University of Applied Sciences and Arts Northwestern Switzerland
School of Architecture, Civil Engineering and Geomatics

MASTER OF SCIENCE IN ENGINEERING

Matrix-basierte Generierung von Graphenlayouts

Durch Graphen-Layouts können vorhandene Zusammenhänge zwischen unterschiedlichen Entitäten visualisiert werden. Die bestehenden, in einem Textfile beschriebenen Verbindungen, werden in Form von Knoten und Kanten in einen Graph importiert und anschliessend in eine gruppenunterteilte Adjazenzmatrix konvertiert. Die Knotenreihenfolge der generierten Adjazenzmatrix wird durch den Cuthill-McKee Algorithmus innerhalb der Gruppen so angeordnet, dass es möglich ist, anhand dieser Matrix ein Layout automatisiert zu erstellen.

Import und Struktur der bestehenden Layouts

Die bestehenden Layouts liegen im JSON-Format vor und werden mithilfe vom Python-Modul Pandas als Dataframe importiert. Das JSON-File beinhaltet die Verbindungen der einzelnen Entitäten (from-to). Anhand dieser Information wird mit dem Python Modul networkx ein Graph erstellt. Der bestehende Graph wird in eine Adjazenzmatrix konvertiert und die einzelnen Gruppen, welche das Layout aufweist, werden unterschiedlich eingefärbt (vgl. Abb. 4). Die Darstellung der Adjazenzmatrix ist abhängig von der Reihenfolge der einzelnen Knoten, was in der Abbildung 2 sehr gut erkennbar ist.

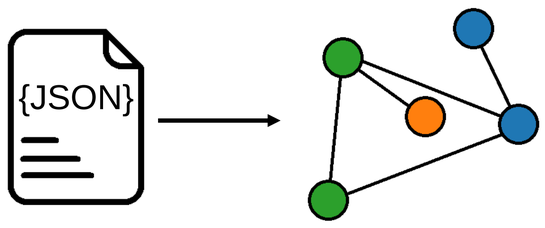


Abb. 1: einfacher Graph, generiert aus einer JSON-Datei

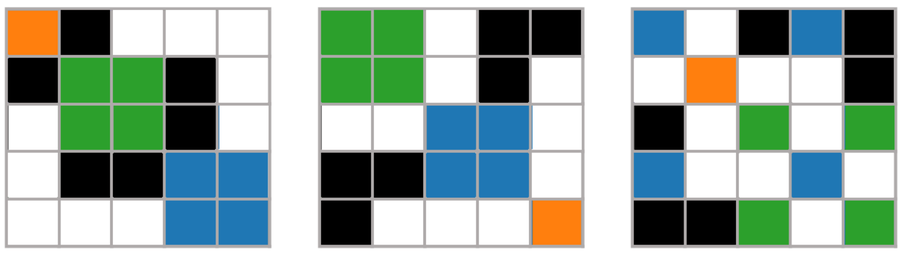


Abb. 2: Adjazenzmatrizen, welche den Graphen der Abbildung 1 beschreiben

Die Adjazenzmatrix links (Abb. 2) ist sortiert nach dem CM-Algorithmus, während die anderen zwei Matrizen zufällig sortiert sind.

Cuthill-McKee-Algorithmus

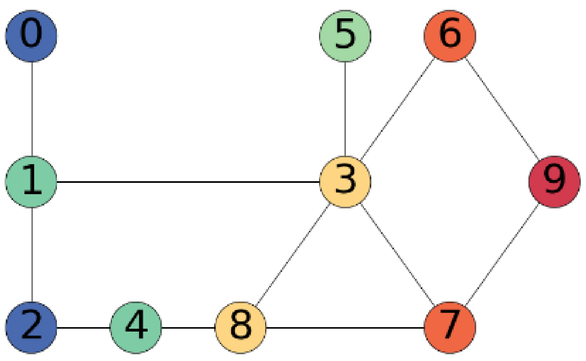
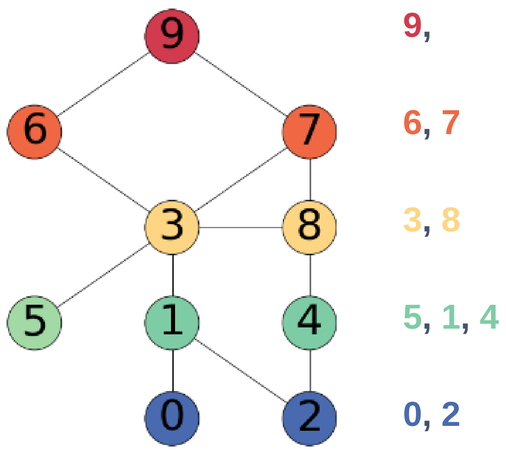


Abb. 3: Graph, eingefärbt nach Ebenen des CM- Algorithmus



9,
6, 7
3, 8
5, 1, 4
0, 2

Abb. 5: Graph, rekonstruiert anhand CM-Algorithmus

Es gibt diverse Algorithmen, um eine Adjazenzmatrix bzw. deren Knoten anzuordnen. Ein sehr starker Algorithmus ist hierbei der Cuthill-McKee-Algorithmus (CM-Algorithmus). Dieser Algorithmus generiert eine vorerst leere Liste und fügt dort einen (pseudo-) peripheren Knoten ein (Abb. 3: [9]). Als nächstes werden jeweils die benachbarten Knoten in dieser Liste ergänzt [6 und 7, als nächste Ebene]. Dann werden nochmals die benachbarten Knoten der eingefügten Knoten abgearbeitet [3 und 8, dritte Ebene]. Dieser Prozess wird so lange wiederholt, bis alle Knoten abgearbeitet wurden und die Liste alle Knoten enthält.

CM-Reihenfolge, gem. Abb. 3:
[9, 6, 7, 3, 8, 5, 1, 4, 0, 2]

Vom CM-Algorithmus zu einem Layout

Um ein Layout zu erstellen, wurde die resultierende Reihenfolge des CM-Algorithmus verwendet und die einzelnen Knoten aufgrund der so generierten Ebenen positioniert. Dieses hierarchische Layout wird für jede Gruppe (Abb. 4) separat erstellt und in einem Gesamtlayout kombiniert. Dadurch werden die Kantenkreuzungen verringert und das Layout ist besser lesbar. Die einzelnen Knoten können noch verschoben und individuell positioniert werden.

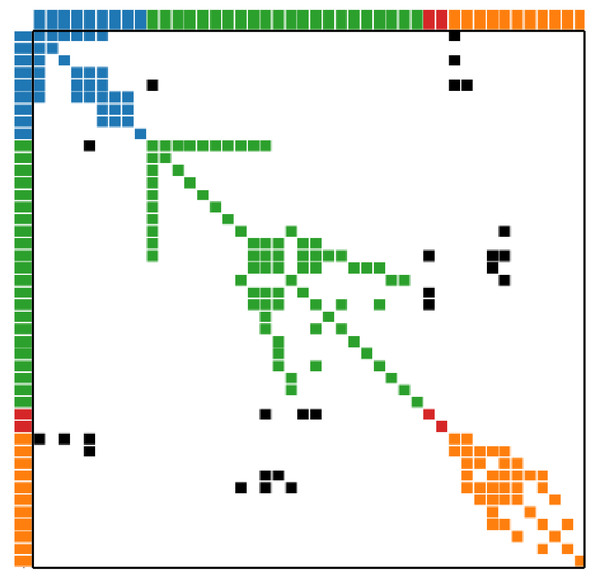
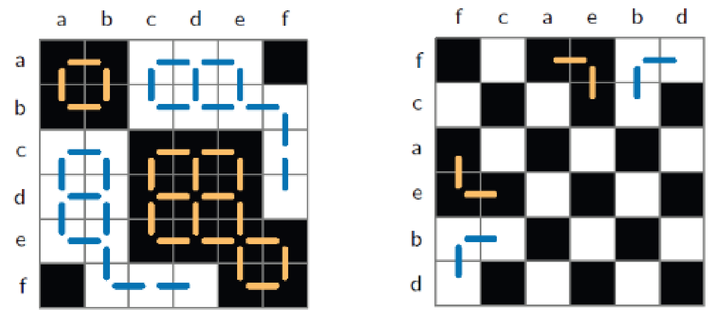


Abb. 4: Adjazenzmatrix des Kundendaten-satzes, einzelne Gruppenunterteilungen sortiert nach CM-Algorithmus

Qualitätsindex zur Beurteilung der Adjazenzmatrizen

Um die Qualität der unterschiedlichen Adjazenzmatrizen zu beurteilen, steht die lokale Statistik von Moran (Morans I) zur Verfügung, welche normiert ist und Werte von -1 (perfekte Verteilung - Schachbrettmuster), über 0 (zufällige Verteilung) bis und mit +1 (perfekte Häufung - alle Felder identisch) annehmen kann. Der CM-Algorithmus liefert von allen untersuchten Algorithmen die höchsten Morans-I Werte



Morans I: +0.33 Morans I -0.73

Abb. 5: Adjazenzmatrizen des selben Graphen, mit unterschiedlichen Morans I-Werten (van Beusekom et al. 2022)

Fazit

Es ist möglich ein bestehendes Modell automatisiert in eine Matrix zu konvertieren und daraus ein Layout zu generieren. Die einzelnen Gruppen liefern hierbei gute Informationen, um die Positionen der Knoten festzusetzen. Somit erhält man ein gutes, jedoch nicht perfektes Layout, welches mit ein paar Korrekturen von "Hand" zu einem ansehbaren Layout wird. Die grösste Schwierigkeit liegt darin, die erstellte 1D-Reihenfolge der Adjazenzmatrix in ein 2D-Layout zu transformieren.

Referenzen:
Van Beusekom, N., Meulemans, W., & Speckmann, B. (2022). Simultaneous Matrix Orderings for Graph Collections. IEEE Transactions on Visualization and Computer Graphics, 28(1)
Stiftinger Martin. (1994). Der Cuthill-McKee-Algorithmus. https://www.iue.tuwien.ac.at/phd/bauer/node55.html [Stand 07.04.2023]
Balci, H., & Dogrusoz, U. (2022). fCoSE: A Fast Compound Graph Layout Algorithm with Cons-traint Support. IEEE Transactions on Visualization and Computer Graphics, 28(12),

Sebastian Biderbost
Supervisor: Prof. Dr. Susanne Bleisch, Expert: Stefan Berner