project Guide

Assistant Professor Dr. rer. nat. Dipl. Inform. Michael Burch

Eindhoven University of Technology
Department of Mathematics and Computer Science
Algorithms and Visualization W&I

Abstract. The general goal of this DBL course is to collaborate in a team to solve a general visualization problem which is the interactive visualization of graph/network data on the web. This kind of data appears in many application fields like software engineering, social networking, biology, eye tracking, and the like. Understanding and exploring the data becomes a tedious task if it is not supported by interactive visualization techniques. Moreover, it is difficult to find and access tools that are able to support analysts focusing on graph/network data which makes web-based tools a suitable alternative to the standard tools. In this project, students should experiment with graphs and networks of different scales and should build such an interactive tool providing adjacency matrix and node-link views on the data in a linked way. The adjacency matrices should be reorderable in different ways while the node-link diagrams should be changeable by applying different layout algorithms. Moreover, people from all over the world should be able to access the interactive visualization tool with the goal to easily upload their own data, intuitively work with the visualizations, and finally, explore their data. Moreover, everybody can easily access all others uploaded datasets and also all found (algorithmic or visual) insights in the data. The students should illustrate the usefulness of their work by applying it to different kinds of datasets from varying sizes and complexities and present the final outcome in a written report and a video presentation.

1 Introduction and General Project Goals

The goal of this design-based learning (DBL) project is to learn about three different aspects and to combine them:

- **Information visualization** produces (interactive) visual representations of abstract data to reinforce human cognition, thus enabling the viewer to gain knowledge about the internal structure of the data and causal relationships in it. Information visualization (InfoVis) is the communication of abstract data through the use of interactive visual interfaces [2].
- **Human-computer interaction** is a diverse scientific and applied field where the focus is on how people use computers and how computers can

be designed to help people use them more effectively. The subject is studied from a wide variety of perspectives from many diverse areas, including (among others): computer science, psychology, ergonomics, information sciences, graphic design, or sociology [1].

– **Web-based applications** are clientserver computer programs in which the client (including the user interface and client-side logic) runs in a web browser [3]. Common web applications include webmail, online retail sales, online auctions, wikis, instant messaging services, and many other functions.

During the project you should also practice solving algorithmic issues, implementing, and experimenting with certain domain-specific datasets. Another aim of the project is to practice or improve several professional skills, such as working effectively in a team, planning, technical and scientific writing, and finally, presenting the achieved results. To reach all of those goals, the students work together in teams on an interactive web-based visualization problem. For this, they have to develop software, meet several times every week, and present their work in written reports and a presentation video.

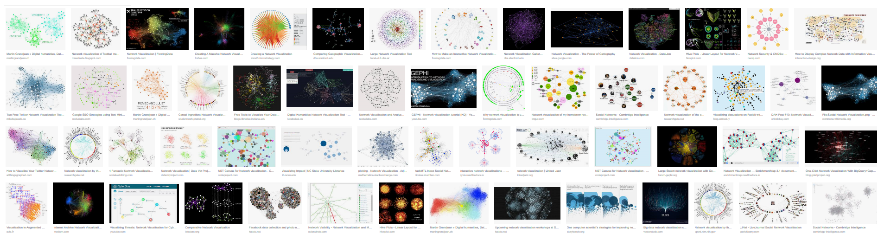

Fig. 1. A repertoire of graph/network visualizations, mainly based on node-link diagrams. You can use Google image search for 'network visualization' to get an impression about possible visualization candidates and to get inspired by the variety of different possibilities.

This project guide discusses how the project is organized and the various deliverables (documents, software, presentations, videos, and so on) on which the grade will be based. Please also carefully read the course web page at

https://canvas.tue.nl/courses/6672

which contains links to a document describing the individual steps focusing on visualization, interaction, and web-based problems to be solved during the project. Also other relevant documents, for example for the video generation, data format, dataset scenarios and the like, can be found here and should be carefully read. The webpage also contains a detailed schedule with all important dates, times, and locations.

The general goal of the visualization project is to visualize graph/network data (see Figure 1 for examples of node-link diagrams). This kind of data is omnipresent and builds basic instruments for information visualizations. For example, the friend relationships on Facebook, the call relations in a software system, or the co-author network in academic research build examples of graphs and networks. In some cases such a graph is undirected, directed, unweighted, weighted, multivariate, contains additional textual information for the vertices (labels) or not, and so on. A graph can have many properties worth investigating and encoding in an interactive visualization. First bringing structure into such a graph dataset is an important task, for example, solvable by reordering or clustering of the corresponding adjacency matrix. Moreover, visualizing such a graph in an intuitive and understandable way can be done by advanced layout algorithms applied to the node-link diagrams.

Visualizing such graphs in only one representation can hide information that would be visible in a different representation, hence the need for adjacency matrices and node-link diagrams in combination. For this reason, it is necessary to show the graph data in several representations that are linked (see for example Figure 2). Interactions should be applicable to change the views and to see the data from different perspectives. For example, selecting and highlighting one element should force a highlight in all the other views.

We recommend to have a look on the web by using Google and searching for 'network visualization' and images (Figure 1 shows an example screenshot). From there you can get inspired by the repertoire of visualization candidates and to get an idea about existing graph visualizations, but **it is not forbidden to be creative and design novel graph visualizations**. For simplicity reasons, all of the graph datasets will be provided in a comma-separated values format given in the form of a weighted adjacency matrix with textual descriptions (labels) (see the Files folder in Canvas for example datasets). The goal of this project is to design a web interface that allows everybody to upload a graph dataset in this specific format. The visualization tool handles this dataset and provides an interactive tool composed of at least two views, an adjacency matrix and a node-link diagram. The uploader can then visually interact with his/her own data to find insights in it.

2 Execution of the Project

There are some major points to take into account concerning teams and team building, planning and implementing, as well as guidance and support. Please always remember those points during the project and check if you have successfully solved them. It may be noted that the conditions might change during the project, e.g., if a team member leaves or is not able to work. In such cases please contact your tutor.

2.1 Teams and Team Building

The project will be executed by teams of six to seven students. We will make the team building based on programming skills. This is important for the group

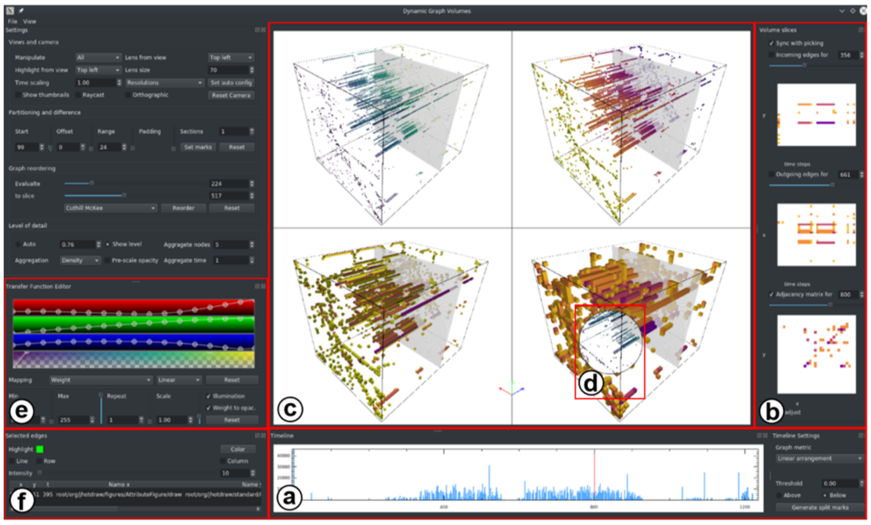

Fig. 2. An example visualization tool (for a different visualization topic) showing many parameter options, a central view that is split into subviews, and additional views for details. All views are linked and interactive.

work and for learning to initially start working in an unknown environment. If there are students left after the team assignment phase, those will be added as an extra team member to some of the teams. Each team should meet as often as necessary, to make sure that all team members are aware of how the project is progressing and which next steps are to do. Moreover, discussions about the current solutions, problems, challenges, design flaws, and ways to tackle these issues should be done. It is also important to divide and distribute the work among the team members. This concept is denoted as scrumming [4] in software development. To facilitate the team work, various project rooms are available, for more information please check the course webpage.

Each team should also select a **team leader/scrum master**, i.e., a person who is responsible for all interactions with the tutor, for material uploads, or for guiding a discussion in the team. This is important to avoid conflicts, i.e., that for example the same question gets asked multiple times by the same group and the answer might be understood differently by all team members. For sure, in case someone has individual questions do not hesitate to ask! The tutors are available for certain time periods which should be told them in advance.

2.2 Planning and Implementing

There are several deliverables and deadlines. Have a look at the webpage and the subsection below. The deadlines are strict, make sure you plan your work carefully and leave enough time in your schedule to be able to cope with unexpected problems (including team members becoming ill). Distribute the work carefully, by defining tasks and subtasks that can be done independently. Note that there are many tasks besides designing and implementing interactive visualizations: datasets and the data handling have to be tested, visualizations have to be understood, literature has to be read, intermediate and final reports have to be written, the presentation has to be designed, and the video has to be recorded (just to mention a few points). Even though such tasks are possibly executed by a subset of students from the team, the whole team is still responsible for the project in its entirety. It is also mandatory that the team leader annotates the written texts and implemented code with the responsible names of the team members. Those names will be taken into account for the final grading!

- **Project Planning:** At the end of the first week you need to make a project planning, which gives a breakdown of the project into tasks and subtasks, and the team member(s) responsible for each task and subtask. The project planning should be updated at the end of each week, adapting it where necessary and making it more detailed and enhanced where possible. The initial project planning as well as each of the updated versions should be sent by the team leader to the tutor (in pdf format) at the end of each week (Friday 6 pm at the latest).
- **Individual Planning and Time Sheets:** Each student has to maintain a time sheet in which s(he) keeps track of how much time (s)he spends on which task and subtask. This information can be used to make a realistic estimate of whether the project is on track with respect to the project planning, and it can help planning future (sub)tasks. As for the required level of detail, you can use the following rule of thumb: put in at least three entries per working day on average. Time sheets should be submitted (in pdf format) to the tutor at the end of each week (Friday 6 pm at the latest).

2.3 Guidance and Support

Each group is guided by a tutor (a student assistant), and any questions the team may have can be discussed with the tutor. General questions or hand-ins should be done by the team leader only (to avoid conflicts and misinterpretations or several interpretations). Only in exceptional cases, problems should be discussed with the project coordinator or the supervisors who are responsible for coordinating the tutors. This means the DBL is managed by a hierarchy of people in order to accelerate the communication. Which tutor guides which team is indicated on the web page. Please find out the corresponding names and email addresses to discuss the meeting schedules with your tutor.

- **The role of the tutor:** The role of the tutor is to observe how the team is functioning, to help the team in evaluating how the team is functioning and how this can be improved, to provide feedback to the team, to inform the supervisors and project coordinator about special questions and requests from the team, and to inform the project coordinator about serious problems encountered by the team.

The tutors have to guide the group process, make sure that the group is on the right track and they are the first person to contact for questions from the group. If they cannot answer it, they will address the supervisor. In the group meetings we will adapt an agile way of working, namely scrumming. It cannot be expected that the tutor is an expert on information visualization, human-computer interaction, or web-based technologies. However, if desired, the tutor may help the team in getting an overview of the advantages and disadvantages of certain approaches, and the tutor may refer the team to experts or literature that could provide answers to certain questions.

– **Meetings with the tutor:** Each team should have a meeting with the tutor twice per week for half an hour (one of which is a stand-up meeting). It is the responsibility of the team to coordinate the time and place of these meetings with the tutor. All students should be present at the meetings with the tutor. For each meeting with the tutor there should be an agenda (which is likely to be very similar for most meetings), and for each meeting minutes should be made. The minutes should at least contain the following information:
  - who was present at the meeting, and who was absent and why
  - a to-do list with things that need to be done in the coming week(s), including for every item on the list: who is responsible for it, and when it should be done
  - an evaluation of the to-do list from the previous meeting, with for every item on the list: whether it has been done, and if not, why not
  - any major decisions that have been taken

  The minutes (in pdf format) should be sent to the tutor and all team members, as quickly as possible after the meeting (not later than Friday in this week). The minutes will not be graded but proper minutes are required for passing the course.

3 Deliverables and Grading

We will provide example datasets in Canvas that are already pretty large. For experimental purposes these dataset are sufficient but it is mandatory that the students look for other dataset scenarios in order to test their tool for scalability issues. This means, it should be checked for which dataset size the visualization tool is still applicable and for which ones not. The datasets, implementation details and the final software product, the interim and final reports, the presentations as video will have an impact on the final grades.

3.1 Composition of Grades

Grades are determined as follows. First, each team gets a grade. Next, each individual team member's grade is determined by adding a modifier to the team's grade which depends on the parts the student has worked on. This should be indicated by the team leader.

Team Grade: The grade for each team is calculated as a weighted average of grades for the following four deliverables (see below for more details on these deliverables):

– Interim report: 10 % of the grade
– Web application/software product: 35 % of the grade
– Final presentation as video: 10 % of the grade
– Final report: 45 % of the grade

If a team scores less than 5 for the report, then the total team grade will be limited to at most 5, even if the weighted average is higher.

Individual Modifiers: It may happen that some team members contribute much more than others. To take this into account, each student will be asked to give a grade to each member of his/her team. This happens anonymously. The grades given by each team member can influence the final grade for each student positively or negatively. Moreover, we take into account the texts and source code your name is placed. Be aware that every group member should equally well participate in the group discussions, implementations, and text writing. Also the video presentation should be prepared by all group members and everybody should talk and present for a while in the video.

Conditions for Receiving a Grade: There are certain deliverables that are not graded, for example time sheets, minutes of the meetings with the tutor, and prototype submissions. Although these deliverables are not graded, they are still required. If you do not do them properly or hand them in late, you may be excluded from the course or receive a substantially lower grade.

3.2 Interim and Final Report (55 percent)

Each team has to write a report about the project and the visualization tool. This report should contain, among other things, a description of the visualizations that have been implemented with figures showing the visualizations in several parameter settings, an explanation of the interactions and their impact on the visualizations, and the web-based aspects reflecting how the visualization tool can be used online. Adjacency matrix visualizations and node-link diagrams should be compared while at least one interaction technique from each of the 7 categories given by Yi et al. [5] should be implemented for each of those. The course webpage has a link to a separate document that contains more information on what the report should contain and how it will be graded. The final report counts for 45 percent of the final grade, so make sure you spend enough time to write a clear, complete, and polished report. The final report should have a length of 8 pages on which at least 6 pages are text and the other ones can be filled with figures. We will provide a LaTeX template for achieving similar conditions for every group.

Roughly halfway the project, each team has to submit an interim report. This report should contain a preliminary version of certain sections of the final report, namely the introduction (which also describes the related work from the literature), the section with the description of the visualization techniques, their design, and interaction techniques. Moreover, the references are important for each scientific research report. See also the separate document on the report that is accessible from the course web page. You are expected to incorporate the feedback you receive on your intermediate report into the final report. The interim report should have a length of 4 pages in our specific template while at least 3 pages must be filled with text.

Both the interim report and the final report should be submitted (in pdf format and as a zip file with LaTeX sources and figures) through Canvas and should be sent to the tutor. The course webpage contains the exact submission deadlines.

3.3 Software and Final Submission (35 percent)

The problem description (available from the course webpage) explains the requirements on the software you have to produce during this course. In addition to the final submission, you have to submit one prototype at the end of the third week. This is not only for your own good, but also for that of the other participants. Make sure you include the submission of the prototype in your project planning, including a description of what you plan to include in the prototype.

The minimum requirement for the prototype is that it can read data input and produce at least one visualization technique from the dataset. The visualizations do not need interactions but it should show a static picture of the dataset. This prototype enables us to make sure that you understand the input data and the visualization techniques. Note that you are not allowed to use other people's code (not from within the course and not from outside the course).

Deadlines can be found in the schedule of the course webpage. The grade for your software is based only on the final submission. The prototype is not graded (but you are nevertheless required to submit it). The grade for the final submission is determined by how well it performs on a number of test datasets. The main criteria are the quality of the visualizations and interactions, the included algorithmic approaches like matrix reordering/clustering and graph layouts, the design, and how well it is applicable over the web. We may also consider creativity or novel visualization and interaction ideas. Software design (structure and clarity of the code) is not taken into account.

3.4 Presentation and Video (10 percent)

Each team will prepare a presentation about the final visualization tool. To prepare the presentation, each team designs a presentation, including the loading of several datasets of varying sizes and complexities, their visualization in different visual metaphors (adjacency matrix and node-link diagrams), and the interactions that are implemented in the tool (also showing reordering techniques and different node-link layouts). It may be noted that the tool should work online, i.e., typically in a web browser. The presentation should be first discussed with the tutor.

We will provide literature and tutorials on how to generate a good presentation and how a well-designed video can be produced. A presentation video of about 5 to 6 minutes should be recorded in which each of the team members speaks about 45 to 60 seconds. All of the individual team member parts will end up in a complete description of the runnable software. Do not forget to tell a story, i.e., which data, visualizations, interaction, and web technologies are used and which explicit tasks are reported. The video should be produced in a way as if it presents a scientific presentation on a certain general topic, but with a high degree of agility (making some fun or jokes is not forbidden as long as it follows the rules!). The videos should be uploaded before the deadline.

Your video presentations will be graded based on content and on some aspects of presentation. In particular, we will pay attention to **Agility, Understandability, Presentation, Content, and Story!** and also if you reacted on possible questions to clarify misunderstandings. Those aspects can be described in more detail:

- Clarity of the structure (main points, transitions between topics, etc.) (Understandability, Presentation)
- Technical depth, focus and balance (focusing on the interesting parts without making it too complicated, critical discussion of experimental results, etc.) (Content, Story)
- Technical clarity and precision (Understandability, Content)
- Your answers to questions asked later (after tutors, supervisors, and project coordinator having watched the video) (Understandability)
- Use of visual aids (text, pseudocode, charts, figures on the slides) (Presentation, Content, Understandability)
- Graphic design of the visual aids (clarity, legibility, appeal) (Presentation, Understandability)
- Timing and flow (not too short, not too long, not losing the thread, etc.) (Presentation, Story, Agility)

3.5 Plenary Meetings and Deadlines

There are two mandatory plenary meetings (that is, meetings where all students from every team should be present): a kick-off meeting and a final meeting (where the 10 best videos are shown and the best one is awarded a price). The exact time and place of these meetings can be found on the course webpage. The webpage also contains the deadlines for the various deliverables.

4 Using Scrum to Develop Professional Job Skills

Using scrum, the DBL project period is split into 4 periods (timeboxes) of 2 weeks called sprints. At the beginning of each sprint, a planning session is organized to discuss and define scope and activities for that sprint (the backlog) with the help of the tutor. At the end of each sprint a product, i.e. a piece of working software, must be demonstrated to the tutor (in the product review).

Based on the input of the customer (represented by the tutor) and a reflection on the process by the team itself (the sprint retrospective), extensions and changes are defined to be accomplished in the next sprint. Each day (or as often in a week as the team works on the project), the team has a short stand-up meeting (daily scrums) in which each team member accounts for the work of the previous day, the intended work of today, and what impediments may prevent that outcome.

The scrum master is responsible for making sure that all of the above meetings occur, according to the scrum principles. The team is responsible for making these meetings productive. The team is supposed to be situated together for optimal communication, including a large display (backlog and task board) presenting the up-to-date status of the product development. In this setting the following skills are practiced in a natural way:

- **Presenting:** Each team member has to present his/her work of the previous day and his/her intended work to the other team members. Also, depending on the role in the team, various demos have to be presented.
- **Cooperation:** Scrum enforces that each team member feels responsible for and contributes optimally to the final product; free-riding is almost impossible because of the daily scrums and visible task board. Self-organization of the team is stimulated by design, e.g. in the sprint retrospectives the process is discussed, individual feedback is given and improvement points formulated.
- **Reflection:** Each team member has to reflect on his/her results of the previous day in the daily scrum, and on both the group process and individual role in the sprint retrospective.
- **Planning and organizing:** Scrum prescribes a group planning for each sprint, and a group and individual plan for each day. Scrum helps to actually start open-ended projects by working with short periods and requiring a demo at the end of each sprint, even if the requirements are unclear. This stimulates self-organization of the team and of its members.
- **Professional job skills learning objectives:** At the end of the course, each student should be able to
  - write a report as a scientific paper.
  - adequately present their own results and intentions in scrum meetings.
  - effectively participate in daily scrums, sprint planning sessions, product reviews, and sprint retrospectives, taking responsibility for the allocated tasks.
  - evaluate one's own task in daily scrums and the team process in sprint retrospectives.

- plan and adapt one's own tasks per day and know how to work with the scrum task board.
- use current methods and tools for organizing software engineering projects.
- find and evaluate various sources of information and to adequately refer to them in a scientific report.

References

1. Card, S.K., Moran, T.P., Newell, A.: The psychology of human-computer interaction. Erlbaum (1983)
2. Keim, D.A., Mansmann, F., Schneidewind, J., Ziegler, H.: Challenges in visual data analysis. In: 10th International Conference on Information Visualisation, IV. pp. 9–16 (2006)
3. Nations, D.: Web applications (2014)
4. Schwaber, K.: SCRUM development process. In: Proceedings of ACM SIGPLAN on Objected-Oriented Programming, Systems, Languages, and Applications (OOPSLA 96) (1996)
5. Yi, J.S., ah Kang, Y., Stasko, J.T., Jacko, J.A.: Toward a deeper understanding of the role of interaction in information visualization. IEEE Transactions on Visualization and Computer Graphics 13(6), 1224–1231 (2007)