Technische Hochschule Rosenheim

Entwicklung von Computerspielen: KI Wegplanung

Fakultät Informatik
FWPM

Technische Hochschule Rosenheim

KI: Wegplanung
Übersicht

- Wegplanung mit A*
- Weltrepräsentation

KI: Wegplanung
Struktur einer KI Engine

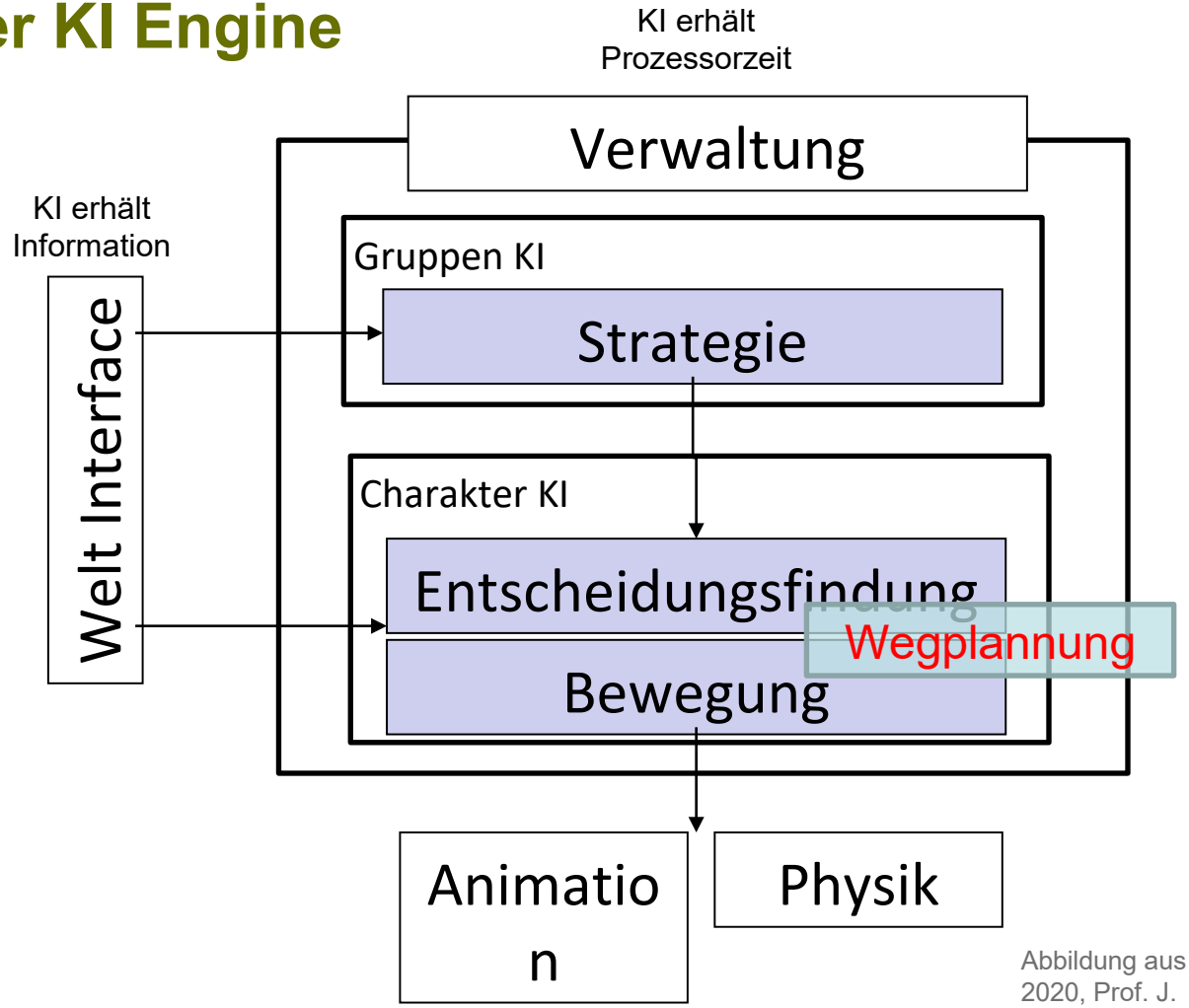


Abbildung aus EVC 2020, Prof. J. Schmidt

Technische Hochschule Rosenheim

KI: Wegplanung
Wegplanung - Anforderungen

- Muss sehr schnell sein
  => Bei vielen NPCs noch wichtiger
  (siehe z.B. Age of Empires 4 vs 4 =Im Endgame=> 200 x 8 = 1600 mögliche KI Agents gleichzeitig)
- Muss glaubwürdige Wege erzeugen

Test: Mehreren Tausend AI Agents in Unity von Brackeys:

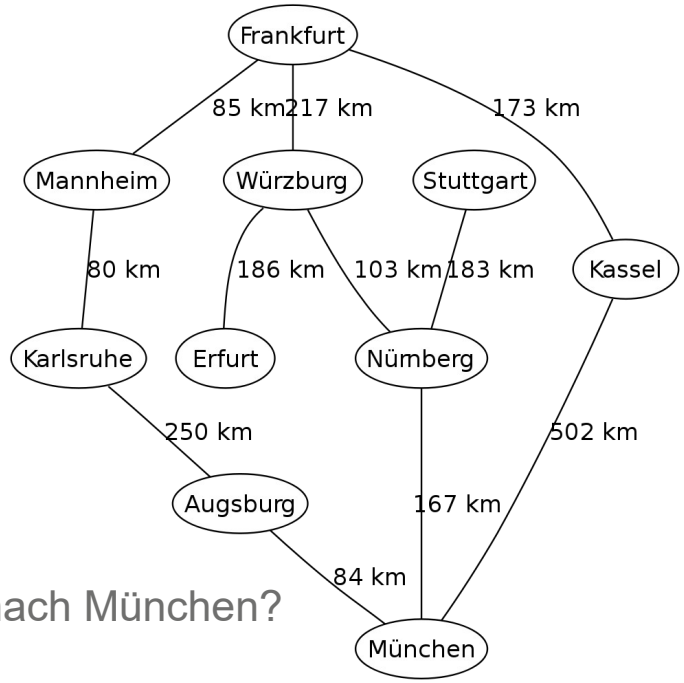
https://www.youtube.com/watch?v=G9Otw12OUvE

KI: Wegplanung
Wegplanung - Anforderungen

- Berechnung vor dem Spiel nicht möglich:
  - Umgebung nicht vollständig bekannt
  - (NPC kennt nur einen Teil der Karte)
  - Umgebung ist dynamisch → erfordert Neuplanung

- Idee:
  - Konstruiere einen Graphen, der das Problem repräsentiert
  - Durchsuche den Graphen nach einem optimalen Pfad

KI: Wegplanung
Wegplanung



- Was ist die kürzeste Verbindung von Frankfurt nach München?
- Benötigt Suchstrategie

Abbildung aus EVC 2020, Prof. J. Schmidt

Technische Hochschule Rosenheim

KI: Wegplanung
Uninformierte (Blinde) Suche

- Verwende nur Information, die in der Problemdefinition verfügbar ist
- Es kann nicht festgestellt werden, ob ein betrachteter Knoten besser ist als ein anderer
- **Algorithmen:**

Tiefensuche
→ findet nicht die kürzeste Route
Breitensuche
→ findet die kürzeste Route nur für *ungewichtete* Graphen
uniforme Kosten Suche (Dijkstra)
→ findet die kürzeste Route für *nicht-negative* Gewichte,expandiert aber unnötig viele Knoten

=> **alle**: exponentielle Zeitkomplexität

Technische Hochschule Rosenheim

KI: Wegplanung
Informierte Suche

- Idee: **Bestensuche** (best-first search)
  zu expandierender Knoten wird auf Basis einer Bewertungsfunktion f(n) ausgewählt
- Bewertungsfunktion misst Entfernung zum Ziel
  wähle den Knoten, der am besten erscheint
- Implementierung:
  Datenstruktur
  - ist eine Prioritätswarteschlange (MIN-Heap)
  - „offen“-Liste: bewertete, aber noch nicht expandierte Knoten (Init = {Startknoten})
  - „geschlossen“-Liste: expandierte Knoten (Init = {})
- Hier betrachtet: A* Suche

Technische Hochschule Rosenheim

KI: Wegplanung
Bestensuche

- Idee:
  vermeide die Expansion von Pfaden, die bereits teuer sind
- Auswahl basierend auf Bewertungsfunktion f(n)
- f(*n*) = g(*n*) + h(*n*)
  g(*n*) **bisherige Kosten** vom Start zum Knoten *n*
  h(*n*) **geschätzte Kosten** von *n* zum **Ziel**
  f(*n*) **geschätzte Gesamtkosten** von Start zum Ziel über *n*
- Wähle den Knoten zur Expansion, der die geringsten Gesamtkosten hat

Technische Hochschule Rosenheim

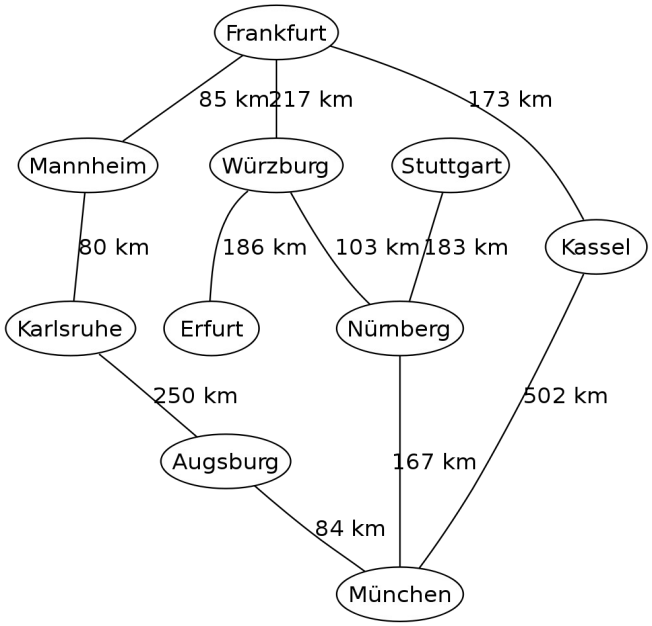
KI: Wegplanung
Heuristische Funktion

- h(*n*) = geschätzte Kosten des kürzesten/billigsten Pfads vom Knoten n zum Zielknoten
- Ist *n* der Zielknoten, dann gilt h(*n*) = 0

- Wichtig:
  h(*n*) muss **zusätzliche Information** bereitstellen,
  d.h. es darf nicht aus dem Graphen selbst berechenbar sein

  => z.B. Luftlinie der Zwischenknoten zum Ziel;
  Luftlinie In Spielen: Distanz zwischen Positionsvektoren
  → In Unity Vector3.Distance(Vektor1, Vektor2);

KI: Wegplanung
Heuristische Funktion - Beispiel



Entfernungen der Städte von München – Luftlinie

Stadt	Entfernung
Frankfurt	304
Mannheim	272
Würzburg	218
Stuttgart	189
Kassel	382
Karlsruhe	253
Erfurt	317
Nürnberg	149
Augsburg	57
München	0

Abbildung aus EVC 2020, Prof. J. Schmidt

Technische Hochschule Rosenheim

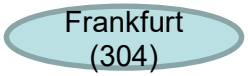
KI: Wegplanung
A* Suche

- A* verwendet eine **optimistische** Heuristik
  - die Kosten zum Ziel werden **niemals** überschätzt
  - **h(*n*) ≤ h*(*n*)**, wobei **h*(*n*)** die tatsächlichen Kosten von ***n*** zum **Ziel** sind
  - wird dieses Kriterium nicht eingehalten, dann findet A* nicht die kürzeste Route

- meist: **monotone** Heuristik
  stärker einschränkend
  zusätzlich muss gelten: **h(*n*) ≤ t(*n*, *n'*) + h(*n'*)**
  - ***n*** und ***n'*** sind Nachbarknoten
  - **t(*n*, *n'*)** sind die tatsächlichen Kosten von ***n*** nach ***n'***
  - die geschätzten Kosten zum Ziel müssen kleiner sein als die Kosten über einen beliebigen Nachbarknoten (Dreiecksungleichung)

  euklidischer Abstand ist monoton (z.B. Luftlinie)

KI: Wegplanung
Beispiel von Frankfurt nach München



f(F) = g(F, F) + h(F) = 0 + 304 = 304

Stadt	Entfernung
Frankfurt (F)	304
Mannheim (MA)	272
Würzburg (WÜ)	218
Stuttgart (S)	189
Kassel (KS)	382
Karlsruhe (KA)	253
Erfurt (EF)	317
Nürnberg (N)	149
Augsburg (A)	57
München (M)	0

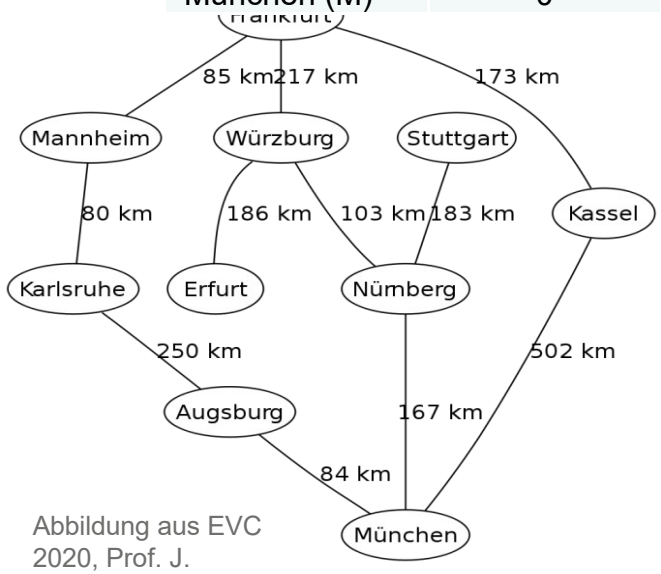
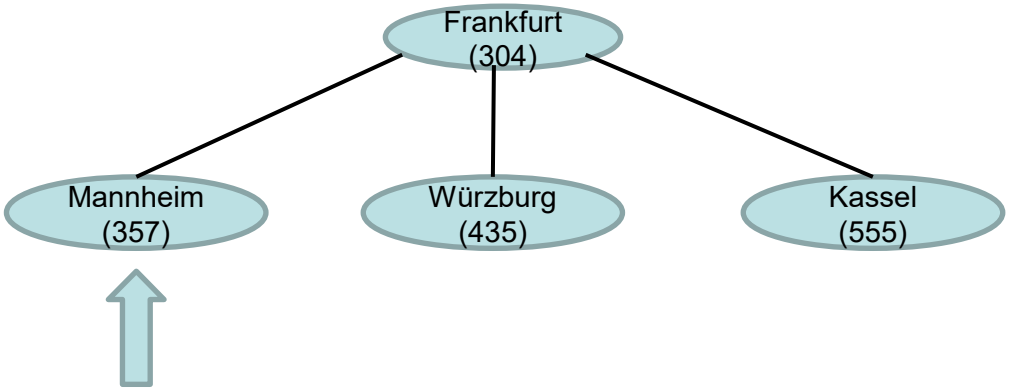


Abbildung aus EVC 2020, Prof. J. Schmidt

offen: F (304)
geschlossen: –

Technische Hochschule **Rosenheim**

KI: Wegplanung
Beispiel von Frankfurt nach München



Stadt	Entfernung
Frankfurt (F)	304
Mannheim (MA)	272
Würzburg (WÜ)	218
Stuttgart (S)	189
Kassel (KS)	382
Karlsruhe (KA)	253
Erfurt (EF)	317
Nürnberg (N)	149
Augsburg (A)	57
München (M)	0

f(MA) = g(F, MA) + h(MA) = 85 + 272 = 357

f(WÜ) = g(F, WÜ) + h(WÜ) = 217 + 218 = 435

f(KS) = g(F, KS) + h(KS) = 173 + 382 = 555

offen: MA (357), WÜ (435), KS (555)
geschlossen: F

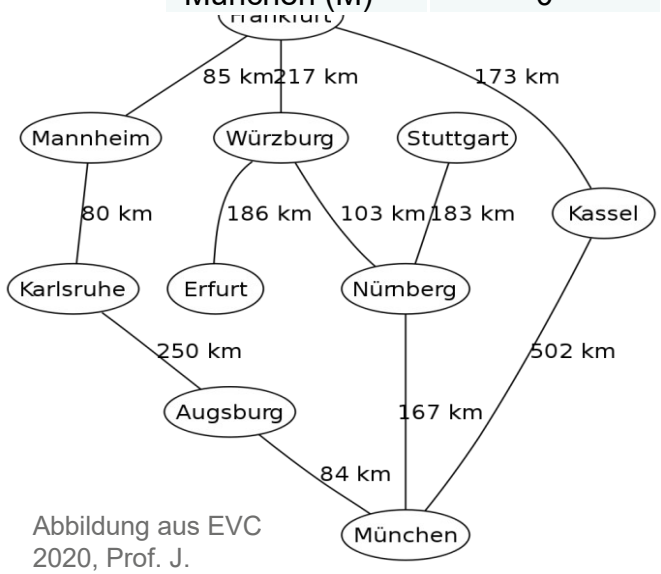
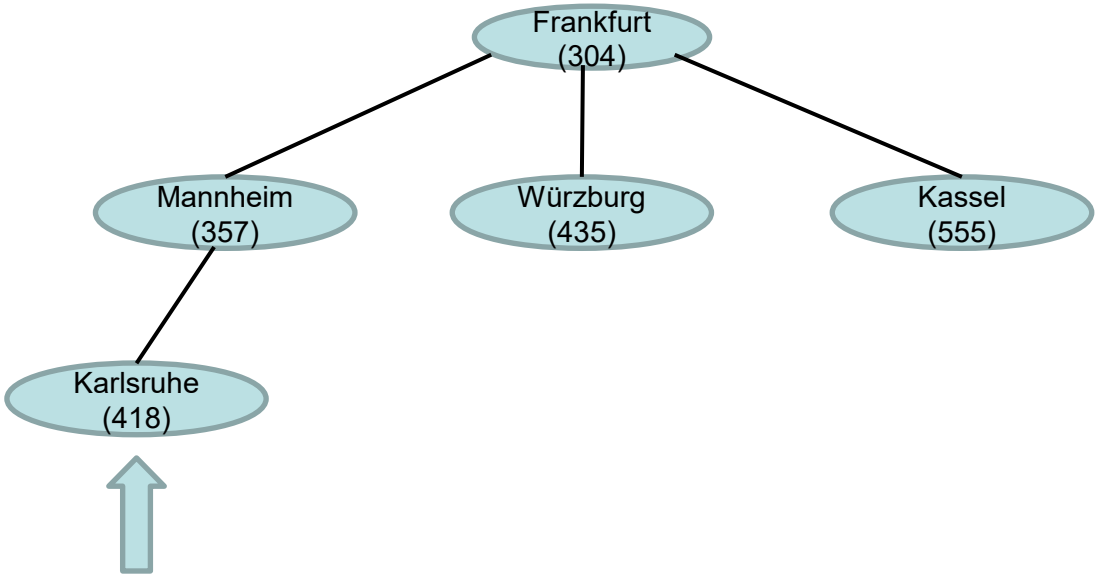


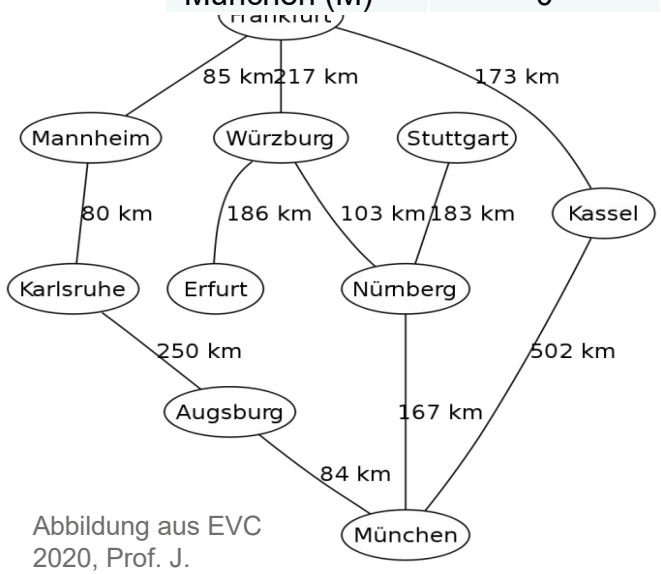
Abbildung aus EVC 2020, Prof. J. Schmidt

Technische Hochschule Rosenheim

KI: Wegplanung
Beispiel von Frankfurt nach München



Stadt	Entfernung
Frankfurt (F)	304
Mannheim (MA)	272
Würzburg (WÜ)	218
Stuttgart (S)	189
Kassel (KS)	382
Karlsruhe (KA)	253
Erfurt (EF)	317
Nürnberg (N)	149
Augsburg (A)	57
München (M)	0



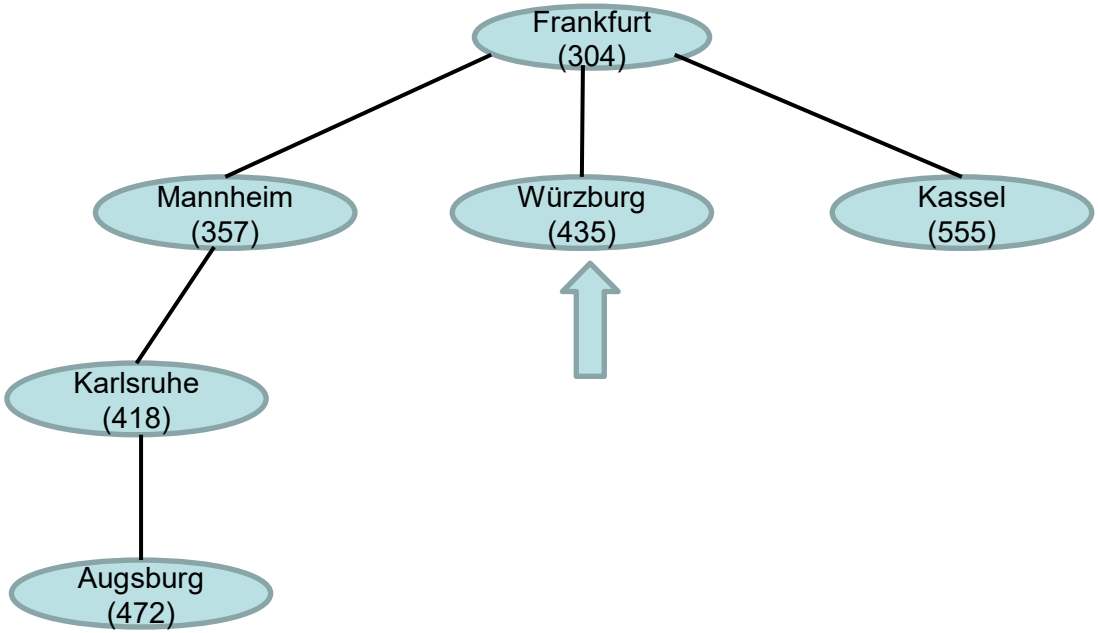
f(KA) = g(F, KA) + h(KA) = 165 + 253 = 418

offen: KA (418), WÜ (435), KS (555)
geschlossen: F, MA

Abbildung aus EVC 2020, Prof. J. Schmidt

Technische Hochschule Rosenheim

KI: Wegplanung
Beispiel von Frankfurt nach München



Stadt	Entfernung
Frankfurt (F)	304
Mannheim (MA)	272
Würzburg (WÜ)	218
Stuttgart (S)	189
Kassel (KS)	382
Karlsruhe (KA)	253
Erfurt (EF)	317
Nürnberg (N)	149
Augsburg (A)	57
München (M)	0

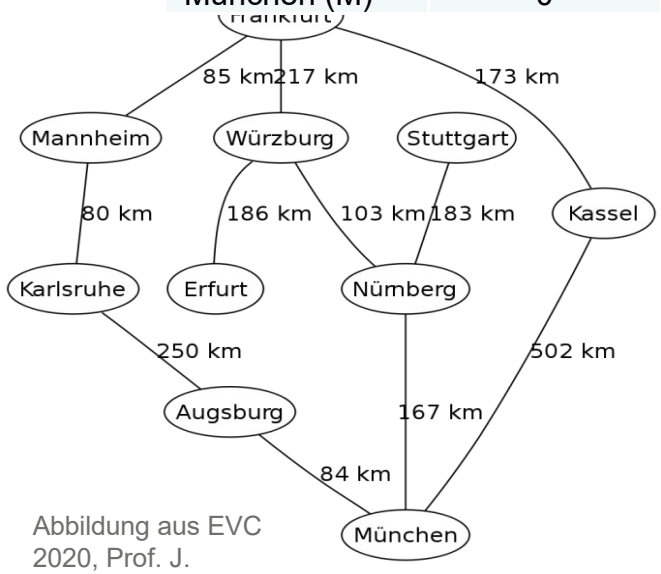


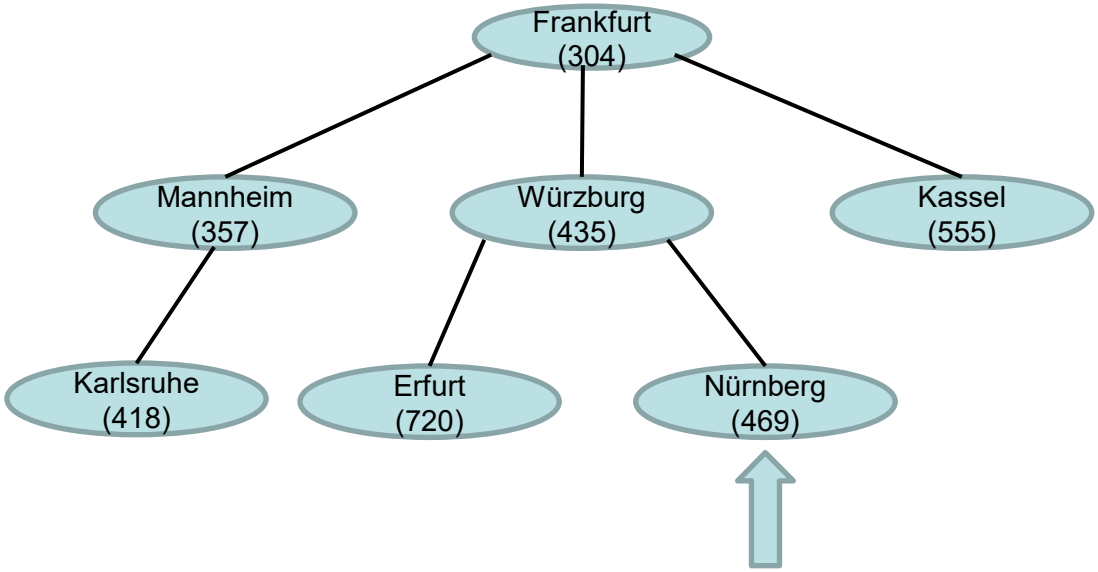
Abbildung aus EVC 2020, Prof. J. Schmidt

f(A) = g(F, A) + h(A) = 415 + 57 = 472

offen: WÜ (435), A (472), KS (555)
geschlossen: F, MA, KA

Technische Hochschule Rosenheim

KI: Wegplanung
Beispiel von Frankfurt nach München



Stadt	Entfernung
Frankfurt (F)	304
Mannheim (MA)	272
Würzburg (WÜ)	218
Stuttgart (S)	189
Kassel (KS)	382
Karlsruhe (KA)	253
Erfurt (EF)	317
Nürnberg (N)	149
Augsburg (A)	57
München (M)	0

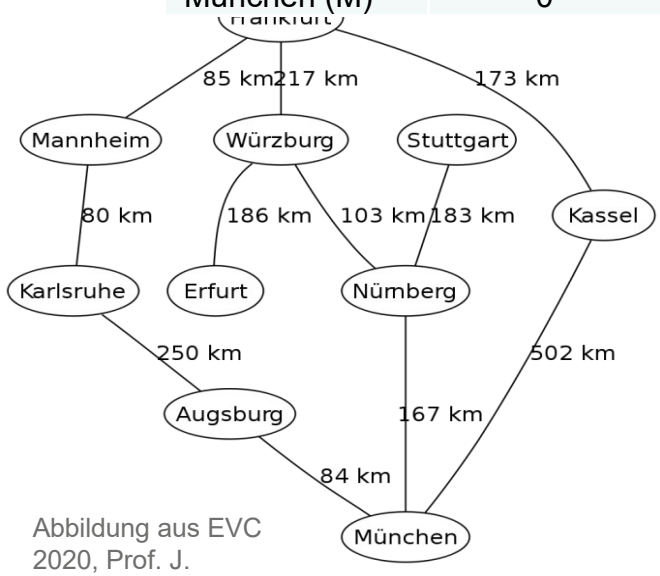


Abbildung aus EVC 2020, Prof. J. Schmidt

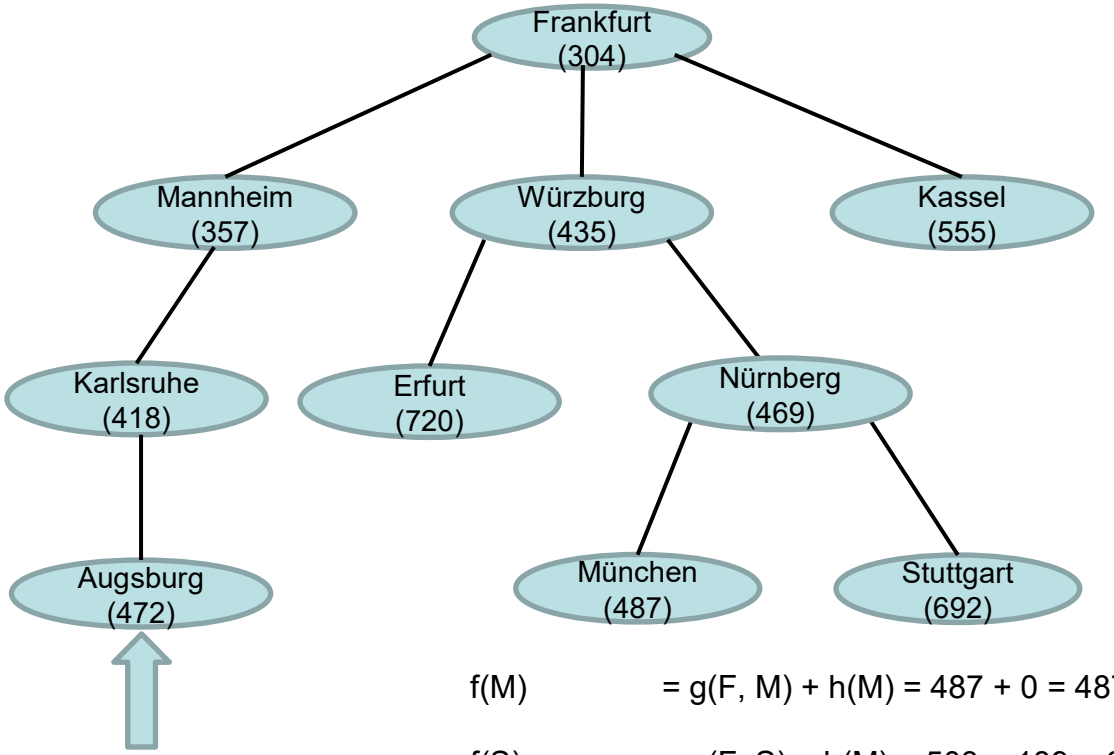
f(EF) = g(F, EF) + h(EF) = 403 + 317 = 720

f(N) = g(F, N) + h(N) = 320 + 149 = 469

offen: N (469), A (472), KS (555), EF (720)
geschlossen: F, MA, KA, WÜ

Technische Hochschule **Rosenheim**

KI: Wegplanung
Beispiel von Frankfurt nach München



Stadt	Entfernung
Frankfurt (F)	304
Mannheim (MA)	272
Würzburg (WÜ)	218
Stuttgart (S)	189
Kassel (KS)	382
Karlsruhe (KA)	253
Erfurt (EF)	317
Nürnberg (N)	149
Augsburg (A)	57
München (M)	0

f(M) = g(F, M) + h(M) = 487 + 0 = 487

f(S) = g(F, S) + h(M) = 503 + 189 = 692

offen: A (472), M (487), KS (555), S (692), EF (720)
geschlossen: F, MA, KA, WÜ, N

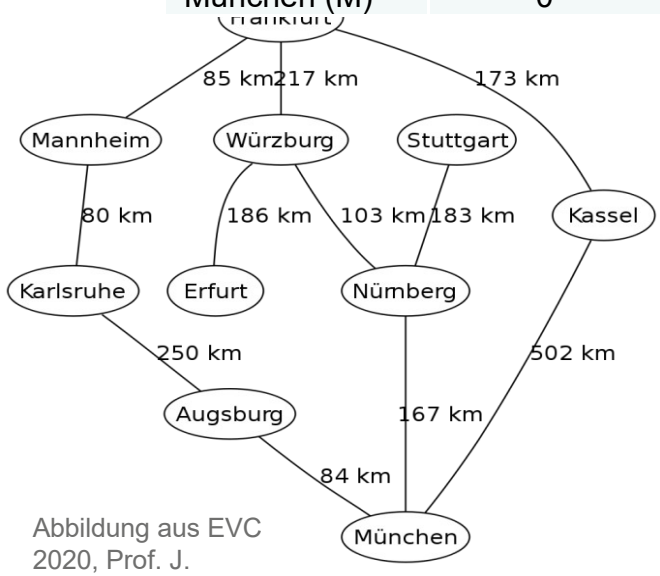
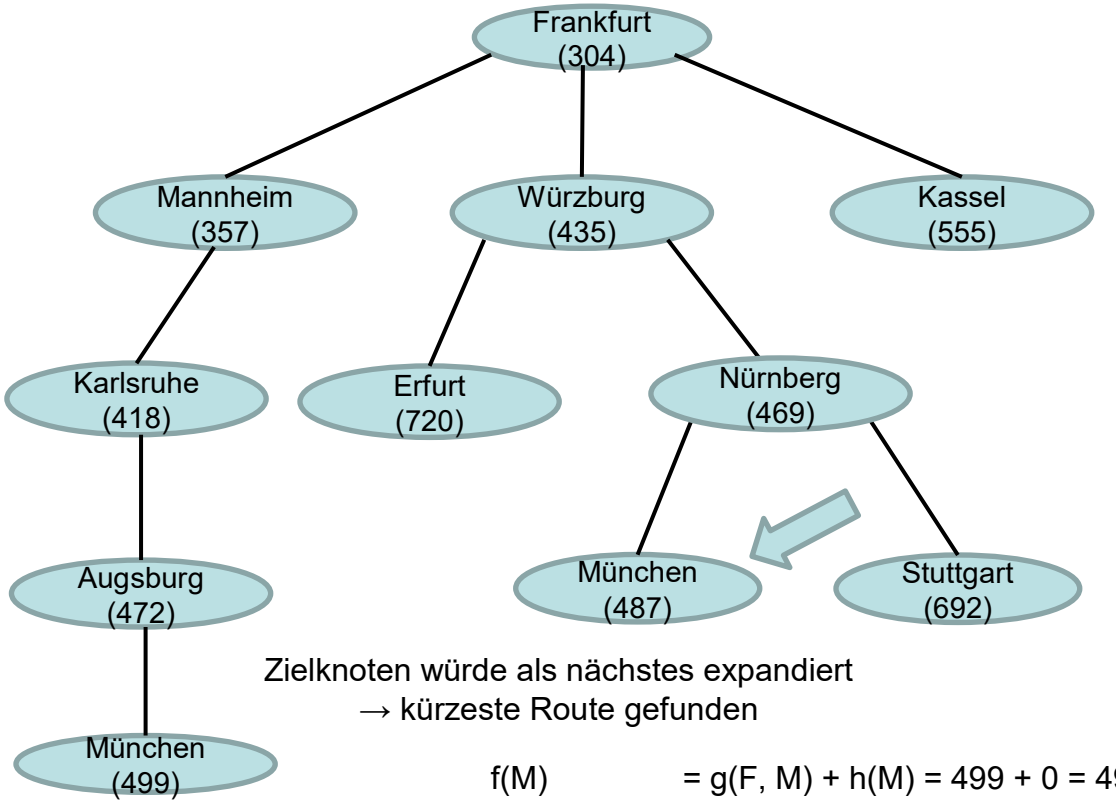


Abbildung aus EVC 2020, Prof. J. Schmidt

Technische Hochschule **Rosenheim**

KI: Wegplanung
Beispiel von Frankfurt nach München



Stadt	Entfernung
Frankfurt (F)	304
Mannheim (MA)	272
Würzburg (WÜ)	218
Stuttgart (S)	189
Kassel (KS)	382
Karlsruhe (KA)	253
Erfurt (EF)	317
Nürnberg (N)	149
Augsburg (A)	57
München (M)	0

Zielknoten würde als nächstes expandiert
→ kürzeste Route gefunden

f(M) = g(F, M) + h(M) = 499 + 0 = 499

offen: M (487), M (499), KS (555), S (692), EF (720)
geschlossen: F, MA, KA, WÜ, N, A

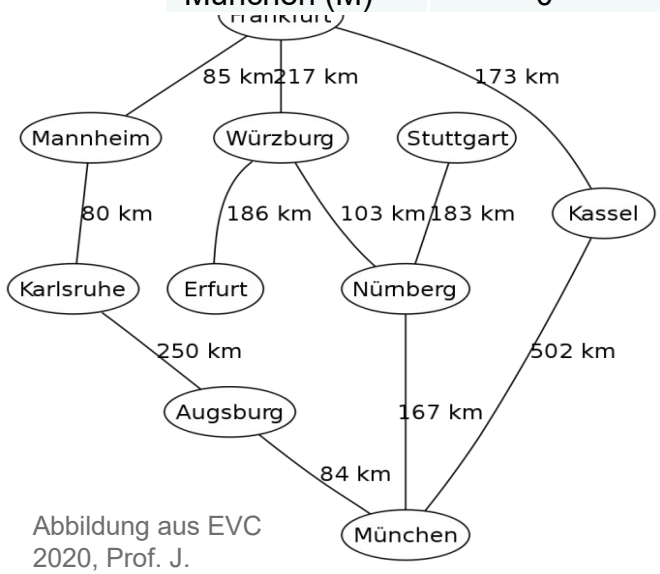


Abbildung aus EVC 2020, Prof. J. Schmidt

Technische Hochschule Rosenheim

KI: Wegplanung
Bewertung A*

- A* findet immer eine Lösung, sofern diese existiert
- A* ist **optimal-effizient**
  kein anderer optimaler Algorithmus expandiert weniger Knoten
  (außer evtl. bei gleicher Bewertung verschiedener Knoten)
  wenn die Bedingungen für **h(*n*)** eingehalten werden
- Zeitkomplexität:
  exponentiell in der Länge des kürzesten Pfads
- Speicherkomplexität:
  hält alle Knoten im Speicher
  daher ist Speicherverbrauch das Hauptproblem, nicht die Rechenzeit

KI: Wegplanung
Heuristiken für Spiele

- Null-Heuristik
  - **h(*n*) = 0**
  - entspricht Dijkstra
  - expandiert sehr viele Knoten
- euklidischer Abstand
  - Entfernung Luftlinie – wie im Beispiel
  - gut für Freiland
  - expandiert in Innenräumen relativ viele Knoten
- Cluster-Heuristik
  - Gruppierung von Knoten
  - nur für Innenräume
  - expandiert weniger Knoten als euklidischer Abstand
  - nicht mehr garantiert optimal

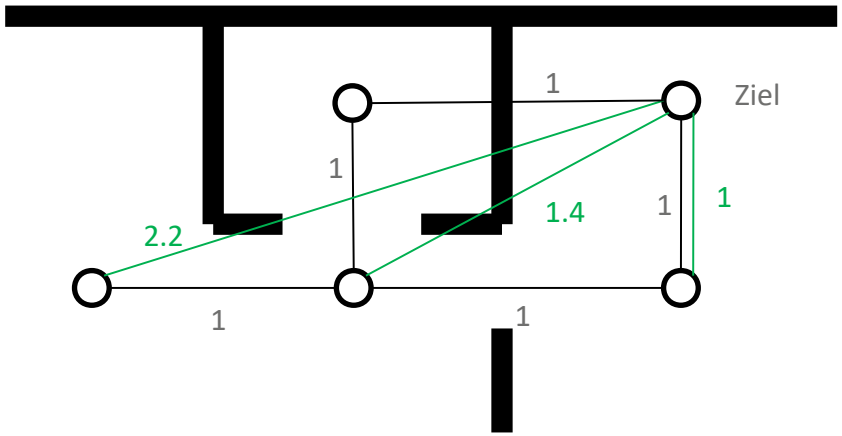


Abbildung aus EVC 2020, Prof. J. Schmidt

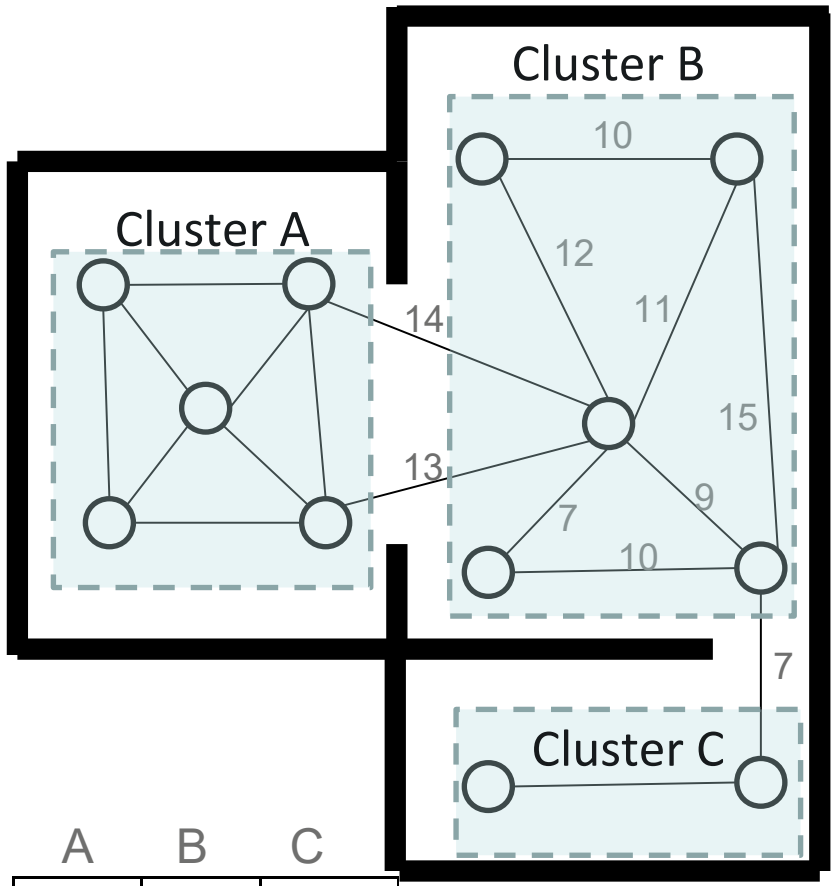
Technische Hochschule Rosenheim

KI: Wegplanung
Cluster Heuristik

- Offline (während der Entwicklung)

  Schritt 1: Gruppiere Knoten
  - entweder manuell (z.B. alle Knoten eines Raums)
  - oder mit Clusteringalgorithmen für Graphen

  Schritt 2: Erstelle Lookup-Tabelle
  - enthält kürzeste Entfernung für jedes Clusterpaar
  - viele A* Suchläufe erforderlich



	A	B	C
A	x	13	29
B	13	x	7
C	29	7	x

Technische Hochschule Rosenheim

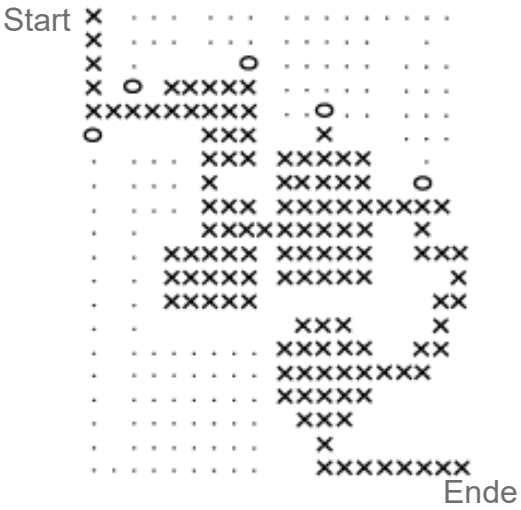
KI: Wegplanung
Cluster Heuristik

- Online (während des Spiels)
  - Start- und Zielknoten in verschiedenen Clustern:
    - verwende Lookup-Tabelle
    - alle Knoten innerhalb eines Clusters erhalten den gleichen Wert für die heuristische Funktion h(*n*)
  - Start- und Zielknoten im gleichen Cluster:
    - verwende euklidischen Abstand
- Eigenschaften
  - berücksichtigt, dass sich zwischen räumlich nahe beieinander liegenden Punkten Wände befinden können
  - besucht innerhalb eines Clusters sehr viele Knoten, da h(n) für alle gleich ist → Clustergröße muss gut gewählt werden

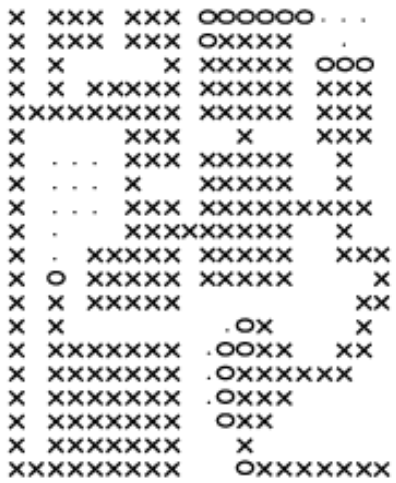
Technische Hochschule **Rosenheim**

KI: Wegplanung
Heuristiken - Innenräume

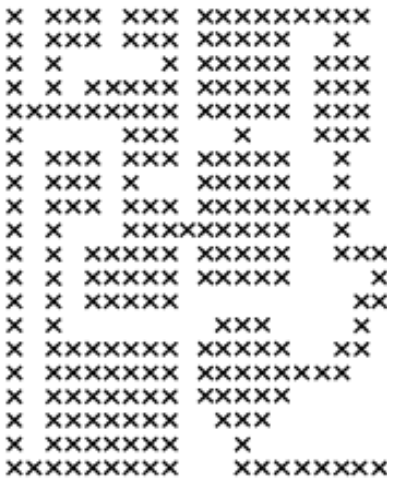
Cluster Heuristik



Euklidischer Abstand



Null-Heuristik



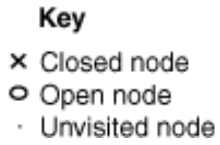
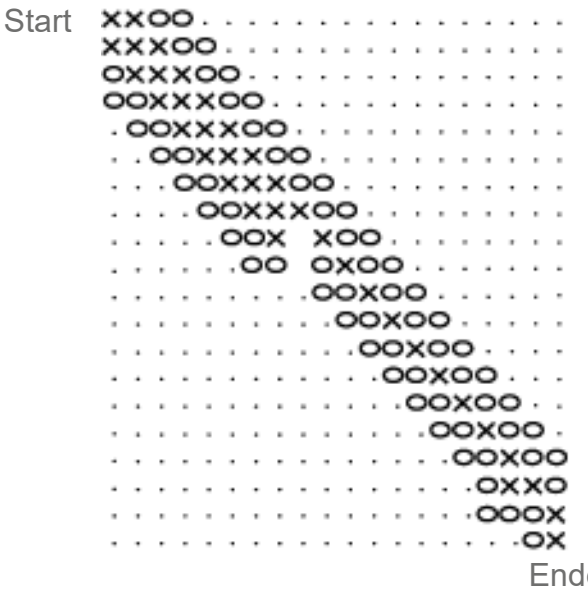


Abbildung aus:
I. Millington, J. Funge: Artificial Intelligence for Games, Morgan Kaufmann, 2. Auflage, 2009.

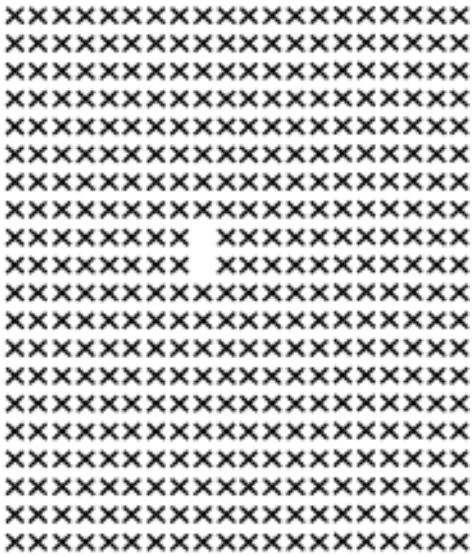
Technische Hochschule Rosenheim

KI: Wegplanung
Heuristiken - Freiland

Euklidischer Abstand



Null-Heuristik



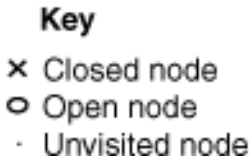


Abbildung aus:
I. Millington, J. Funge: Artificial Intelligence for Games, Morgan Kaufmann, 2. Auflage, 2009.

Technische Hochschule Rosenheim

KI: Wegplanung
Heuristiken – quadratische Gitter

- 4-Nachbarschaft:
  L1-Norm (Manhattan-Abstand)

$$h(n) = |x_Z - x_S| + |y_Z - y_S|$$

- 8-Nachbarschaft:
  L∞-Norm (Maximum-Norm)

$$h(n) = \max(|x_Z - x_S|, |y_Z - y_S|)$$

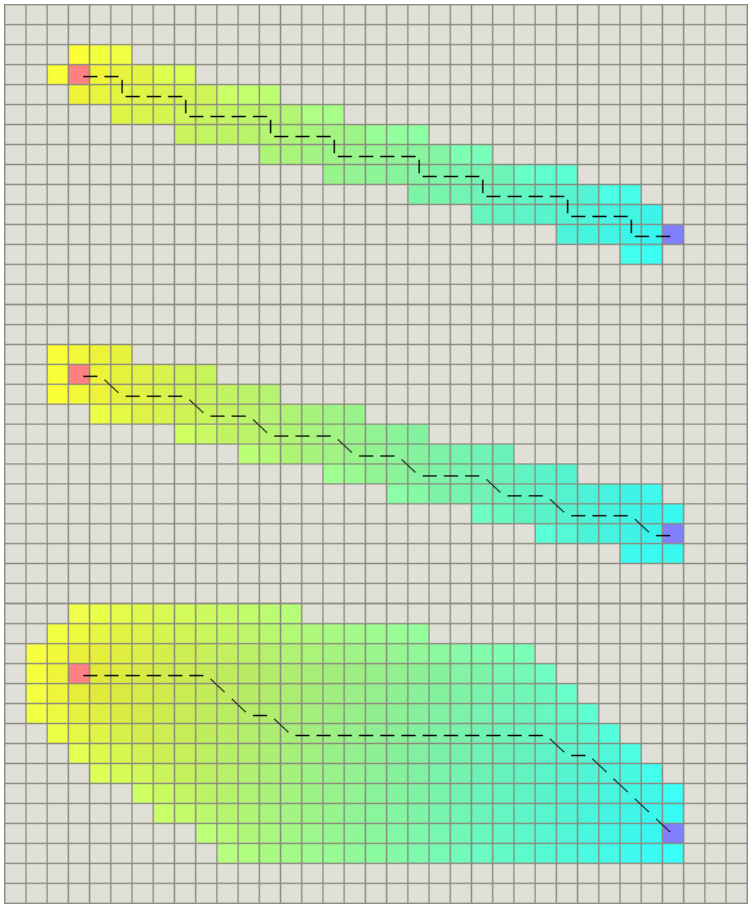
  L2-Norm (euklidischer Abstand)

$$h(n) = \sqrt{|x_Z - x_S|^2 + |y_Z - y_S|^2}$$

- Bewegung in beliebigem Winkel
  L2-Norm (euklidischer Abstand)

Technische Hochschule Rosenheim

KI: Wegplanung
Heuristiken – quadratische Gitter



Manhattan-Abstand

Maximum-Norm

euklidischer Abstand

Abbildungen aus: A. Patel: *Amit's Game Programming Information*, 2009, http://www-cs-students.stanford.edu/~amitp/gameprog.html

Technische Hochschule Rosenheim

KI: Wegplanung
Heuristiken - Anmerkungen

- Die Wahl der richtigen Heuristik hat erheblichen Einfluss
  - auf die Anzahl der expandierten Knoten
  - und damit auf die Leistungsfähigkeit der Suche
- Nicht immer ist der euklidische Abstand die beste Heuristik
- Wie kann man prüfen, wie gut die Heuristik ist?
  → erzeugen Sie Füllmuster, wie in den Beispielen vorher gezeigt
- Kosten
  - sind nicht immer Entfernungen
  - können stattdessen auch bewerten, wie schwer es ist (oder wie lange es dauert), von einem Punkt zum anderen zu gelangen
  - Beispiele:
    - 5m zu rennen geht schneller als 5m eine Leiter hochzuklettern
    - 20m über eine Brücke zu laufen geht schneller als 20m durch einen Fluss zu schwimmen
    - Bewegungen auf einer Straße sind einfacher/schneller als über einen Waldweg, diese wiederum schneller als durchs Unterholz
  - genauso z.B. Bewertung der Gefährlichkeit bestimmter Gebiete
  - die Performance von A* hängt **extrem** von der richtigen Bewertung der Kanten ab

  → sehr wichtiger Punkt

Technische Hochschule Rosenheim

KI: Wegplanung
Hierarchische Wegplanung

- Planung längerer Strecken besser hierarchisch
  macht auch der Mensch so
  ausgehend von einer abstrakten Planung auf hoher ebener
  geht es immer weiter nach unten auf detailliertere Pläne
- Idee:
  es gibt mehrere Graphen, auf verschiedenen Abstraktionsebenen

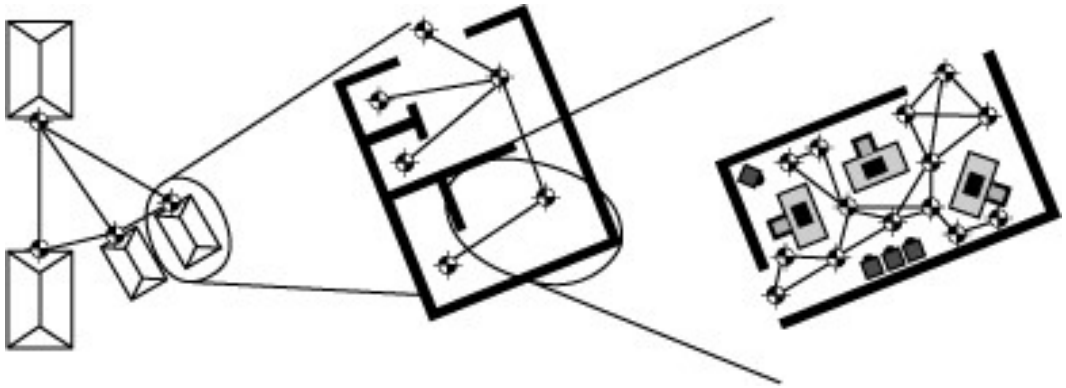


Abbildung aus:
I. Millington, J. Funge: Artificial Intelligence for Games, Morgan Kaufmann, 2. Auflage, 2009.

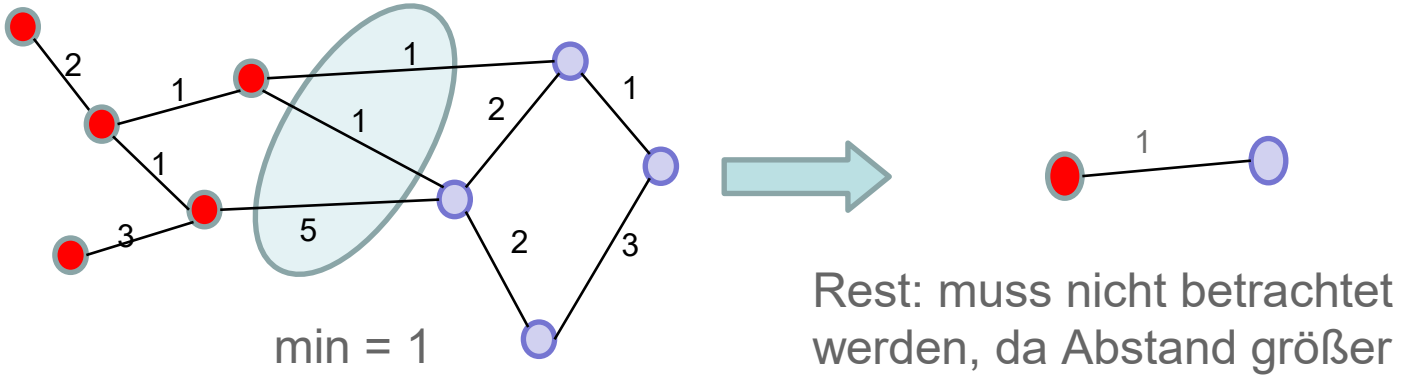
Technische Hochschule Rosenheim

KI: Wegplanung
Hierarchische Wegplanung - Heuristik

- wie sind die Kantengewichte auf höheren Ebenen zu wählen?
- verschiedene Heuristiken möglich, z.B.
  - Minimal-Abstand
  - Maximin-Abstand
  - durchschnittlicher Minimal-Abstand
- Achtung: alle Methoden machen irgendwann Fehler
  - allerdings an unterschiedlichen Stellen
  - sie finden also nicht den optimalen Pfad
  - dafür wird die Wegplanung erheblich beschleunigt

KI: Wegplanung
Hierarchische Wegplanung – Minimal Abstand

- Kosten zwischen 2 Gruppen = Kosten der kürzesten Verbindung zwischen 2 Knoten der beiden Gruppen
- optimistische Schätzung
  Annahme: Bewegung innerhalb der Gruppe kostet nichts
  Kosten sind normalerweise höher



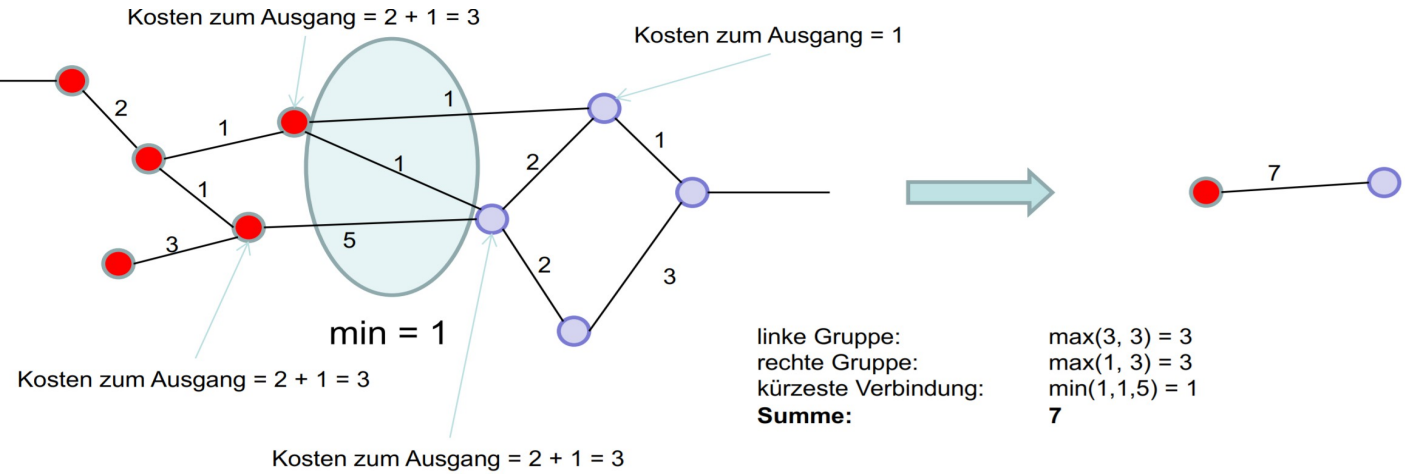
min = 1

Rest: muss nicht betrachtet werden, da Abstand größer wird

Abbildung aus EVC 2020, Prof. J. Schmidt

KI: Wegplanung
Hierarchische Wegplanung – Maximin Abstand

- Kosten zwischen 2 Gruppen
  - berechne kleinste Kosten zwischen jedem Eingangs- und Ausgangsknoten jeder Gruppe
  - nimm davon jeweils das Maximum
  - Addiere diese beiden Werte zur kürzesten Verbindung zwischen den Gruppen
- pessimistische Schätzung
  Annahme: Bewegung innerhalb der Gruppe kostet das maximal mögliche
  Kosten sind normalerweise geringer



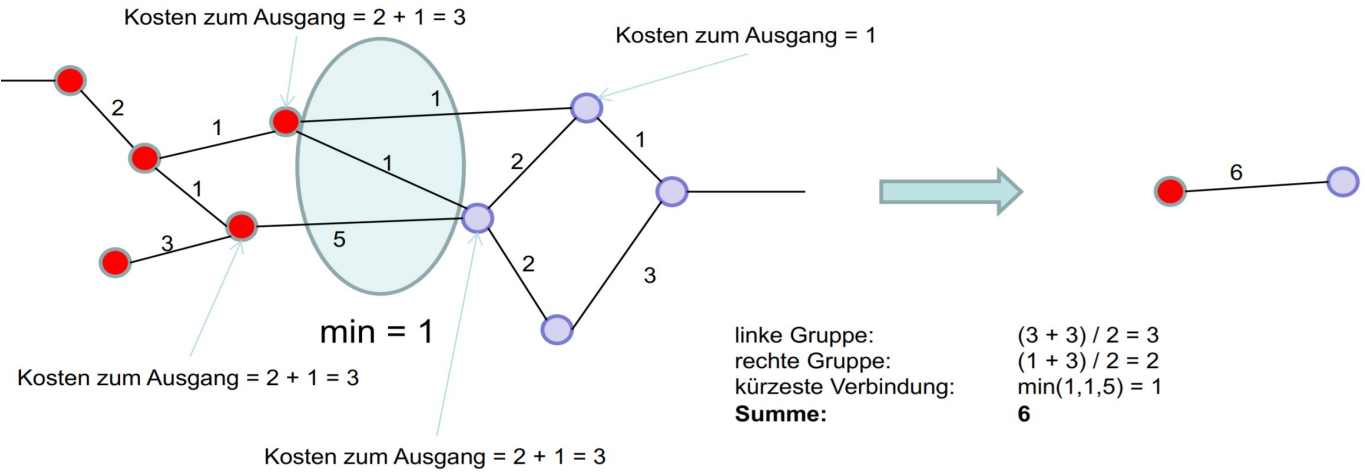
linke Gruppe:	max(3, 3) = 3
rechte Gruppe:	max(1, 3) = 3
kürzeste Verbindung:	min(1,1,5) = 1
Summe:	**7**

Abbildung aus EVC 2020, Prof. J. Schmidt

KI: Wegplanung

Hierarchische Wegplanung – durchschnittlicher Minimal Abstand

- Kosten zwischen 2 Gruppen
  - berechne kleinste Kosten zwischen jedem Eingangs- und Ausgangsknoten jeder Gruppe
  - berechne davon den Mittelwert
  - Addiere diese beiden Werte zur kürzesten Verbindung zwischen den Gruppen
- **pragmatische** Schätzung
  - im Mittel stimmen die Kosten



linke Gruppe:	(3 + 3) / 2 = 3
rechte Gruppe:	(1 + 3) / 2 = 2
kürzeste Verbindung:	min(1,1,5) = 1
Summe:	**6**

Abbildung aus EVC 2020, Prof. J. Schmidt

KI: Wegplanung
Hierarchische Wegplanung - Probleme

- Es wird nicht mehr unbedingt der optimale Pfad gefunden
- Beispiel: Minimal-Abstand
  ähnliche Beispiele findet man für andere Heuristiken

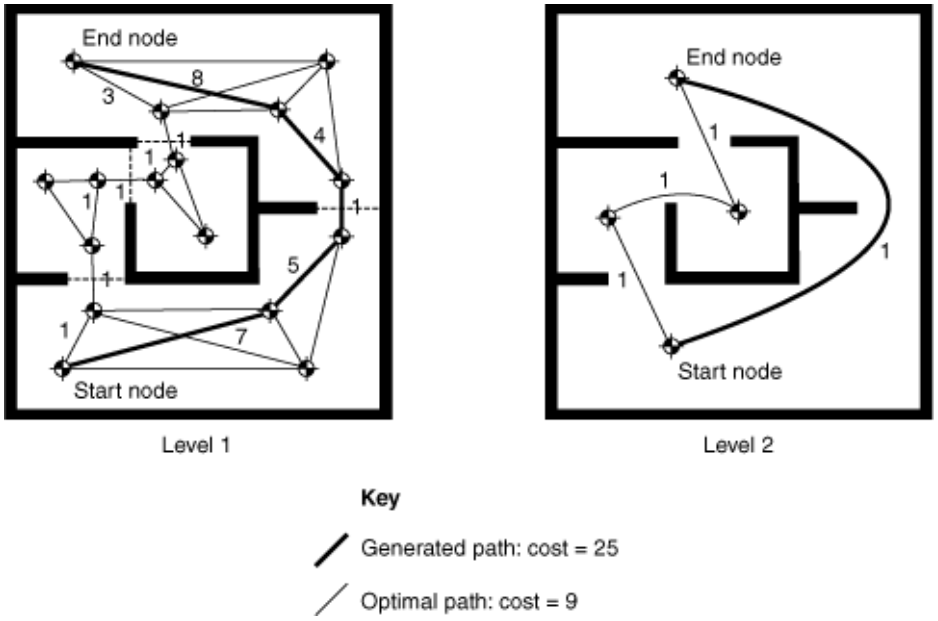


Abbildung aus:
I. Millington, J. Funge: Artificial Intelligence for Games, Morgan Kaufmann, 2. Auflage, 2009.

KI: Wegplanung
Hierarchische Wegplanung - Bewertung

- In Extremfällen (schlecht strukturierte Graphen) kann hierarchische Planung langsamer sein als nichthierarchische
- In der Praxis
  - mit mehreren Stufen und großen Graphen (zehntausende Knoten): signifikant schneller (bis Faktor 100)
  - laut [Mil09]: Wegplanung mit 100 Millionen Knoten in Echtzeit möglich

[Mil09]:
I. Millington, J. Funge: Artificial Intelligence for Games, Morgan Kaufmann, 2. Auflage, 2009.