Hochschule Harz	FB Automatisierung und Informatik
Programmierung2	Dipl.-Inf., Dipl.-Ing. (FH) M. Wilhelm
Aufgabe 12:	„Programmierung 2“ für MI / WI Thema: **Dijkstra und A*-Algorithmus**

Versuchsziele

Kenntnisse in der Anwendung vom:

- Berechnung von kürzesten Wegen mit dem Verfahren
  - Dijkstra
  - A*

Aufgabe12:

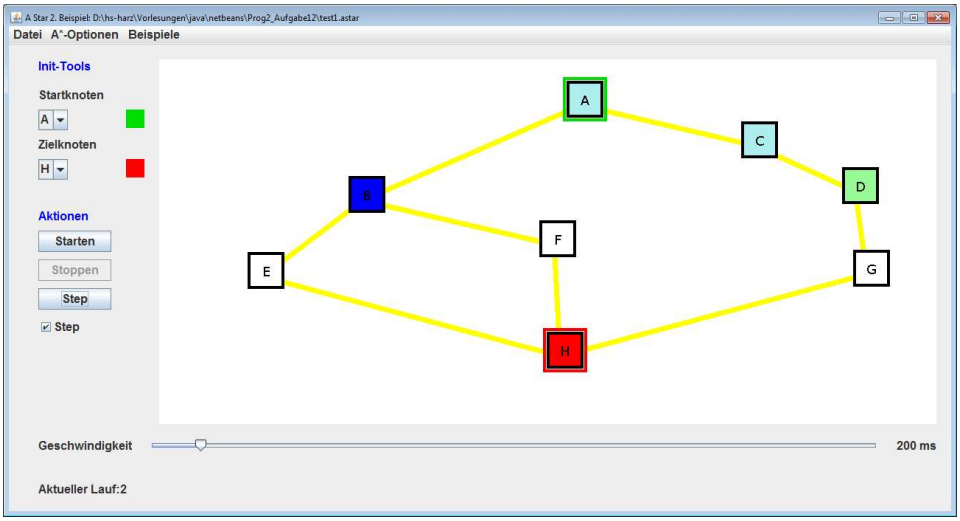

Abbildung 1 Musterlösung

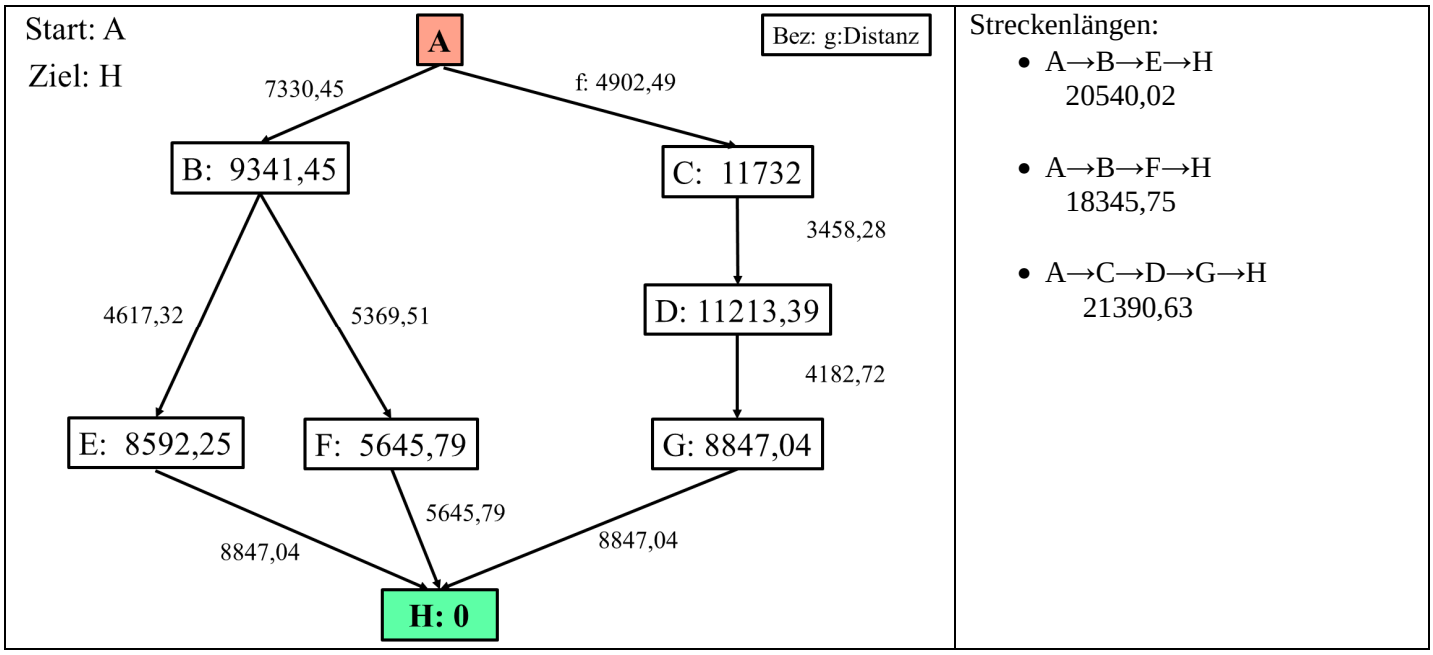

Abbildung 2 Strecken des Systems

In den Knoten steht die Heuristik: hier Euklid (Pythagoras)

Klasseübersicht:

- Aufgabe12 Hauptklasse mit der GUI
- GPanel zeichnet die Knoten und Kanten
- Knoten Bezeichnung, Koordinaten, Status, f,g,Distanz, Liste mit Kanten
- Kante speichert eine Kante, hat den Zielknoten
- DebugFrame Anzeige der laufenden Informationen
- IClose Interface, wird beim Schließen des Fensters benötigt

Aufgaben

1. Teilaufgabe: Projekt erstellen und aufbauen:

- Projektname: Aufgabe12
- Im Projekt soll der vorgegeben Quellcode eingebaut werden (Homepage):
    - Aufgabe12.java hier fehlt Code
    - GPanel.java komplett
    - Knoten.java komplett
    - Kante.java komplett
    - Basis.java komplett
    - IClose komplett
    - DebugFrame.java komplett

2. Teilaufgabe: Implementierung des A*-Algorithmus:

Implementieren Sie den A*-Algorithmus:

Detaillierter Algorithmus:

- Initialisierung:
    - Existiert Startknoten und Zielknoten
    - Im Startknoten wird f,g,distanz auf Null gesetzt
    - In allen anderen wird f,g,distanz auf Double.MAX_VALUE gesetzt
    - Setze den Status aller Knoten auf unknown (ohne Start/Zielknoten)
    - Löschen von openlist
    - Löschen von closedlist
    - Einfügen des Startknotens in die closedList
- While (openList nicht leer)
    - Bestimme alle Nachfolger des aktuellen Knotens
        - Bestimme die Werte f, g und distanz des Nachfolgers
            - f: Abstand vom Start
            - g: geschätzte Distanz zum Ziel (Heuristik)
            - distanz = f+g
        - Ist der Nachfolger in closeList continue
        - Test, ob dieser schon berechnet wurde
            - Falls der Nachfolger i eine größere Entfernung, Variable „distanz“, als die distanz des aktuellen Knotens hat, wird der „distanz“-Wert des Node i durch den niedrigeren Wert ersetzt **und** der aktuellen Knoten als Vorgänger des Knoten i eingetraen.
            - Falls die Distanz zum Ziel größer als die bereits ermittelte ist, wird der aktuelle Nachfolger in die closeList eingetragen.
    - Einfügen des aktuellen Knotens in closeList
    - Bestimme den Knoten aus der openList mit der geringsten Entfernung zum Zielknoten
        - Collection.min **und** compareTo mit dem Attribut „distanz“

Heuristik:

- Manhattan
  - node.x-nodeZiel.x + |node.y-nodeZiel.y|

- Euklid / Pythagoras
  - $\sqrt{(node.x - nodeZiel.x)^2 + (node.y - nodeZiel.y)^2}$

- Chebyshev
  - max(node.x-nodeZiel.x , |node.y-nodeZiel.y|)

Methoden der Klasse „Aufgabe12“:

a) bnStartAction_click:
- Test beim Startknoten
- Test beim Zielknoten
- Startknoten != Zielknoten
- holen nodeStart und nodeZiel
- clear_init_Nodes, wenn man die Datei zweimal startet
- nodeStart und nodeZiel initialisieren
- listen loeschen
- debugfenster starten
- nodeStart: f,g,distanz setzen
- nodeActual setzen
- count auf einen sinnvollen Wert setzen
- Timer

b) action
- Test der Variable count
- Nachfolger suchen und ablaufen
  - closeList -
  - Bestimmung von f,g, distanz
  - check, Knoten schon berechnet?
- NodeActual in closeList
- Minimum aus openList bestimmen

c) drawRoute
- Ausgabe der minimalen Route
- Ausgehend vom Ziel, geht man über das Attribut „prev“ bis zum Startknoten.
- Ausgegeben werden soll aber die richtige Reihenfolge!
- Beachten Sie eine Methode der Klasse Collections

Beachten Sie die Ergebnisse der Testläufe:
- test1_astar.txt
- test2_astar.txt

Testbilder:

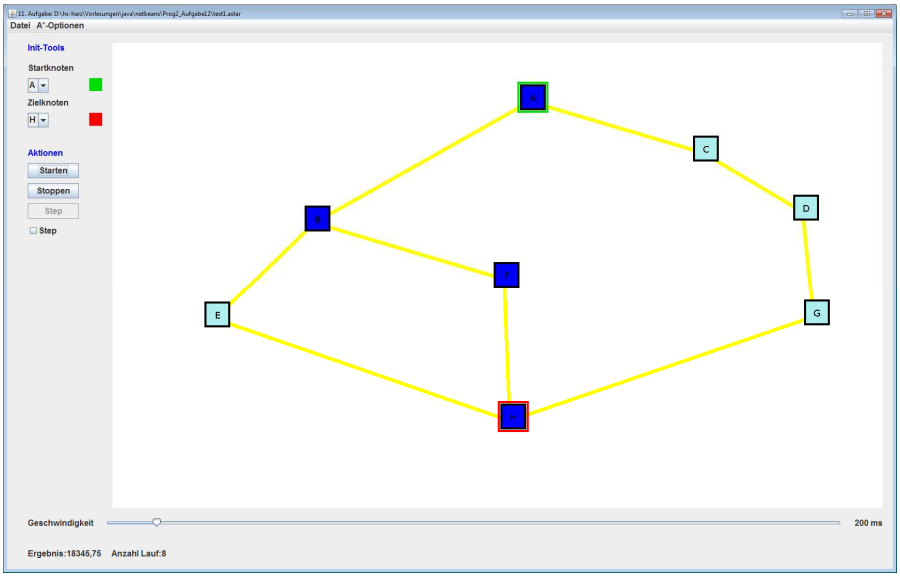

Abbildung 3 test1.astar

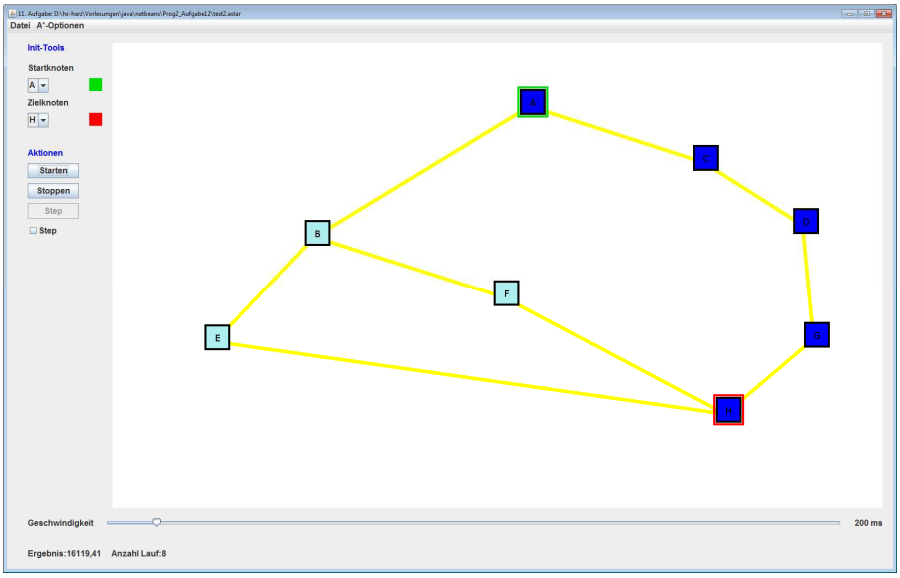

Abbildung 4 test2.astar

Anzeige der Koordinaten in ArcView:

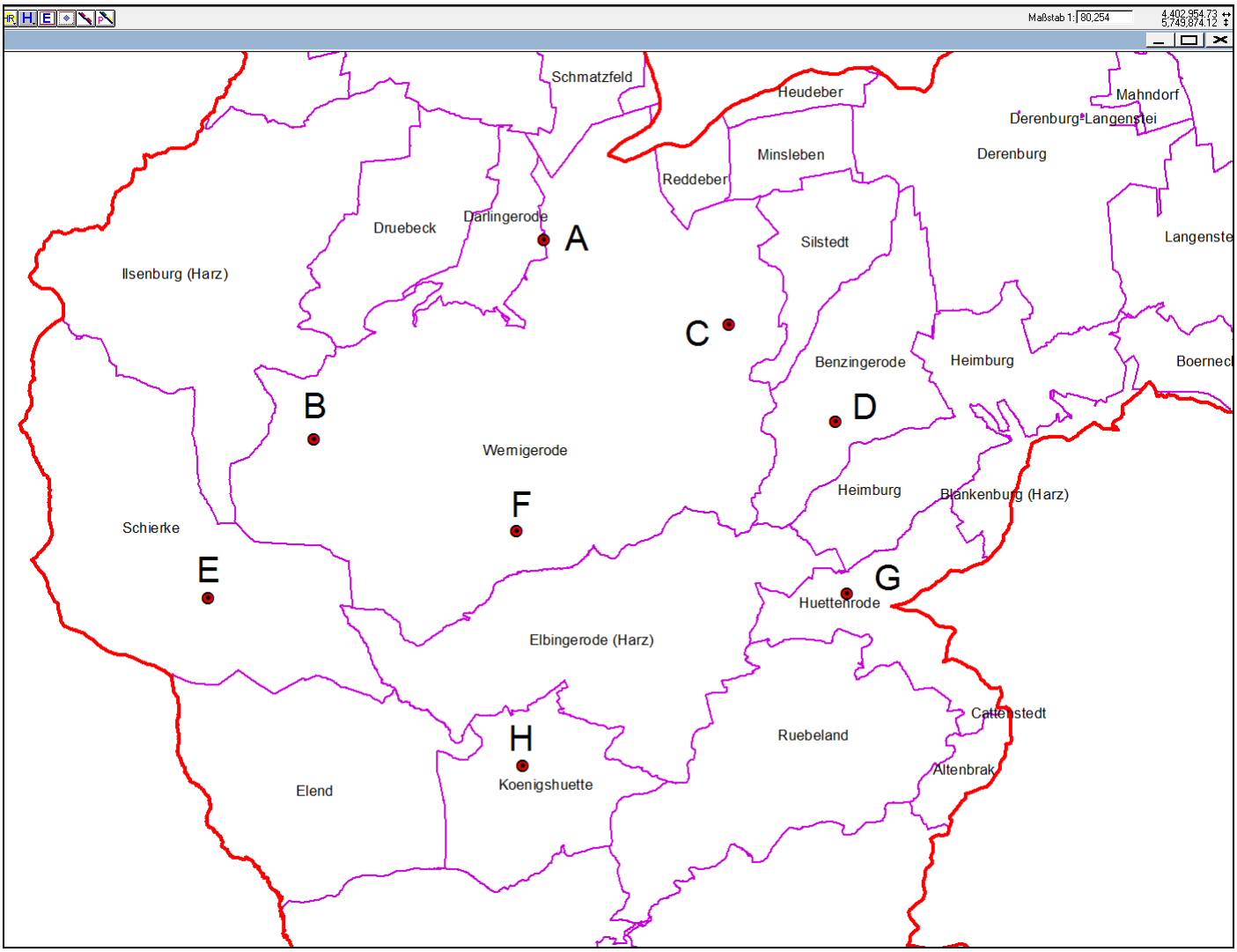

Abbildung 5 Anzeige der Koordinaten in einem Grafischen Informationssystem