1 Lösungsidee

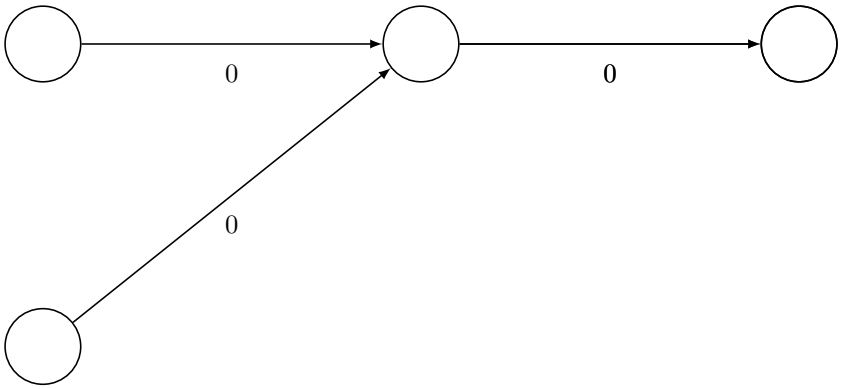
Die Lösungsidee besteht darin, den Dijkstra Algorithmus derart umzugestalten, dass er den Weg mit der geringsten Zahl an Abbiegungen statt der geringsten Länge sucht.

Zuerst wird der kürzeste Weg berechnet, woraufhin dessen Länge und die Anzahl der darinenthaltenen Abbiegungen bestimmt wird. Daraufhin beginnt die Ermittlung des endgültigem Wegs.

Grundlegend verläuft dieser Prozess wie der normale Dijkstra Algorithmus, jedoch mit ein paar kleinen Unterschieden: Der erste Unterschied besteht darin, dass der Algorithmus alle Wege mit einer Länge größer der Länge des kürzesten Wegs plus die erlaubte Extrastrecke (also +15% oder +30%) eliminiert. Der Hauptunterschied ist, dass Dijkstra hierbei nicht nach Länge, sondern nach Anzahl an Abbiegungen optimiert.

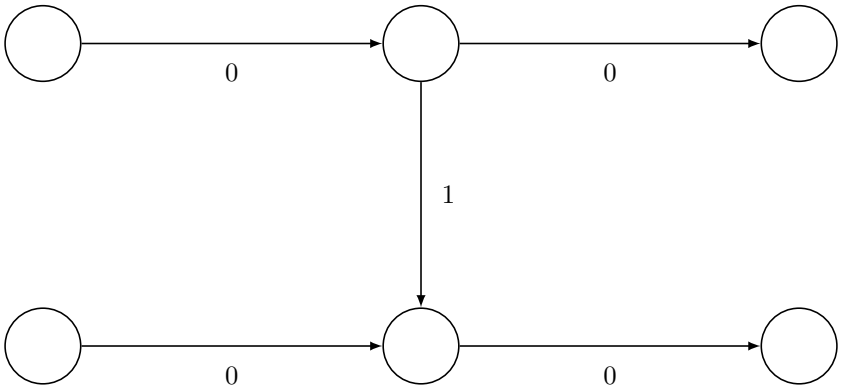
1.1 Modifizierter Dijkstra im Detail

Um Dijkstra im Hinblick auf die minimale Anzahl an Abbiegungen umzudefinieren, könnte die zugrunde liegende Kostenfunktion anstelle der Länge des Graphen die Existenz von Abbiegungen betrachten, entweder mit einer '1' für eine Abbiegung oder einer '0' für eine Gerade. Dabei fällt auf, dass zur Berechnung der Kosten für eine Kante auch die im Weg vorangehende Kante betrachtet werden muss. Dies bedeutet aber, dass Dijkstra nicht in seiner ursprünglichen Form verwendet werden kann. Es könnte ansonsten der Fall auftreten, dass Knoten des Graphen von einem anderen Weg aus erneut besucht würden, was die Kosten der Kanten verändern würde, die mit dem Knoten verbunden sind.



Um dieses Problem zu umgehen, wird der Graph verändert: Zwar werden Kreuzungen innerhalb des Graphen immer noch als Knoten dargestellt, jedoch gibt es für jede Kreuzung mehrere Knoten, nämlich jeweils einen für jede geradlinige Straße, die in der Kreuzung vorhanden ist. Alle Knoten einer Kreuzung sind dabei verbunden durch Kanten mit Kosten von 1.

Das verstehe ich leider auch nicht genau, wo sind denn die mehreren Knoten für eine Kreuzung in dem Bild?



Der