Prüfungsprotokoll InfoVis WS17/18

Hinweise:

Entspannte Atmosphäre. Es wurde häufiger nachgefragt, wenn ich etwas genauer erklären sollte. Ansonsten konnte ich recht frei über ein Themengebiet reden und so auch das Gespräch auf das lenken, was ich halbwegs wusste. Die Endnote war sehr gut. Kritik war, dass ich als Informatiker manche Sachen eigentlich wissen sollte und die Antworten etwas zögerlich waren (wusste nicht immer sofort, worauf er genau hinauswollte). Entsprechend hat er wohl nicht so viele Themen durchbringen können wie gewollt.

Vorbereitung:

In der Vorlesung war ich nur zu Anfang (Montag 8:00 Uhr…). Ich habe etwa zwei Wochen vorher mit dem lernen angefangen und bin zu Anfang etwa jeden Tag eine Vorlesung durchgegangen. Dabei hatte ich nicht jeden Tag Zeit zum Lernen. Etwa eine Woche vor der Prüfung war ich in der Fragestunde und habe mich danach auf die dort angesprochenen Themen konzentriert und bin diese immer wieder durchgegangen. 4 von 5 Übungen habe ich unter dem Semester gemacht, bin diese aber nur ganz kurz für die Prüfung durchgegangen. Es wurden aber auch zu Code Fragen gestellt, speziell zu Fruchterman-Reingold. Es ist definitiv sinnvoll, grob zu wissen, wie die Algorithmen aussehen und welche Laufzeiten diese haben.

Fragen:

F: Was sind denn Netzwerke?

A: Netzwerke sind Daten, die miteinander in Beziehung stehen. Wenn man Netzwerke untersucht, werden diese in Form von Graphen betrachtet. Graphen bestehen aus Nodes und Edges und haben eine gewisse Topologie.

F: Wie kann man denn Graphen darstellen?

A: Wir haben in der Vorlesung verschiedene Möglichkeiten durchgesprochen. Zum Beispiel das Circular Layout oder verschiedene Force Directed Ansätze.

F: Was gibt es über das Circular Layout zu sagen?

A: Habe angefangen, ein Circular Layout aufzumalen und zu erklären. Man kann dabei z.B. die Nodes nach Wichtigkeit sortieren.

F: Sie haben dort verschiedene Größen für die Nodes gemalt. Wieso?

A: Die einzelnen Nodes können verschieden Wichtig sein. Dies kann man anhand der Centrality messen. Wir haben in der Vorlesung verschiedene Ansätze dafür kennen gelernt: Degree, Closeness, Betweenness, Eigenvector und PageRank.

F: Erklären Sie mal.

A: Also die drei wichtigsten waren die Degree, Closeness und Betweenness Centrality. Degree Centrality priorisiert die Nodes mit vielen Kanten. Closeness ist das Inverse von Farness. Farness ist

die Summe der kürzesten Wege zu allen anderen Knoten (dabei kurz verheddert, ich habe vom Maximum geredet). Betweenness misst, wie oft ein Knoten in allen kürzesten Wegen vorkommt.

F: Und wie sieht es mit dem PageRank aus?

A: Kurz erklärt, dass dies Googles berühmter Algorithmus sei. Dann habe ich erzählt, dass dabei die Nodes ihre Wichtigkeit (geteilt durch die Anzahl der outgoing edges) an die umliegenden Nodes verteilen. Dies kam aber anscheinend nicht gut genug rüber, und es gab ein gewisses hin und her.

F: Wie kann man die Nodes noch anordnen (Circular Layout)?

A: Anhand der Communities, die man auch verschiedenfarbig markieren kann.

F: Was kann man noch bei einem Circular Layout ändern?

A: Farbe, Größe, Anordung. Wusste nicht worauf er genau hinaus wollte.

F: Gut egal. Weiter. Was gibt es noch für Layouts?

A: Wir haben verschiedene Force Directed Ansätze durchgesprochen. Zum Beispiel Fruchterman-Reingold. Solche Ansätze basieren auf dem Springfedermodell, indem anziehende und abstoßende Kräfte simuliert werden (alles kurz skizziert und die Gleichung für Fa und Fr vom Fruchterman-Reingold aufgeschrieben).

F: Und wie würde man das dann implementieren?

A: Da war ich nicht ganz drauf vorbereitet gewesen, wollte erst zwischen jedem Node die Distanz berechnen. Er wollte aber darauf hinaus, dass man für Fa über die Edges gehen kann und erst für Fr die Distanz von jedem Node zu jedem anderen braucht. Der Displacement-Vektor war ihm dabei auch wichtig, genauso wie die Laufzeit. Dabei auch kurz den Cooling-Factor erklärt.

F: Was gibt es noch für Force Directed Ansätze?

A: Kurz ForceAtlas und Kamada-Kawai genannt.

F: Was ist der wesentliche Unterschied zwischen diesen?

A: Kamada-Kawai berechnet die graphentheoretischen Distanzen (kürzester Pfad) zwischen den Nodes, während ForceAtlas nur die direkte Distanz nimmt.

F: Welcher Ansatz hat die längste Laufzeit?

A: Kamada-Kawai, da man für jeden Node die Distanz zu allen anderen berechnen muss (also n mal Dijkstra anwenden, welcher eine Laufzeit von n*log(n) hat).

F: Was gibt es noch Eigenschaften von Graphen?

A: Graphen können auch zeitabhängig sein.

F: Erklären Sie mal.

A: Echte Netzwerke sind häufig zeitabhängig. Es ist aber schwer, diese gut darzustellen. Bin dann kurz auf die Mental Map eingegangen und habe angefangen den Force Directed Ansatz zu erklären. Dabei bin ich vor allem auf das Node Aging eingegangen.

F: Gibt es das Node Aging auch für Kanten?

A: Ja, das wird aber normal nicht für das Layout verwendet, sondern nur um zeitliche Änderungen hervorzuheben.

F: Neben dem Aging gibt es noch einen einfacheren Ansatz. Welcher?

A: Da bin ich nicht draufgekommen, er wollte auf das Node Pinning hinaus.

F: Was gibt es noch für Möglichkeiten, um die Mental Map beizubehalten.

A: Man kann die zeitliche Reihenfolge z.B. auf einer Timeline nebeneinander darstellen. Bei größeren Netzwerken und kleineren Änderungen ist es dabei aber schwer, Änderungen zu visualisieren.

F: Was gibt es neben zeitabhängigen Graphen noch?

A: Graphen können auch eine Hierarchie beinhalten.

F: Wie erzeugt man so eine Hierarchie aus einem normalen Graphen?

A: Da bin ich auch nicht direkt draufgekommen, er wollte auf die Breitensuche hinaus.

F: Wo fängt man bei so einer Breitensuche an?

A: Normal beim root node.

F: Und wenn es keinen gibt? Dann könnte man z.B. den Node mit der höchsten Centrality nehmen. Was für einen wichtigen Aspekt gibt es bei Graphen noch?

A: Das Clustering (mir ist der Begriff von Communities nicht eingefallen). Diese kann man z.B. anhand der Modularität berechnen.

F: Was sind denn Communities?

A: Kurz anhand einer Skizze erklärt. Bin dabei auch darauf eingegangen, dass Nodes auch zu mehreren Communities gehören können.

F: Wie kann man einen Graphen auf Communities untersuchen?

A: Wir haben drei verschiedene Möglichkeiten angesprochen: lokal, global und mithilfe von Vertex Similiarity. Für die lokale Communitysuche bietet sich der Clustering Coeeficient (CC) an, der angibt, wie stark ein Node ein Star oder eine Clique ist. Habe diesen dann kurz erklärt.

F: Was ist denn eine Clique?

A: Kurz anhand einer Skizze erklärt. Dann sollte ich noch eine 3er, 4er und 5er Clique aufzeichnen.

F: Gut, wie waren die anderen Ansätze?

A: Global werden Communities anhand der Modularität berechnet. Für den Algorithmus kann man beispielsweise die Betweenness der Nodes berechnen und nach und nach Nodes entfernen. Dann habe ich kurz die Berechnung mithilfe der abgewandelten Breitensuche erklärt. Auch habe ich erwähnt, dass die globale Definition als Abweichung von einem zufälligen Graphen anzusehen ist.

F: Was sind denn zufällige Graphen?

A: Kurz das klassische Modell und Erdös-Renyi erklärt.

F: Wie berechnet man denn die Modularität – gesehen, dass die Zeit um ist – ok dann war es das jetzt auch.