

FACHHOCHSCHULE SÜDWESTFALEN

# **Verflechtungsanalyse des Transparenzregisters**

Tim Ronneburg

Prof Dr. Doga Arinir

6. Oktober 2023

# Inhaltsverzeichnis

<b>1</b>	<b>Einleitung</b>	<b>1</b>
1.1	Problemstellung . . . . .	1
1.2	Zielsetzung und Aufbau der Arbeit . . . . .	1
<b>2</b>	<b>Graphentheorie</b>	<b>2</b>
2.1	Begriffliche Definition . . . . .	2
2.2	Sociogram/ Social Network/ Social Graph . . . . .	3
2.3	Social Network Analysis (SNA) . . . . .	3
2.3.1	Element-Level Metriken . . . . .	4
2.3.2	Group-Level Metriken . . . . .	6
2.3.3	Network-Level Metriken . . . . .	7
<b>3</b>	<b>Ein Social Graph für das Transparenzregister</b>	<b>7</b>
3.1	Aufbau des Social Graph/ SNA und Prämissen . . . . .	8
3.2	Anwendung der Social Network Analysis (SNA) . . . . .	11
3.2.1	Erstellen eines Graphen mit der Element-Level Metriken . . . . .	11
3.2.2	Berechnung, Aufbau und Interpretationsmöglichkeiten der Group-Level Metriken . . . . .	15
3.2.3	Berechnung, Aufbau und Interpretationsmöglichkeiten der Network-Level Metriken . . . . .	16
3.3	Weitere Analysen . . . . .	16
3.4	Perspektive auf einen Graphen mit Unternehmens und Personen Daten . .	17
3.5	Handlungsempfehlung . . . . .	18
<b>4</b>	<b>Zusammenfassung</b>	<b>19</b>
4.1	Kritische Reflexion . . . . .	19
4.2	Fazit . . . . .	20
4.3	Ausblick . . . . .	20
<b>5</b>	<b>Literatur</b>	<b>22</b>

# 1 Einleitung

In den letzten Jahren mehren sich die großen Wirtschaftsskandale, in denen Unternehmen Bilanzen fälschen oder untereinander zusammenarbeiten, um Steuerschlupflöcher auszunutzen, wie bei Wirecard oder der Cum-Ex Affäre. Ein wichtiger Bestandteil für das Erkennen oder Aufarbeiten solcher Skandale besteht darin, sich einen Überblick über die Verflechtungen der beteiligten Akteure zu verschaffen. Diese Arbeit resultiert aus einem Projekt, welches genau solche Verflechtungen transparent darzustellen versucht. Bei dem sogenannten Transparenzregister werden die Beziehungen von Unternehmen und einzelnen natürlichen Personen wie Wirtschaftsprüfer, Kommanditisten etc. aus Deutschland und der Europäischen Union dargelegt.

## 1.1 Problemstellung

Die angesprochenen Verflechtungen sind teilweise komplexe Strukturen von Unternehmen und anderen Akteuren, die sich erst bei einer umfassenderen Betrachtung erkennen lassen. Die Analyse solcher Konstrukte ist aus den reinen Rohdaten, ohne aufwendiger Aufarbeitung und Visualisierung, nahezu unmöglich. Damit aus den Daten Informationen und Wissen gewonnen werden können, müssen diese aufbereitet und in einer verständlichen Form dargestellt werden. Spätestens seit der F8 von Facebook im Jahr 2007 ist die Nutzung von Graphen für solche Verflechtungen etabliert. Eine solche Verflechtung nennt man ein *Social Network*, *Social Graph* oder auch *Sociogram*. Mit diesem Graph können Analysen einfacher durchgeführt werden, da sowohl das Wissen aus der Graphentheorie zum Tragen kommt als auch die Beziehungen einzelner Akteure mit dem bloßen Auge ersichtlich sind.

## 1.2 Zielsetzung und Aufbau der Arbeit

Ziel dieser Arbeit ist die Vermittlung der Grundlagen für eine solche Verflechtungsanalyse. Es wird aufgezeigt, wie ein solcher Graph aufgebaut werden kann, welche Bedingungen gelten und welche Kennzahlen man berechnen kann.

Das Werk beginnt mit einer Einführung in die Graphentheorie, aus dieser leiten sich *Social Networks* beziehungsweise *Social Graphs* ab. Neben der Graphentheorie wird auch auf die Analyse von *Social Networks* eingegangen. Im Hauptteil werden diese Grundlagen auf das zugrunde liegende Projekt angewendet. Es wird verdeutlicht, wie ein solcher Graph für das Projekt aufgebaut wird und welche Bedingungen dafür erfüllt sein müssen. Des Weiteren werden Kennzahlen vorgestellt und gebildet mithilfe derer die Analyse einer solchen Verflechtung durchgeführt werden kann.

Abgeschlossen wird diese Arbeit mit einer Handlungsempfehlung für das Projekt sowie einem Fazit und Ausblick.

## 2 Graphentheorie

In diesem Abschnitt werden die Grundlagen der Graphentheorie erläutert. Mithilfe dieses Wissen lassen sich Verflechtungen besser verstehen und analysieren, da auch Verflechtungen Graphen sind.

### 2.1 Begriffliche Definition

Graphen sind nach der Graphentheorie “Strukturen aus Punkten und Verbindungen zwischen diesen Punkten” [1, S. 257]. Die Punkte werden als **Ecken/Knoten** oder im Englischen **Nodes** und die Verbindungen als **Kanten/Verbindungen**, oder im Englischen **Edges** bezeichnet. Dabei liegt der Kern eines Graphen nicht in der Visualisierung, sondern in dessen mengentheoretischen Eigenschaften. [1, S. 257]

Es gibt diverse Arten von Graphen: **Ungerichtete Graphen** und **gerichtete Graphen** beziehungsweise **Digraphen**. Für ungerichtete Graphen gilt folgende Definition:

“Ein ungerichteter Graph  $G$  ist ein Paar  $(V, E)$ . Hierbei ist  $V$  eine endliche Menge, welche die Ecken repräsentiert, und  $E$  ist eine Menge, die aus Mengen der Form  $v1, v2$  besteht, wobei  $v1, v2 \in V$  gilt.  $E$  repräsentiert die Menge der Kanten.” [1, S. 257]

Anhand dieser Definition lassen sich Graphen mit beliebig vielen Kanten zwischen den Ecken bilden. Die Kanten müssen dabei nicht geradlinig verlaufen, sodass derselbe Graph auf verschiedene Weisen dargestellt werden kann. [1, vgl. S. 257-258]

Im Gegensatz dazu besitzen gerichtete Graphen Kanten mit einer vorgegebenen Richtung. Diese Richtung wird anhand eines Pfeiles auf der Kante visualisiert. Auch ein gerichteter Graph kann auf verschiedene Arten dargestellt werden. Eine mögliche Definition von gerichteten Graphen ist die folgende:

“Ein gerichteter Graph oder Digraph  $D$  ist eine Struktur  $(V, E)$ . Hierbei ist  $V$  eine endliche Menge der Ecken und  $E$  ist eine Menge, die aus Paaren der Form  $(v1, v2)$  besteht, wobei  $v1, v2 \in V$  gilt.  $E$  repräsentiert die Menge der gerichteten Kanten, welche auch Bögen genannt werden.” [1, S. 258]

Die bereits angesprochene Möglichkeit, einen Graphen mit denselben Eigenschaften auf unterschiedlichste Weise darzustellen, bezeichnet man als **Isomorphie**. Es ist einfach von einem Graphen einen isomorphen Graphen zu erzeugen, aber deutlich komplexer die Isomorphie von zwei Graphen festzustellen. [2, vgl. S. 272]

## 2.2 Sociogram/ Social Network/ Social Graph

Ein Sociogram ist ein Model eines Netzwerks von sozialen Verbindungen die durch einen Graphen repräsentiert werden. Diese Idee wurde 2007 von Facebook als Social Graph in der F8 vorgestellt. Diese Art von Graph basiert auf der Graphentheorie. Die Stärken dieses Graphen liegen in der Veranschaulichung der sozialen Verflechtungen.

Ein solcher Graph oder ein solches Netz wird aufgebaut, indem jede Ecke des Graphen einen Akteur (Person oder Unternehmen), jede Kante eine Verbindung (Beauftragung, Verwandschaft, Arbeitsverhältnis) darstellt. Die Kanten können mit Gewichten versehen werden. Jede Kante ist dabei gerichtet.[3, vgl. S. 8]

Das Ergebnis kann als *Social Graph*, *Social Network* oder *Sociogram* bezeichnet werden. In dieser Arbeit wird hauptsächlich der Begriff *Social Network* (SN) genutzt.

Die kleinste Struktur in einem *Social Network* wird als *Dyad* bezeichnet und ist eine soziale Gruppe bestehend aus zwei Knoten mit einer gerichteten oder ungerichteten Kante. Die nächst größere Form ist eine *Triad*, welche offen oder geschlossen sein kann. Offen bedeutet, dass über einen Knoten die anderen beiden verbunden sind. Hingegen ist bei einer geschlossenen *Triad* jeder Knoten mit beiden anderen Knoten über eine Kante verbunden. Die größte soziale Gruppe stellt ein *Quad* dar und besteht aus vier Ecken.[4, S. 12-14]

Die Ansammlung von mehreren Akteuren durch enge Verbindungen wird als **Cluster** oder **Group** bezeichnet.

## 2.3 Social Network Analysis (SNA)

Bei der *Social Network Analysis* (SNA) werden die sozialen Strukturen anhand von Metriken aus der Graphentheorie untersucht.

Eine Analysemöglichkeit ist die Bestimmung der *number of hops*. Diese gibt an, wie viele Verbindungen benötigt werden, um von einem Punkt zu einem anderen zu gelangen. Dieser Wert kann auf einen Teil des Graphen sowie auf den gesamten Graphen gemittelt werden. Ist der Median der *number of hops* im Gesamtgraphen beispielsweise bei 5, so werden Verbindungen, die diesen Schwellwert überschreiten, zu einem Cluster kombiniert. [3, S. 9]

Weitere Einsichten werden über ein Netzwerk erlangt, in dem man Teile des Netzwerkes oder das gesamte Netzwerk in drei verschiedene Level abstrahiert. **Element-Level** ist die Betrachtung der Auswirkungen und Einflüsse einzelner Ecken und Kanten. **Group-Level** analysiert die Zusammenhänge und Dichte von Gruppen innerhalb des Netzes. **Network-Level** ist das Interesse an den topologischen Eigenschaften des Netzwerkes. [5, vgl.]

### 2.3.1 Element-Level Metriken

Die folgenden Metriken sind aus der Element-Level Analyse und betrachten die Bedeutung der einzelnen Knoten und Kanten.

Metriken zur Bedeutung von Verbindungen:

**Transitive** beschreibt die Menge an gleichen Kanten zweier Ecken, die über eine Kante verbunden sind. In einem sozialen Netzwerk für Personen gibt es die Wahrscheinlichkeit, dass zwei Bekannte einer Person sich anfreunden.

**Reciprocity** gibt die Wahrscheinlichkeit an, mit der sich eine Ecke mit sich selbst verbindet.

**Assortativity** drückt aus, wie sehr sich ein Akteur mit anderen Akteuren verbindet, die ähnlich sind hinsichtlich der Größe des Grades.

**Homophily** ist die Wahrscheinlichkeit von Verbindungen sehr ähnlicher Akteure untereinander. [5, vgl.]

Weitere Algorithmen und Kennzahlen sind die Folgenden:

**Degree Centrality** gibt die Anzahl der Kanten je Knoten an. Knoten mit einer hohen *Degree Centrality* haben die meisten Verbindungen und können einen hohen Einfluss aufweisen oder gut platziert sein. Es wird eingesetzt, um gut verbundene, beliebte, informationshaltende oder Reichweiten starke Akteure zu finden. Die Kennzahl kann bei *directed graphs* in in-degree (eingehende) und *out-degree* (ausgehende) Kanten aufgeteilt werden.[6, vgl. S. 168-169]

Ein Graph kann nach und nach immer weiter nach der *Degree Centrality* gefiltert werden. Dadurch erhält man die am besten verbundenen Ecken. Dieses Vorgehen bezeichnet man als *degenerate graph* oder *Degeneracy*.

Um diesen Wert zu berechnen benötigt es lediglich die ausgehenden Kanten an den jeweiligen Ecken zu berechnen. Um einen standardisierten Wert zu berechnen, nimmt man die Anzahl an Ecken je Graph ( $n$ ) und nutzt die Summe minus 1 ( $n-1$ ) als Teiler für den Wert der Kanten je Ecke. Wenn ein Graph 10 Ecken hat und eine Beispiel-Ecke 5 ausgehende Kanten, ergibt sich daraus eine *Degree Centrality* von 5 und ein standardisierter Wert von  $1/3$  ( $5/(10-1)$ ).

Im Zusammenhang des Transparenzregisters lassen sich mit diesen Kennzahlen gut vernetzte Unternehmen oder Akteure mit großer Reichweite ermitteln.

**Betweenness Centrality** hebt Knotenpunkte hervor, welche besonders oft als Verbindungsknoten zwischen zwei anderen Einheiten dienen. Sie werden als "Brücken" benutzt und können der kürzeste Pfad in einem Netzwerk sein. Mit dieser Kennzahl werden die Akteure gefunden, die den Fluss des Netzwerks am meisten beeinflussen. Bei der Interpretation dieser Kennzahl muss allerdings mit Vorsicht agiert werden.

Eine hohe *Betweenness Centrality* kann ausdrücken, dass ein Akteur einen großen Einfluss und Autorität über einen Cluster im Netzwerk verfügt, es kann jedoch auch sein, dass der Akteur nur als Vermittler beider Enden dient. [7, Vgl.]

Dieser Wert lässt sich dadurch berechnen, indem jedes Ecken-Paar des Netzwerkes genommen wird und die Anzahl der zwischen ihnen liegenden Ecken auf dem kürzesten Weg gezählt werden (*geodesic distance*). Man zählt dann, wie oft ein Knoten als "Brücke" fungiert.

Beim Transparenzregister Projekt sind es die Unternehmen und Personen, die als Vermittler fungieren und zentrale Rollen in Bereichen einnehmen können. Auf diese wird ein besonderes Augenmerk gelegt, da hier die meisten Auffälligkeiten vermutet werden.

**Closeness Centrality** hilft dabei, Cluster von Knoten zu finden, die sehr nahe aneinander sind. Dies geschieht über einen Algorithmus, der den kürzesten Weg zwischen den Knoten sucht und die Knoten mit einer Punktzahl aus der Summe aller Pfade versieht. Knoten mit einer hohen *Closeness Centrality* haben einen kurzen Weg zu allen anderen Knoten. Diese sind sehr effizient bei der Informationsverteilung - somit sind Akteure mit einer hohen *Closeness Centrality* in der Lage, schnell das gesamte Netzwerk zu beeinflussen.

Bei der Interpretation dieser Kennzahl können Informationsverteiler bestimmt werden, jedoch haben in einem sehr verbundenen Netzwerk die Ecken meist einen sehr ähnlichen *Closeness Centrality* Wert. Daher ist es bei diesen Netzwerken sinnvoll, eher Informationsverteiler in den einzelnen Clustern auszumachen. [7, Vgl.]

Dieser Wert wird berechnet, indem man die Gesamtanzahl an Schritte zu einer Ecke zählt und diesen Wert invertiert.

Neben der *Betweenness Centrality* ist dies eine der besonderen Kennzahlen, da hiermit Akteure gefunden werden, die aufgrund ihrer möglichst geringen direkten Verbindungen gar nicht auffallen würden, aber durch die kurzen Wege eventuell doch Beziehungen zu vielen Akteuren besitzen.

**Eigenvector Centrality** gewichtet, anders als bei der *Degree Centrality*, die Nachbarn unterschiedlich. Dafür werden die Kanten des Ausgangsknoten gemessen, aber auch die Kanten der Folgeknoten und so weiter bis der gesamte Graph durchlaufen ist. Nicht jeder Nachbar hat nach dieser Metrik den gleichen Wert. Dadurch werden Ecken erkannt, die einen Einfluss durchs gesamte oder einen Großteil des Netzwerkes haben. Die *Eigenvector Centrality* kann sowohl für gerichtete als auch ungerichtete Graphen verwendet werden - in der Praxis zeigt sich allerdings, dass die ungerichteten Graphen deutlich besser funktionieren. Die Problematiken der *Eigenvector Centrality* bei gerichteten Graphen kann mittels der *Katz Centrality* behoben werden, welche aber in dieser Arbeit nicht weiter behandelt wird. [6, Vgl. S. 169-171]

Berechnet wird die Kennzahl je Knoten durch das Bilden eines Eigenvektors und Iterieren über jede der Kanten. Wenn die Kennzahl durch  $x$  repräsentiert wird und die Kanten durch  $i$ , können die Mengen der Kanten der Nachbarn durch  $x = \sum A_{ii} X_i$ , bestimmt

werden, wobei  $A_{ii}$  ein Element der Adjacency Matrix ist. Dieser Prozess muss iterativ durchgeführt werden, womit man  $x(t) = A^t x(0)$  erhält.

**PageRank Centrality** ist eine Variante der *Eigenvector Centrality*. Bei diesem Wert wird jeder Knoten mit einer Punktzahl abhängig der eingehenden Verbindungen ausgestattet. Die Verbindungen werden dann abhängig vom ausgehenden Knoten gewichtet. Diese Kennzahl wird genutzt, um bei *directed Graphs* einflussreiche Akteure auszumachen. Es war einer der ersten Rangfolgen Algorithmen hinter der Google Search Engine und wurde nach dem Entwickler und Gründer Larry Page benannt. [7, vgl.]

Akteure mit einem hohen *PageRank Centrality* Wert können als besonders einflussreich über ihre direkten Verbindungen hinaus interpretiert werden.

### 2.3.2 Group-Level Metriken

**K-Cores** ist eine Drill-Down Möglichkeit im Netz. Jeder Knoten erhält ein k-Wert abhängig von seinem degree. Die Knoten werden dann gruppiert und gefiltert. Werte mit einem niedrigen k-Wert werden raus genommen. Somit bleiben nur Werte mit einem hohen k-Wert übrig und es bilden sich semi-autonome Gruppierungen innerhalb des Netzwerks. [8, vgl.]

Der k-Wert bietet eine Möglichkeit für das Transparenzregister verschiedene Zoom Stufen einzubauen, damit gerade bei hohen Mengen an Daten man noch einen Überblick gewinnt.

**Distance/ shortest path** gibt an, wie viele “hops” benötigt werden, um von einer Ecke zur anderen zu kommen. Der kürzeste Weg gibt die Route an, mit der man mit so wenigen “hops” wie möglich durchs Netz kommt. Die “hops” können auch gewichtet werden, um Distanzen berechnen zu können oder die Menge an “hops”. [8, Vgl.]

Dieser Wert sagt etwas zur Weite des Netzwerks aus. Im Zusammenhang mit dem Projekt liefert diese Metrik eher unwichtigere Erkenntnisse. Bahnbrechende Besonderheiten lassen sich im Umfeld von Unternehmen und Personen Verflechtungen mit dem kürzesten Weg nicht herausfinden.

**Network Diameter** ist die kürzeste Verbindung der beiden am weitesten entfernten Ecken. Es zeigt Einblicke über den Weg, der genommen werden muss, um alle Ecken des Netzes zu erreichen.

Die Aussagekraft dieser Metrik ist vergleichbar zu der des kürzesten Weges und ist für das Transparenzregister vernachlässigbar. Diese Metriken können bei zeitlichem Puffer als zusätzliches Feature berechnet werden.

**Graph density** ist das Verhältnis der Anzahl vorhandener Ecken zu möglichen Ecken. Die Dichte des Gesamtgraphen ist 1. Bei isolierten Knoten wäre es 0.



Mit dieser Metriken erhält man einen Einblick über die Dichte und somit die Stärke und Menge an Kanten im Graph - man kann so erkennen, welche Gruppe an Unternehmen besser vernetzt sind als andere. Damit sollte diese Kennzahl, zwar nicht als Top Priorität, aber im späteren Verlauf des Projektes, mit eingebaut werden.

### 2.3.3 Network-Level Metriken

**Modularity** ist die Aggregation des Netzwerks in Untergruppen abhängig der Stärke der Verbindungen. Die Untergruppen werden Module oder *Communities* genannt. Die Modularität gibt die Stärke der Verbindungen an. Eine hohe Modularität sagt aus, dass eine enge Verbindung innerhalb der *Community* besteht.

Die meisten reinen Netzwerk-Level Metriken sind generell einfacher zu berechnen, da sie oft einen konkreten Wert darstellen und nicht pro Ecke kalkuliert sind. Somit können sie schnell eingebaut werden. Die Aussagekraft der Modularität steigt mit der Anzahl an Netzwerken, die man vergleichen möchte. Für das Projekt liegt hier Potential im Vergleich von Ländern oder Regionen, oder der Vergleich einzelner Unternehmens-Verflechtungen. Dieses Level ist aber zum aktuellen Zeitpunkt noch nicht im Betrachtungsraum des Projektes.

**Connected Components** sind Untergruppen von Ecken-Paaren, welche jeweils über Wege verbunden sind. Bei Graphen mit mehreren Connected components gibt die Vereinigung dieser die Summe der Kanten des Graphes wieder.

Mit diesem Wert können zwei Akteure oder Unternehmen gefunden werden, die mit einander verbunden sind. Auch diese Metrik ist im Vergleich zu den Element-Level Metriken erst mal der Priorität nach nachgestellt.

**Average Clustering Coefficient** gibt die Wahrscheinlichkeit an, dass zwei verbundene Ecken einen Cluster bilden.

**Average Path length** ist die durchschnittliche Anzahl an Verbindungen, um zwei Ecken zu verbinden.

Sowohl *Average Clustering Coefficient* als auch *Average Path length* können dem Transparenzregister zusätzliche Analyseinformationen liefern, sind aber wie die anderen Netzwerk-Level Kennzahlen eher in späteren Iterationen und Funktionen einzubauen.

## 3 Ein Social Graph für das Transparenzregister

In diesem Abschnitt wird der Einstieg in die Verflechtungsanalyse für das Transparenzregister gegeben. Betrachtet wird, anhand der Ergebnisse eines Jupyter Notebooks, wie die Metriken auf die ersten Daten des Projektes angewendet werden können, welche Resultate sich vermuten lassen und wie das weitere Vorgehen im Projekt sein wird.

### 3.1 Aufbau des Social Graph/ SNA und Prämissen

Für die Umsetzung des Graphen wird die Programmiersprache Python in der Version 3.11 mit der freien Bibliothek NetworkX genutzt. NetworkX ist ein Framework zur Erstellung von Graphen und Netzwerken mit Python Datenstrukturen. Es können Graphen, Digraphen und Multigraphen damit erstellt werden. Des Weiteren kann eine NetworkX Struktur auch in Matplotlib oder PyVis visualisiert werden. Zu den Feinheiten der Visualisierung wird mehr in der Hausarbeit "Datenvisualisierung" berichtet.

Da zum Entstehungszeitpunkt dieser Arbeit das Projekt noch nicht vollständig mit Daten versorgt ist, wird an dieser Stelle mit Mockup-Daten ein Graph erzeugt und gegen Ende dieser Arbeit eine Vorschau mit den ersten Daten in einem Netzwerk gezeigt.

Für die Erstellung der Daten wird mit der Python Bibliothek Pandas aus einer Excel Datei Mockup-Daten zu verschiedenen Automobilherstellern geladen.

```
# import pandas
import pandas as pd

# create dataframe based on the sample data
df_nodes = pd.read_csv('companies.csv', sep = ';')

# define shape based on the type
node_shape = {'Company': 'dot', 'Person': 'triangle'}
df_nodes['shape'] = df_nodes['type'].map(node_shape)

# define color based on branche
node_color = {'Automobilhersteller': '#729b79ff',
              'Automobilzulieferer': '#475b63ff',
              'Branche_3': '#f3e8eeff',
              'Branche_4': '#bacdb0ff', 'Branche_5': '#2
              e2c2fff'}
df_nodes['color'] = df_nodes['branche'].map(node_color)

# add information column that can be used for
the mouse over in the graph
df_nodes = df_nodes.fillna('')
df_nodes['title'] = df_nodes['label'] + '\n' +
df_nodes['branche']

# show first five entries of the dataframe
print(df_nodes.head())
```

Als Ergebnis erhält man ein Dataframe mit den verschiedenen Automobilherstellern.

Tabelle 1: Tabelle der Automobilhersteller.

ID	Name	Typ
1	Porsche Automobil Holding	Company
2	Volkswagen AG	Company
3	Volkswagen	Company

Neben den Daten zu den Firmen wird noch eine zweite Tabelle "relations" eingelesen, welche die Beziehungen zwischen den Akteuren beinhaltet. Aus den beiden Tabellen wird ein harmonisiertes Dataframe erstellt, aus welchem mit der Bibliothek NetworkX ein Graph erstellt wird. Dafür wird die Methode `from_pandas_edgelist` genutzt - diese erstellt aus einem Dataframe einen Graphen.

```
# import networkx
import networkx as nx

# create edges from dataframe
graph = nx.from_pandas_edgelist(df_edges, source="from",
                                target="to", edge_attr="label")
```

Anschließend wird der erzeugte Graph mit PyVis visualisiert.

```
# visualize using pyvis
from pyvis.network import Network

net = Network(
    directed=False, neighborhood_highlight=True,
    bgcolor="white", font_color="black")

# pass networkx graph to pyvis
net.from_nx(graph)

# set edge options
net.inherit_edge_colors(False)
net.set_edge_smooth("dynamic")

adj_list = net.get_adj_list()

# calculate and update size of the nodes
depending on their number of edges
for node_id, neighbors in adj_list.items():
    # df["edges"] = measure_vector.values()

    size = 10 # len(neighbors)*5
```

```

next(
    (node.update({"size": size}) for node in net.nodes
    if node["id"] == node_id),
    None,
)

# set the node distance and spring length using
  repulsion
net.repulsion(node_distance=150, spring_length=50)

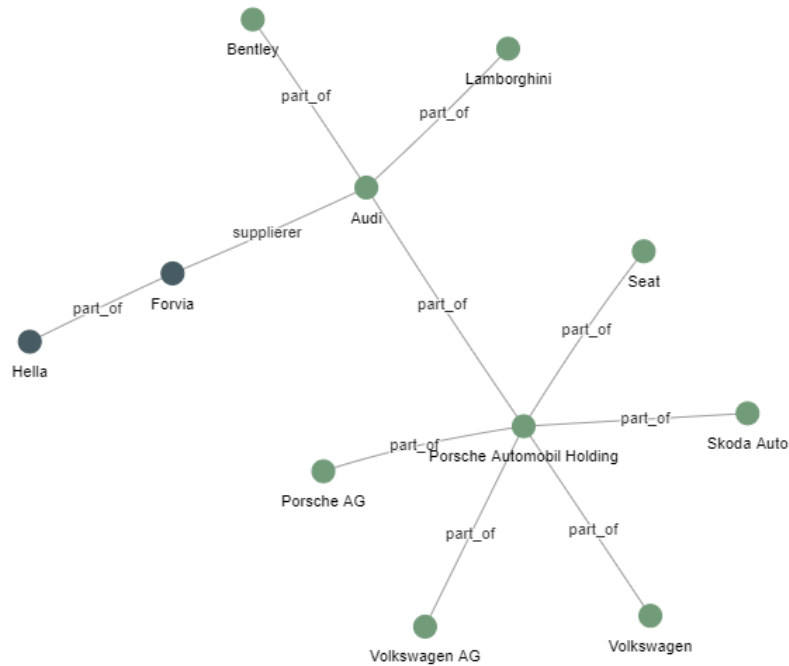
# activate physics buttons to further explore the
  available solvers:
# barnesHut, forceAtlas2Based, repulsion,
  hierarchicalRepulsion
net.show_buttons(filter_=["physics"])

# save graph as HTML
net.save_graph("./metrics/test.html")

```

Das Resultat ist ein vollständiger Graph, welcher als HTML gespeichert ist. Öffnet man den Graphen in einem Browser, kann man die einzelnen Knoten auswählen und bekommt ein Highlighting der verknüpften Knoten. Des Weiteren können Einstellungen an der Physik vorgenommen werden, um die Ansicht des Graphen zu verändern, beispielsweise die Knoten auseinander zu ziehen.

Abbildung 1: Abbildung eines Graphens mit Mockdaten



## 3.2 Anwendung der Social Network Analysis (SNA)

Der Graph kann nun mit den verschiedenen Metriken bestückt werden, sodass die SNA vollzogen werden kann.

### 3.2.1 Erstellen eines Graphen mit der Element-Level Metriken

Über das Framework NetworkX besteht die Möglichkeit, die Metriken direkt berechnen zu lassen. Dazu wird die Methode *eigenvector\_centrality* auf den Graphen angewandt. Als Rückgabewert gibt es ein Dictionary mit den Eigenvector Werten. Das Dictionary muss mit einem Faktor multipliziert werden, um einen sichtlichen Unterschied bei den Größen der Ecken zu erhalten. Über die adjacency list des Netzwerks kann auf die Ecken des Netzwerks zugegriffen werden. Diese wird zu Nutze gemacht, um in einer for-Schleife die Größe der Ecken neu zu setzen. Der Quellcodes sieht wie folgt aus:

```
adj_list = net.get_adj_list()

measure_vector = {}

if measure_type == "eigenvector":
    measure_vector = nx.eigenvector_centrality(graph)
```

```

df["eigenvector"] = measure_vector.values()
if measure_type == "degree":
measure_vector = nx.degree_centrality(graph)
df["degree"] = measure_vector.values()
if measure_type == "betweenness":
measure_vector = nx.betweenness_centrality(graph)
df["betweenness"] = measure_vector.values()
if measure_type == "closeness":
measure_vector = nx.closeness_centrality(graph)
df["closeness"] = measure_vector.values()

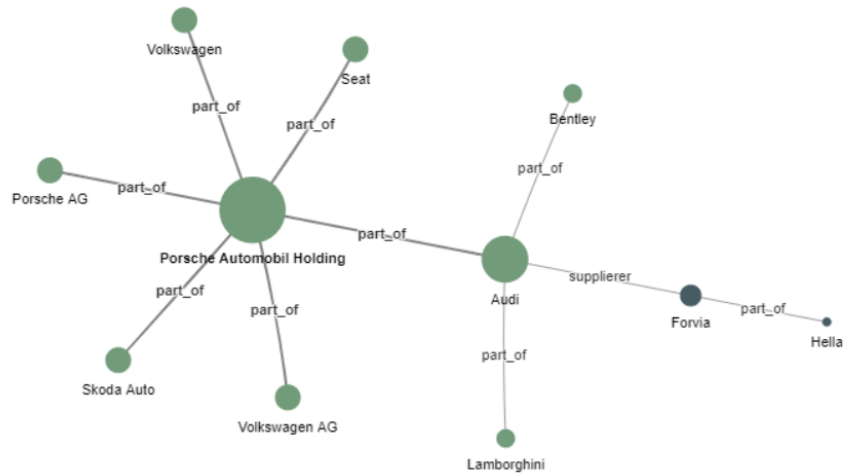
# calculate and update size of the nodes depending on
# their number of edges
for node_id, neighbors in adj_list.items():
# df["edges"] = measure_vector.values()

if measure_type == "edges":
size = 10 # len(neighbors)*5
else:
size = measure_vector[node_id] * 50
next(
(
node.update({"size": size})
for node in net.nodes
if node["id"] == node_id
),
None,
)

```

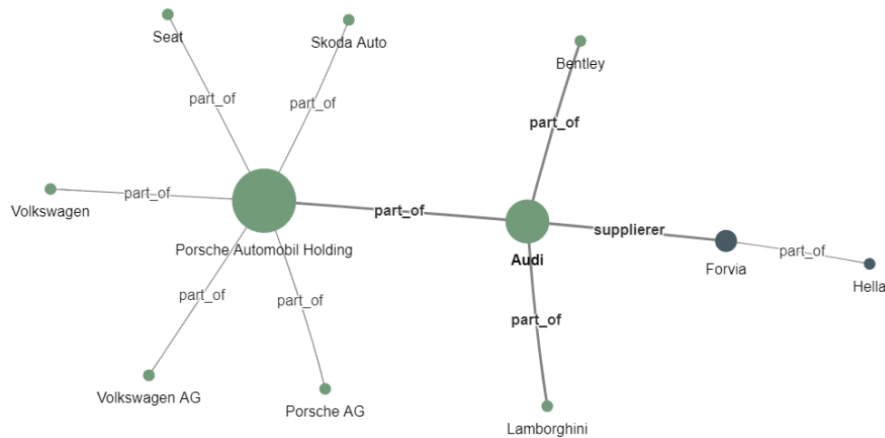
Selbiges wird mit den Kennzahlen *degree\_centrality*, *betweenness\_centrality* und *closeness\_centrality* durchgeführt. Über die *save\_graph* Methode kann das Netzwerk dank des Pyvis Frameworks als HTML gespeichert und das fertige Netz im Browser betrachtet werden.

Abbildung 2: Netzwerk mit der Metrik eigenvector centrality.



Anhand der Veränderung des Netzwerks kann man sehen, wie die Auswirkungen der Kennzahlen sind. Der Eigenvector misst, wie viele Verbindungen von einem Knoten ausgehen und wie viele vom nächsten Nachbarn aus weitergehen, bis das Netz durchdrungen ist. Daher sticht vor allem die Porsche AG in diesem Beispiel deutlich hervor, da diese viele direkt Verbindungen hat und mit dem Audi Knoten verbunden ist, der wiederum die zweit meisten Verknüpfungen besitzt.

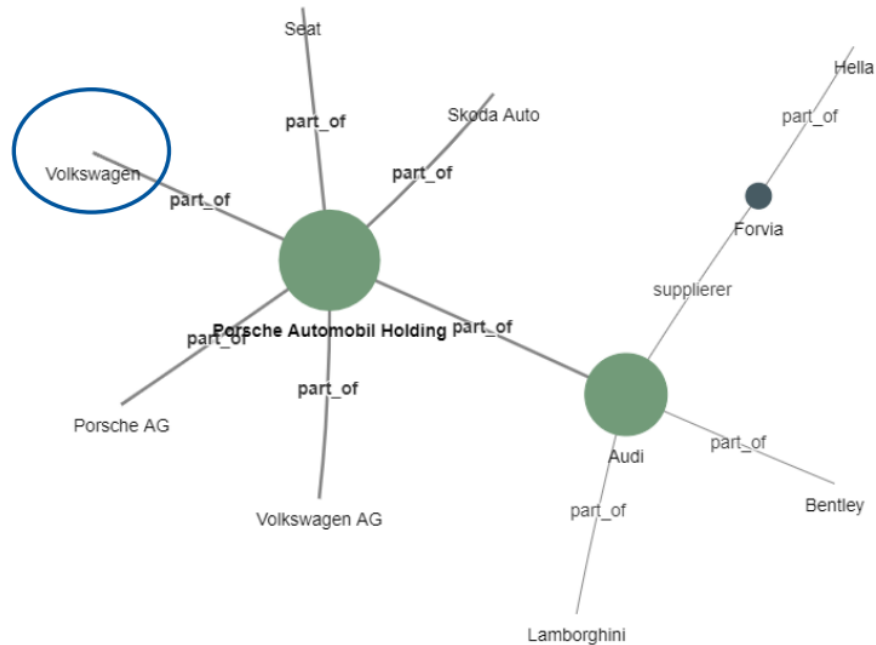
Abbildung 3: Netzwerk mit der Metrik degree centrality.



Die *Degree Centrality* zeigt hingegen ein etwas anderes Bild. Hier sind die Hauptakteure noch einmal deutlich größer im Verhältnis zu den Blättern des Netzes. Es wurden hierfür nur die direkten Verbindungen der Knoten gezählt, deswegen ist "Hella" auch genau gleich

groß wie “Seat” und “Skoda Auto”, da alle nur eine direkte Verbindung besitzen. Beim vorherigen Graphen war “Seat” größer, da es mit einem einflussreichen Knoten verbunden war und Hella nicht.

Abbildung 4: Netzwerk mit der Metrik *betweenness centrality*.



Im dritten Graphen mit der *betweenness centrality* sieht man, dass die Blätter keinen Knoten mehr haben, da dieser nicht als "Brücke" fungiert. In einem sehr großen Netzwerk könnte man solche Knoten wegfällen lassen, um ein genaueren Überblick der wichtigen Akteure zu erhalten.



Abbildung 5: Netzwerk mit der Metrik closeness centrality.



Die letzte Metrik der Element-Level Metriken zeigt ein eher homogenes Bild. Die Knoten sind generell größer, was daran liegt, dass es hier keine unterschiedlichen Subnetze gibt.

### 3.2.2 Berechnung, Aufbau und Interpretationsmöglichkeiten der Group-Level Metriken

Für die Group-Level Metriken wurden zwei Metriken an den Testdaten ausprobiert:

- Distance/shortest path
- Network Diameter

Da in den Testdaten keine klare Gruppierung festgestellt wurde, können die Kennzahlen lediglich auf das gesamte Netzwerk angewendet werden. Die ermittelten Ergebnisse zeigen, dass die Distanz einen Wert von 3 und der sogenannte *Network Diameter* einen Wert von 4 aufweisen. Dies bedeutet, dass die kürzeste Verbindung im Netzwerk lediglich 3 Verbindungen erfordert, während die längste Verbindung maximal 4 Verbindungen benötigt. Diese Werte deuten darauf hin, dass das Netzwerk insgesamt recht effizient in Bezug auf die Verbindungen zwischen seinen Knotenpunkten ist, wobei die meisten Verbindungen relativ kurz sind und es nur wenige längere Verbindungen gibt, was bei einem so kleinen Netzwerk nicht verwunderlich ist.

### 3.2.3 Berechnung, Aufbau und Interpretationsmöglichkeiten der Network-Level Metriken

In der Analyse der Netzwerkkennzahlen wurden ausschließlich die Testdaten verwendet, um die durchschnittliche Pfadlänge zu berechnen. Diese beträgt 2,3 Verbindungen. Dies bedeutet, dass es im Schnitt nur etwa 2,3 Schritte braucht, um von einem Punkt zum anderen im Netzwerk zu gelangen. Eine niedrige durchschnittliche Pfadlänge zeigt an, dass das Netzwerk effizient und gut miteinander verbunden ist.

## 3.3 Weitere Analysen

Neben den bereits besprochenen Analysemetriken aus der Graphentheorie werden für das Transparenzregister weitere Elemente benötigt, um mehr Informationen aus den Daten zu gewinnen. Dazu wird ein Ausgangsgraph erstellt, welcher alle Akteure als uneingefärbte Knoten darstellt und diese miteinander nach den Beziehungen aus der Datenbank über Kanten verbindet. Die Beziehung in der Datenbank sind gerichtete Werte zwischen zwei Akteuren. Es können mehrere Beziehungen in dieselbe Richtung gehen. Beispielsweise kann zwischen Unternehmen A und Unternehmen B die Verbindung ist\_Teil\_von, teilen\_x\_Mitarbeiter und beauftragt bestehen. Damit diese in einem *Social Network* lesbar dargestellt werden können, müssen die Kanten ungerichtet und unterschiedlicher Länge sein. Dafür werden die Beziehungen ähnlich wie bei der *Degree Centrality* gemittelt und gewichtet. Die Länge der Kanten ist disproportional abhängig von der Gewichtung und liegt zwischen 1 und 10. Eine hohe Gewichtung resultiert also in einer Kantenlänge nahe 1, eine niedrige nahe 10. Damit wird eine Federkraft generiert, die Knoten mit starken Verbindungen anzieht. Des Weiteren ist eine Mindestdistanz zwischen Knoten festgelegt, damit sich beim Generieren des Netzes keine Knoten überlappen. Somit ergibt sich ein Ausgangsgraph, welcher über die folgenden Anpassungen verändert werden kann, um weitere Erkenntnisse zu gewinnen.

Die erste Option besteht im Verändern der Größe der einzelnen Knoten. Analog zur Größenveränderung bei den Metriken aus der Graphentheorie ist es bei diesem Graphen machbar, eine der folgenden Metriken aus dem Transparenzregister Projekt als Ausgangspunkt für den Radius der Knoten zu wählen und/oder zu filtern:

EBIT, Umsatz, Aktienkurs, Mitarbeiteranzahl und die Wachstumsrate

Personen werden bei dieser Betrachtung weiterhin mit der Einheitsgröße 1 dargestellt oder raus gefiltert. Der Filter kann mit einem Schwellenwert versehen werden, um nur Unternehmen in bestimmter Größenordnung zu vergleichen. Damit wird dem Betrachter ersichtlich, welche Unternehmen beispielsweise einen hohen Umsatz erzielen im Vergleich zum Rest.

Neben dieser Funktionalität können die Knoten nach den folgenden Kategorien eingefärbt oder gefiltert werden:

Branche, Heimatland, Mutterkonzern, Wachstumsrate, und Positiver/Negativer Berichtserstattung

In Kombination mit dem vorherigen Ansatz sind Analysen hinsichtlich der Unternehmen mit den größten Umsätze nach Ländern denkbar.

Abgesehen von diesen eher visuellen Betrachtungen besteht die letzte Konfiguration im Verändern der Kanten. Darüber hinaus kann das Netzwerk nach den einzelnen Beziehungstypen aufgebaut werden.

Weiterhin ist eine Variante auswählbar, bei der die Formel der Federspannung über eine Gewichtung der Akteure erweitert wird. Verbindungen zu Wirtschaftsprüfern werden zum Beispiel geringer bewertet als geteilte Vorstandspersonen. Somit wird der Fokus auf Unternehmensgruppen gelegt, bei denen die priorisierten Merkmale deutlich herausstechen.

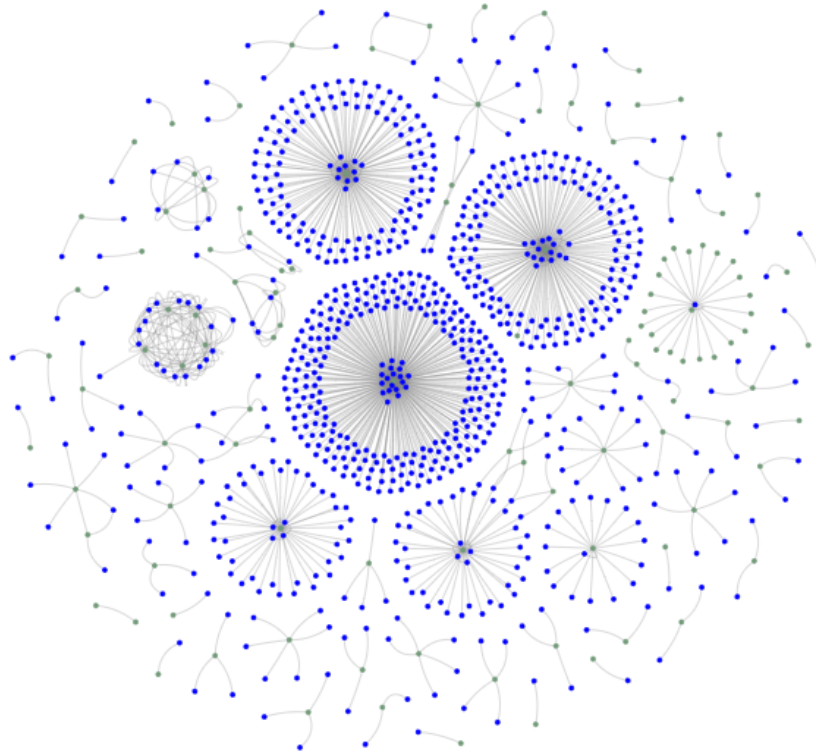
Die letzte Selektion ermöglicht es, Verbindungen zur selben Holding oder Mutterkonzern heraus zu nehmen. Nach diesem Filter sind nur noch externe Verbindungen ersichtlich und der Graph wird deutlich übersichtlicher. Es wird einfacher erkennbar, welche Unternehmen mit welchen Akteuren zusammenarbeiten.

Diese eigenen Analyseoptionen können nach Belieben miteinander kombiniert werden, um verschiedene Sichtweise auf des Netzgefüge zu erhalten.

### **3.4 Perspektive auf einen Graphen mit Unternehmens und Personen Daten**

Gegen ende dieser Arbeit ist hier noch ein Graph mit einem Auszug an realen Daten aus dem Transparenzregister zu sehen. Bereits ohne die Verwendung von Metriken sind Gruppenbildungen deutlich zu erkennen. Insgesamt wurden 1000 Verflechtungen hier dargestellt. Als Knoten dienten Personen und Firmen. Die Unternehmen wurden grün dargestellt, sind aber aufgrund der Größe eher schlecht von den Personen mit blauen Punkten zu unterscheiden.

Abbildung 6: Graph mit Unternehmens- und Personendaten



### 3.5 Handlungsempfehlung

Für das Projekt Transparenzregister leitet sich aus dieser Arbeit folgende Empfehlung für die SNA ab:

Für die Umsetzung der SNA eignet sich die Bibliothek NetworkX sehr gut, da sie anhand eines Pandas Dataframe bereits ein gutes Netzwerk aufbauen kann. Des Weiteren können eine Vielzahl an Metriken effizient mit dem Netzwerk berechnet werden und benötigen keine extra Implementierung von Algorithmen. Zusätzlich wird empfohlen, das Framework PyVis zur Visualisierung zu nutzen, da es das Netzwerk gut aufbereitet und zur Laufzeit Anpassungen an der Grafik zulässt.

Bezogen auf die Auswahl der Metriken wird auf den Nutzen vor allem von Element-Level Metriken verwiesen. Spezifisch sollten folgende Metriken betrachtet werden:

- Degree Centrality
- Betweenes Centrality
- Closeness Centrality

- Eigenvector Centrality

Für die Group-Level Metriken werden folgende Metriken angeraten:

- Distance/shortest path
- Network Diameter

Zuletzt werden für die Netzwerk-Level Metriken die nachstehenden Metriken befürwortet:

- Average Path length
- Modularity

Natürlich können auch die anderen Metriken genutzt werden. Die hier vorgeschlagene Auswahl ergeben nach der Einschätzung und Erläuterungen dieser Arbeit den größtmöglichen Nutzen für das Projekt und sollten aus den genannten Gründen priorisiert werden.

## 4 Zusammenfassung

In diesem Abschnitt wird die Arbeit rückwirkend kritisch betrachtet, zusammengefasst und ein Fazit gezogen. Zum Schluss wird ein kleiner Ausblick auf die nächsten Fragestellungen gegeben, die es im genannten Projekt zu beantworten gilt.

### 4.1 Kritische Reflexion

Zielsetzung war es, im Rahmen dieser Arbeit sich mit der Verflechtungsanalyse auseinanderzusetzen. Die grundlegenden Metriken und Vorgehensweisen wurden aufgezeigt, eine intensivere Betrachtung der Metriken und Anwendung auf größere Daten sollte in einer erneuten Betrachtung als Ziel gesetzt werden. Mit einem Ausschnitt an realen Daten sind bessere und deutlichere Einschätzung der Metriken und Treffen von Aussagen denkbar. Leider bestand der Zugriff auf echte Daten erst gegen Ende der Bearbeitungszeit und konnte deshalb nicht für das Testen der Analyse genutzt werden.

## 4.2 Fazit

In der vorliegenden Untersuchung wird die Bedeutung und Anwendung einer Verflechtungsanalyse verdeutlicht. Die Verflechtungsanalyse findet in Rahmen eines Transparenzregister Projektes für Unternehmensverflechtungen statt und soll im späteren Verlauf des Projektes angewendet werden. Diese Arbeit dient als Grundlage für das Projekt. Deshalb wurde aufgezeigt, auf welcher Grundlage ein Netzwerk an Verflechtungen basiert und wie es mithilfe von NetworkX und PyVis aufgebaut werden kann.

Zusätzliche wurde ein Einblick in die Social Network Analyse gegeben und eine größere Anzahl an Metriken vorgestellt. Die Metriken wurden anhand des Projektes bewertet und eingeordnet, sodass mittels dieser Einschätzung eine Auswahl an Metriken stattfinden kann.

Im Hauptteil des Werks wurde dann auf die konkrete Umsetzung mit Python eingegangen. Es wurde gezeigt, wie ein Netzwerk mit Beispieldaten aufgebaut wird und wie dazu die benötigten Metriken angewandt werden. Ein weiterer Aspekt bestand in der Veranschaulichung, wie die Metriken in das Netz eingebaut werden können, um eine visuelle Analyse durchzuführen.

Neben den Metriken wurde auch auf weitere Analysemöglichkeiten durch Farbgestaltung oder Größenveränderung der Knoten eingegangen. Zu guter Letzt gab es einen Ausblick auf einen Prototypen Graph mit realen Unternehmensdaten, aber ohne weitere Analysen.

## 4.3 Ausblick

Der nächste logische Schritt besteht in der konkreten Ausarbeitung der Verflechtungsanalyse mit realen Daten. Dort gilt es zu überprüfen, welche Erkenntnisse mithilfe der SNA auf den Daten gewonnen werden können. Liefern die vorgeschlagenen Metriken die zu erwartenden Resultate? Sind alle Metriken überhaupt mit den Daten anwendbar? Diese Fragestellungen werden im Rahmen des Projektes angegangen.

## Eigenständigkeitserklärung

Ich versichere, dass ich die schriftliche Ausarbeitung selbstständig angefertigt und keine anderen als die von mir angegebenen und bei Zitaten kenntlich gemachten Quellen und Hilfsmittel benutzt und die vorliegende Arbeit an keiner anderen Stelle zur Erlangung eines Abschlusses vorgelegt habe.

T. Ronneburg

Issum, 06.10.2023

---

Tim Ronneburg

## 5 Literatur

- [1] S. Iwanowski und R. Lang, „Graphentheorie,“ ger, in *Diskrete Mathematik mit Grundlagen*, Wiesbaden: Springer Fachmedien Wiesbaden, 2020, S. 257–320, ISBN: 3658327596.
- [2] P. Hartmann, „Graphentheorie,“ ger, in *Mathematik für Informatiker*, Wiesbaden: Springer Fachmedien Wiesbaden, 2020, S. 269–300, ISBN: 9783658265236.
- [3] I. Pitas, *Graph-Based Social Media Analysis*. CRC Press, 2016, ISBN: 9780429162602.
- [4] X. Fu, *Social Network Analysis*. CRC Press, 2017, ISBN: 9781315369594.
- [5] E. Yüksel, <https://medium.com/@emreeyukseel/a-brief-introduction-to-social-network-analysis-2d13427f5189>.
- [6] M. Newmans, *Networks An introduction*. Oxford University Press Inc., 2010, ISBN: 9780199206650.
- [7] A. Disney, <https://cambridge-intelligence.com/keylines-faqs-social-network-analysis/>.
- [8] C. Intelligence, <https://cambridge-intelligence.com/social-network-analysis/>.