Grundlagen der Künstlichen Intelligenz

Prof. Dr. Sibylle Schwarz
HTWK Leipzig, Fakultät IM
Gustav-Freytag-Str. 42a, 04277 Leipzig
Zimmer ZU 411 (Zuse-Bau)
https://informatik.htwk-leipzig.de/schwarz
sibylle.schwarz@htwk-leipzig.de

Wintersemester 2026/27

Was ist Künstliche Intelligenz?

EU-Factsheet on Artificial Intelligence
(`https://digital-strategy.ec.europa.eu/en/library/factsheet-artificial-intelligence-europe`)

> *Artificial intelligence (AI) refers to systems that show intelligent behaviour: by analysing their environment they can perform various tasks with some degree of autonomy to achieve specific goals.*
> *Mobile phones, e-commerce tools, navigation systems and many other different sensors constantly gather data or images. AI, particularly machine-learning technologies, can learn from this torrent of data to make predictions and create useful insights.*

Aussage über das derzeitige (eingeschränkte) Verständnis von KI

Können Maschinen denken?

Alan Turing 1950

Konkretisierung der Frage:
Können Maschinen denken?

zur überprüfbaren Frage:
Können Maschinen konstruiert werden, die einen speziellen Test bestehen?

Imitation Game

Imitation Game (Alan Turing 1950):

- zwei verschlossene Räume,
  in einem befindet sich Herr A, im anderen Frau B
- eine Person C (Frager) stellt Fragen, A und B antworten
- Kommunikation über neutrales Medium,
  an welchem das Geschlecht nicht erkennbar ist,
- C soll herausfinden, in welchem der Räume Frau B ist
- Herr A versucht, C irrezuführen
- Frau B kooperiert mit C

Herr A besteht den Test, wenn ihn C für Frau B hält.

Wie erkennt man Intelligenz: Turing-Test

Turing-Test 1950: verschiedene Versionen des Imitation Game

- A ist Machine statt Mann (B Person beliebigen Geschlechts)
- verschiedene Kooperationsverhalten von A und B

Vorschlag zur Bewertung natürlichsprachlicher Kommunikationsfähigkeiten

Beginn koordinierter Forschung zur Künstlichen Intelligenz

John McCarthy	Marvin Minsky	Claude Shannon
(1927-2011)	(1927-2016)	(1916-2001)
Programmiersprachen	Kognitionswissenschaft	Informationstheorie

stellten 1955 die Vermutung auf, dass

„jeder Aspekt des Lernens oder jedes anderen Ausdrucks von Intelligenz prinzipiell so präzise beschrieben werden kann, dass sich eine Maschine konstruieren lässt, die ihn simuliert. “

Begriff Künstliche Intelligenz

McCarthy formulierte das Ziel,

„herauszufinden, wie man Machinen konstruiert, die

- natürliche Sprache benutzen,
- Abstraktionen und Begriffe entwickeln,
- Aufgaben lösen, die (bis dahin) nur Menschen lösen konnten,
- sich selbst verbessern.“

und prägte dafür den Begriff Künstliche Intelligenz.

Beginn koordinierter Forschung zur Künstlichen Intelligenz

1956: erste Konferenz zur Künstlichen Intelligenz
Dartmouth Summer Research Project on Artificial Intelligence

Themen:

- Berechnungsmodelle in Computern
- Kommunikation mit Computern in natürlicher Sprache
- Berechenbarkeitstheorie
- Neuronale Netzwerke
- Selbst-Verbesserung
- Abstraktionen
- Zufälligkeit und Kreativität

Forschung zur Künstlichen Intelligenz

Momentaufnahme 2006:
Dartmouth Artificial Intelligence Conference: The Next Fifty Years

Themen:

- Modelle des (menschlichen) Denkens
- Neuronale Netzwerke
- (Maschinelles) Lernen und Suchen
- Maschinelles Sehen
- Logisches Schließen
- Sprache und Kognition
- KI und Spiele
- Interaktion mit intelligenten Maschinen
- Ethische Fragen und zukünftige Möglichkeiten der KI

KI-Entwicklung – einige Meilensteine

- 1943 abstraktes Modell künstlicher Neuronen
- 1945 frühe Schachprogramme (ohne Implementierung)
- 1955 Logic Theorist: automatischer Beweiser
- 1958 erster erfolgreicher Neurocomputer Mark I Perceptron
- 1961 General Problem Solver, löst z.B. Rätsel und Intelligenztests
- 1966 Chatbot Eliza (maschineller Psychotherapeut)
- 1972 erster mobiler Roboter
- ab ca. 1970 Beschränkung auf spezialisierte Expertensysteme
- 1976 MYCIN (Medizinisches Diagnosesystem)
- 1980 Dendral (Molekülstruktur aus Massenspektrogramm)
- 1982 XCON (Konfiguration von Computersystemen)
- ab ca. 1980 autonome Fahrzeuge, Expertensystem-Shells
- seit 1993 RoboCup Roboter-Fußball
- 1997 Deep Blue gewinnt gegen amtierenden Schach-Weltmeister
- 2011 Watson schlägt zwei Meister in Quizshow Jeopardy!
- 2012 Zulassung des ersten autonomen Fahrzeuges
- 2016 AlphaGo schlägt Go-Meister Lee Sedol
- 2022 Large Language Models (LLM, aus NLP)
  + Generative Pre-Trained Transformer (GPT, aus KNN)
  = Chatbot-Revival

Ansätze intelligenter Systeme

- Simulation menschlichen Verhaltens
  (Verständnis und eigenes Denken nicht notwendig)
  Modellierung von Kognition,
  statistische Verfahren, Training mit vielen Fällen
  Getroffene Entscheidungen werden nicht begründet.
  schwache (subsymbolische) künstliche Intelligenz

- Verstehen und Nachbilden des menschlichen Denkens
  (Verständnis und eigenes Denken notwendig)
  Denkmodelle, mentale Modelle als Grundlage
  logisches Schließen, Abstraktion
  Jede Entscheidungen kann nachvollziehbar begründet werden.
  starke (symbolische) künstliche Intelligenz

Kritik am Turing-Test

Kritik:

schwache KI genügt, um den Turing-Test zu bestehen

Searle (1980) Chinese-Room-Argument:
eine (nicht chinesisch verstehende) Person B in einem verschlossenen Raum mit einem (riesigen) Regelbuch mit chinesischen Fragen und passenden Antworten.

- A stellt Fragen, B antwortet.
- B antwortet mit Hilfe des Buches immer passend, ohne die Frage verstanden zu haben.

These: (anscheinend) intelligentes Verhalten ist noch keine Intelligenz, wenn Verständnis fehlt (Ansatz der starken KI)

außerdem: praktisch nicht umsetzbar (Datenmenge)

ELIZA

1966: Chatbot ELIZA (maschineller Psychotherapeut)
besteht Turing-Test

ELIZA-Entwickler Joseph Weizenbaum:
„The reaction to the program ELIZA showed me more vividly than anything I had seen hitherto the enormously exaggerated attributions an even well-educated audience is capable of making, even strives to make, to a technology it does not understand.“

Symbolische KI-Methoden

Wissensrepräsentation: formale Beschreibung von Umwelt (Randbedingungen) und Problem

Problemlöseverfahren: zur Lösung vieler Probleme anwendbares Standardverfahren (z.B. logisches Schließen)

Beispiele:

- Entscheidungsbäume und -tabellen
- Regelsysteme, Logiken, logisches Schließen
- Constraint-Systeme und -Löser
- deklarative Programmierung (logisch, funktional)
- fallbasiertes Schließen (durch Analogien)
- Simulation

typische Anwendungen klassischer KI-Methoden:

- Entscheidungsunterstützung (z.B. Finanzwirtschaft)
- Diagnosesysteme (z.B. in Medizin, Technik)
- Bewegungs- und Ablaufplanung

Subsymbolische KI-Methoden

oft besser geeignet für Probleme

- die unvollständig beschrieben sind,
- die keine eindeutige Lösung haben,
- für die keine effizienten Lösungsverfahren bekannt sind, usw.

einige Ansätze:

- künstliche neuronale Netze
- evolutionäre Algorithmen
- Schwarm-Intelligenz, Ameisen-Algorithmen
- Fuzzy-Logiken, probabilistische Logiken

Aktuelle Entwicklung

starker Fortschritt einiger KI-Methoden („Deep Learning“)
in den letzten 20 Jahren aufgrund der Entwicklung bei

- Computertechnik: Parallelrechner, GPU (70% Einfluss)
- Speichermöglichkeit großer Datenmengen, Verfügbarkeit großer strukturierter und annotierter Datenmengen (20%)
- neue Typen künstlicher neuronaler Netze, bessere Algorithmen (10%)

sowie starkem Medieninteresse an bestimmten Erfolgen, z.B.

- 1997 Deep Blue gewinnt gegen amtierenden Weltmeister
- 2011 Watson schlägt zwei Meister in Quizshow Jeopardy!
- 2012 erste Zulassung eines autonomen Fahrzeugs für den Test auf öffentlichen Straßen
- 2016 AlphaGo schlägt Go-Meister
- Unterhaltsames in Bilderkennung, -interpretation, -erzeugung
- 2022 Chatbot-Systeme, LLMs, z.B. ChatGPT
- ...

führt zum wiederholten Aufflammen der Euphorie bzgl. subsymbolischer KI-Methoden

Leistung aktueller (subsymbolischer) KI-Systeme

nahe und teilweise über den menschlichen Fähigkeiten z.B. bei

- Erkennung von Objekten in Bildern
- Einordnung / Klassifikation von Objekten und Situationen
- Reaktion auf klar erkannte Situationen
- strategischen Spielen mit endlichem Zustandsraum
  z.B. Schach, Go

prinzipielle Herausforderungen:

- Zuverlässigkeit, Sicherheit
- Begründung, Erklärung
- Kontext menschlichen Denkens (z.B. Ethik, Recht)
- individuelle / kulturelle Bezugssysteme
- mentale Modelle

Schwächen aktueller (subsymbolischer) KI-Systeme

KI derzeit noch weit von menschlichen Fähigkeiten entfernt bzgl.

- Erkennung der eigenen Grenzen
- Lernen ohne vorheriges Training mit großen Mengen (manuell) annotierter Daten
- Übertragen von Wissen zwischen verschiedenen Anwendungsbereichen
- Intuition
- Aufstellen und Überprüfen sinnvoller Annahmen bei unvollständig vorhandener Information
- Kombination verschiedener Methoden
- Schließen im Kontext menschlicher Bezugssysteme

Einordnung in die Informatik

Informatik Wissenschaft von der Darstellung und Verarbeitung symbolischer Information durch Algorithmen

Einordnung in die Teilgebiete der Informatik:

theoretisch

- Sprachen zur Formulierung von Information und Algorithmen,
- Berechenbarkeit durch Algorithmen,
- Grundlagen für technische und praktische (und angewandte) Informatik

Grundlagen, z.B. Logik, formale Sprachen

technisch

- maschinelle Darstellung von Information
- Mittel zur Ausführung von Algorithmen

Anwendung, z.B. technische Diagnose

praktisch Entwurf und Implementierung von Algorithmen
Grundlagen, z.B. Graph-Suchverfahren, Inferenzalgorithmen, Algorithmen zum Constaint-Lösen

angewandt Anwendung von Algorithmen, z.B.
Anwendung, z.B. KI, Spracherkennung, Bilderkennung, Suchmaschinen, autonome Agenten

Inhalt der Lehrveranstaltung

Motivation und Grundlagen:

- Daten, Information, Wissen
- Wissenbasierte Systeme (Aufgaben, Aufbau)

Wissensrepräsentations- und -verarbeitungsprinzipien:

- symbolische Verfahren:
  - Suche in Zustandsübergangssystemen (vollständig, heuristisch)
  - Klassifikation
  - Regelsysteme
  - Logiken (klassische und nichtklassische)
  - Logische Programme
  - Unvollständiges Wissen (nichtmonotones Schließen)
  - Entscheidungstabellen, -bäume und -diagramme
- subsymbolische Verfahren:
  - mehrwertiges / fuzzy / unscharfes Schließen
  - maschinelles Lernen
  - künstliche neuronale Netze

Literatur

Folien, Aufgaben, ... zur aktuellen Vorlesung unter
`https://informatik.htwk-leipzig.de/schwarz/lehre/ws26/kib`

Bücher:

- Ingo Boersch, Jochen Heinsohn, Rolf Socher:
  Wissensverarbeitung (Spektrum, 2007)
- Wolfgang Ertel:
  Grundkurs Künstliche Intelligenz (Springer, 2016)
  (elektronische Version in HTWK-Bibliothek)
- Ronald Brachman, Hector Levesque:
  Knowledge Representation and Reasoning
  (Morgan Kaufmann 2004)
- Stuart Russell, Peter Norvig:
  Künstliche Intelligenz (Pearson 2004)
- George Luger: Künstliche Intelligenz (Pearson 2001)

Organisation

5 ECTS (Präsenzzeit 56 h, Vor- und Nachbereitungszeit 94 h)

- wöchentlich
  - eine Vorlesung
  - Übungsaufgaben (ÜS + gelegentlich Autotool)
  - Übung je Gruppe
- Prüfungsvorleistung: Beleg
  - 3 Präsentationen der Lösung der Übungsaufgaben
    (je ein vor Ende November, Ende Dezember, Ende Januar)
  - 50 % aller Punkte für Autotool-Pflichtaufgaben
- Prüfung: Klausur 90 min
  Aufgabentypen wie in Übungsserien
  zulässiges Hilfsmittel: A4-Blatt, handbeschrieben

Daten, Wissen, Intelligenz

Umwelt		Reize, Eindrücke
Agent	Wahrnehmen, Beobachten	Daten
	Erkennen, Verstehen	Information
	Anwenden, Können	Wissen
	Lernen	Wissenserwerb (Intelligenz?)
	Reflektieren, Begründen, Erkennen der Grenzen, Verstehen	Intelligenz

Beispiel: Daten, Information, Wissen, Intelligenz

Daten	Darstellungsform (Syntax) Zeichenketten, Bilder, Ton, . . . (z.B 39.7)
Information	Bedeutung der Daten (Semantik) in einem bestimmten Kontext im Beispiel: Körpertemperatur= 39.7
Wissen	Information mit einem Nutzen, trägt zur Lösung eines Problemes bei, Nutzen abhängig von vorhandenem Kontextwissen im Beispiel: Kontext Körpertemperatur $>$ 39.0 ist Fieber, Fieber ist Symptom von Erkältungen oder Nebenwirkung einer Impfung oder ... mögliche Behandlung, z.B. Wadenwickel, Medikamente Diagnose und Auswahl aus Therapie-Alternativen speziell für die zu behandelnde Person
Wissenserwerb	selbständige Informationsgewinnung (auch zum Kontext) im Beispiel über (derzeit typische) Auslöser, Nebensymptome, neue Behandlungsmethoden, klinische Studien
Intelligenz	Überweisung zu Spezialisten an Grenzen eigener Expertise Reflexion, Erklärung, Entwicklung neuer Methoden

Explizites und implizites Wissen

explizites Wissen

z.B. Fakten, Aussagen, Zusammenhänge, Verfahren
symbolische Repräsentation ermöglicht
Anwendung symbolischer / logischer Verfahren

implizites Wissen

z.B. Fähigkeiten wie Laufen, Autofahren, Schachspielen
wird durch Training erworben,
(ggf. mit Hilfe expliziten Wissens, z.B. Spielregeln)
Nachbildung durch statistische Verfahren

Kommuniziert werden kann nur explizites Wissen.

Zur Kommunikation (in Raum und Zeit) ist
Transformation von implizitem in explizites Wissen notwendig.

Arten von Wissen

deklarativ über Zustände (der Welt) Fakten, Aussagen, Zusammenhänge, z.B.

- Fliegenpilze sind ungenießbar.
- Es existieren gerade Primzahlen.
- Eine Liste $(x_1, \ldots, x_n)$ ist genau dann aufsteigend sortiert, wenn sie leer ist oder ($x_1 \leq x_2$ und $(x_2, \ldots, x_n)$ aufsteigend sortiert ist).

prozedural über Zustandsübergänge Regeln, Algorithmen, Funktionen, z.B.

- Euklidischer Algorithmus
- aussagenlogisches Resolutionsverfahren
- Sortierverfahren
- Kochrezept

Ist die folgende Aussage Fakten- oder prozedurales Wissen?
Jedes Kind eines Kindes einer Person X ist ein Enkel von X.

Also: Repräsentationen von Regeln, Algorithmen und Funktionen lassen sich auch als Faktenwissen auffassen.

KI / Wissensverarbeitung – Methoden (Beispiele)

Suche / Planen:

- Kontext: Zustandsübergangssystem
- Aufgabe: Startzustand und Anforderungen an Zielzustände
- Lösung: Zielzustand (und evtl. Pfad dorthin)

Lösungsverfahren: Suche (vollständig oder heuristisch)

Inferenz, logisches Schließen:

- Kontext: Menge logischer Formeln
- Aufgabe: Gilt die Behauptung (logische Formel) im Kontext?
- Lösung: ja / nein (evtl. mit Begründung)

Lösungsverfahren: logisches Folgern oder Schließen

Statistische Klassifikation:

- Kontext: klassifizierte Datenmenge (bekannte Fälle)
- Aufgabe: neuer Fall
- Lösung: Zuordung zu einer Klasse / mehreren Klassen

Lösungsverfahren: statistische Verfahren (z.B. trainiertes KNN)

Was bisher geschah

- Definition KI (Versuche früher und jetzt)
- KI-Geschichte
- KI-Tests (Turing, Chinese Room)
- Daten, Information, Wissen, Intelligenz
- symbolische / subsymbolische KI
- explizites und implizites Wissen

Problemlösung durch Suche in Graphen – Beispiele

- Finden von Wegen in einem Graphen
  - Aufgabe:
    - gegeben: Graph G (Tafel)
    - gesucht: Weg (Pfad) in G von Knoten u zu Knoten v
  - Lösungsidee: Suche im Graphen
- Münzenstapelspiel (für eine Person)
  - Aufgabe:
    - gegeben: Stapel von n Münzen
    - gesucht: Zugfolge durch erlaubte Züge (zwei Münzen von einem Stapel nehmen und auf beide Nachbarn verteilen) bis zu einer Situation, in der kein Zug möglich ist
  - Lösungsidee:
    - Modellierung als Zustandsübergangssystem (Graph mit Markierungen)
    - Suche im Graphen
- 3 Krüge
  - Aufgabe:
    - gegeben: 3 volle Krüge mit Volumen 9l, 7l, 4l,
    - gesucht: genau 6l in einem der 3 Krüge
  - Lösungsidee: Zustände als Knoten eines Suchbaumes

Suchprobleme – Kontext, Aufgabe und Lösung

Darstellung von

Kontext (Zustandsübergangssystem)
als Graph $G = (V, E)$ mit

- Menge V von Zuständen (evtl. unendlich)
  oft beschrieben durch Eigenschaften
- (gerichtete) Kanten:
  mögliche Übergänge zwischen Zuständen
  Übergangsrelation $E \subseteq V \times V$

Aufgabe definiert durch

- Startzustand $s \in V$
- Menge $Z \subseteq V$ von Zielzuständen
  (oder Eigenschaften der Zielzustände)

Lösung: Folge von Zuständen
(Weg von einem Start- zu einem Zielzustand)
(Mitunter interessiert nur der erreichte Zielzustand.)

Entfaltung des Graphen zu einem Baum:

Pfade im Graphen = Knoten im Baum

Beispiel Missionare und Kannibalen

informale Problembeschreibung:

- Zu Beginn: 3 Missionare und 3 Kannibalen am linken Flussufer
- Ziel ist das Übersetzen aller Personen an das rechte Ufer.
- Links liegt ein Ruderboot, welches höchstens zwei Personen fasst.
- Sobald an einem Ufer mehr Kannibalen als Missonare sind, werden alle Missionare dort gefressen.

Aufgaben:

- formale Darstellung
  - Zustände
  - Übergänge
  - Zielbedingung
- Heuristiken
- geeignete Suchverfahren, Verbesserungsmöglichkeit

Problemlösen durch Suchen

- formale Darstellung des Problemes
  als Graph (z.B. Baum, DAG)
- formale Beschreibung der Lösung als Eigenschaft von
  - Pfaden im Graphen
  - Knoten im Baum

Möglichkeiten zum Problemlösen:

- Pfadsuche im Graphen
- Knotensuche im Baum

Suche in Graphen

(aus dem Modul Algorithmen und Datenstrukturen schon bekannte) Verfahren zur Suche in Graphen (und Bäumen):

Tiefensuche (depth-first search):
Suche zuerst in Teilbäumen eines noch nicht besuchten Nachbarn des aktuellen Knotens

Breitensuche (breadth-first search):
Suche zuerst in Teilbäumen eines noch nicht besuchten Knotens mit der geringsten Tiefe

Allgemeines Suchverfahren

Daten:
- L_a Menge der noch zu expandierenden Knoten (Agenda)
- L_x Menge der expandierten Knoten
- s Startknoten
- φ Anforderungen an Lösung (Eigenschaft der Zielknoten)

Allgemeiner Suchalgorithmus:

1. $L_a = \{s\}, L_x = \emptyset$
2. solange $L_a \neq \emptyset$:
   - 2.1 Wähle auf festgelegte Art einen Knoten u in L_a und verschiebe u aus L_a in L_x.
   - 2.2 Füge alle Nachbarn v von u mit $v \notin L_a \cup L_x$ und $v \notin \text{Mod}(\varphi)$ auf eine festgelegte Art in L_a ein.
     (falls für ein v gilt $v \in \text{Mod}(\varphi)$: Abbruch mit Lösung v)

prominente Spezialfälle:

Tiefensuche
- Wähle Konten, der zuletzt in L_a eingefügt wurde
- Einfügen der Nachbarn an den Anfang der Liste L_a
- Verwaltung von L_a als Stack

Breitensuche
- Wähle Konten, der zuerst in L_a eingefügt wurde
- Einfügen der Nachbarn an das Ende der Liste L_a
- Verwaltung von L_a als Queue

Eigenschaften von Suchverfahren

Korrektheit: jede gefundene Lösung erfüllt die Lösungsbedingung

Vollständigkeit: jede Lösung wird gefunden

Fairness: jeder Knoten wird expandiert

Optimalität: (eine) beste Lösung wird zuerst gefunden

Zeitaufwand: (im ungünstigsten Fall) größtmögliche Anzahl der Schritte von einem Start- zu einem Endzustand

(Speicher-)Platzbedarf: (im ungünstigsten Fall) größtmögliche Länge der Anzahl der noch nicht expandierten Knoten in der Agenda

Laufzeit für Baum mit maximaler Tiefe t und maximalem Verzweigungsgrad d:

- Breitensuche: vollständig,
  (optimal, falls beste Lösung = geringe Tiefe)
  großer Zeitaufwand: $O(d^t)$
  großer Platzbedarf: $O(d^t)$
- Tiefensuche: unvollständig bei Bäumen mit unendlich langen Zweigen, nicht optimal
  großer Zeitaufwand: $O(d^t)$
  geringerer Platzbedarf: $O(td)$

Suche mit schrittweiser Vertiefung

beschränkte Tiefensuche:

1. festgelegte Tiefenbeschränkung $m \in \mathbb{N}$
2. Tiefensuche auf allen Pfaden bis zur Tiefe m

nicht vollständig, weiter entfernte Lösungen werden nicht gefunden

Schrittweise Vertiefung (iterative deepening)
Kombination aus Breiten- und Tiefensuche durch Nacheinanderausführung der beschränkten Tiefensuche für alle $m \in \mathbb{N}$, solange keine Lösung gefunden wurde

vollständig, optimal
(asymptotischer) Zeit- und Platzbedarf wie Tiefensuche

Gleiche-Kosten-Suche (kleinste bisherige Kosten)

(uniform-cost-search)

bei Zustandsübergängen mit verschiedenen Kosten

Ziel: Lösung (Pfad vom Start- zu einem Lösungsknoten) mit möglichst geringen Pfadkosten
(Pfadkosten = Summe der Kosten aller Übergänge auf dem Pfad)

Kostenfunktion für Knoten $k : V \to \mathbb{R}_{\geq 0}$

$k(u)$ = minimale (bisher entdeckte) Pfadkosten vom Startknoten zu u

Datenstruktur zur Verwaltung von L_a: Priority Queue
Priorität eines Knotens u: $k(u)$

prominente Spezialfälle:

- Breitensuche (Kosten = Tiefe des aktuellen Knotens u)
- Dijkstra-Algorithmus zur Bestimmung kürzester Wege (bekannt aus Algorithmen und Datenstrukturen)
  Kosten = minimale bisher bekannte Kosten vom Startknoten zum aktuellen Knoten u

Beispiel Schiebefax

- Zustände $u \in \{0, \ldots, 8\}^{3\times 3}$, 3×3-Matrix mit Einträgen $\{0, \ldots, 8\}$ (jede Zahl genau einmal, 0 bezeichnet Lücke / leeres Feld)
- Zulässige Züge: Verschieben des leeren Feldes auf ein Nachbarfeld d. h. Vertauschen von 0 und einem Wert in einem Nachbarfeld (gleicher Zeilen- oder Spaltenindex)
- Zielkonfiguration

1	2	3
8	0	4
7	6	5

- Aufgabeninstanz: gegebene Ausgangskonfiguration (Matrix), z.B.

8	0	3
2	1	4
7	6	5

- Lösung: Folge von zulässigen Zügen (Bewegung der Lücke 0) von der Ausgangs- zur Zielkonfiguration
- Bewertung der Lösung: Anzahl der Züge (Länge der Lösungsfolge)

Heuristische Suche – Motivation

Heuristik: Effizienzsteigerung durch Zusatzinformationen
(z.B. Erfahrungswerte)

Anwendung bei

- falls vollständige Suche zu aufwendig
- Aufgaben mit mehreren Lösungen (z.B. Wege in Graphen)
- unterschiedliche Qualität der Lösungen
  (z.B. Länge des Weges)
- Suche nach optimalen Lösungen (z.B. kürzester Weg)

Ziele:

- Wahl einer geeigneten Such-Reihenfolge, unter welcher gute Lösungen zuerst gefunden werden
- Verwerfen von Knoten, die wahrscheinlich nicht zu einer Lösung führen
  (beabsichtigte Verletzung der Fairness-Eigenschaft)

Schätzfunktionen

Ziel: sinnvolle Auswahl der in jedem Schritt zu expandierenden Knoten unter Verwendung von Zusatzinformationen

Schätzfunktion (heuristische Funktion) $h : V \to \mathbb{R}_{\geq 0} \cup \{\infty\}$
(oder in eine andere geordnete Menge)
Schätzung der erwartete Restkosten vom Knoten u bis zum Ziel

h repräsentiert die Zusatzinformation

ohne Zusatzinformation: uninformierte (blinde) Suchverfahren, z.B. Tiefensuche, Breitensuche, Dijkstra

mit Zusatzinformation: informierte Suchverfahren
heuristische Suche

Schiebefax – Heuristische Funktionen

Heuristische Funktionen $h_i : \{0, \ldots, 8\}^{3\times 3} \to \mathbb{N}$ mit

- h_1 Anzahl der Zahlen, die sich nicht an ihrer Zielposition befinden
- h_2 weitester Abstand einer Zahl zu seiner Zielposition
- h_3 Summe der Manhattan-Abstände jeder Zahl zu seiner Zielposition

Tafel: Bestensuche mit Bewertungsfunktionen $f(u) = h_i(u)$

Qualität der Schätzfunktionen:

- gute Trennung verschiedener Zustände
- fair: zu jedem $n \geq 0$ existieren nur endlich viele $u \in V$ mit $h(u) \leq n$

Eigenschaften von Heuristiken

Schätzfunktion $h : V \to \mathbb{R}_{\geq 0} \cup \{\infty\}$ heißt

perfekt (Schätzfunktion $H(u)$), gdw. $\forall u \in V : H(u) =$ genau die Kosten einer optimalen Lösung durch u ($H(u) = \infty$, falls keine Lösung über u existiert)

zielerkennend gdw. für jeden Lösungsknoten $u \in V$ gilt $h(u) = 0$

sicher gdw. aus jedem Knoten $u \in V$ mit $h(u) = \infty$ ist kein Lösungsknoten erreichbar
d.h. $\forall u : (h(u) = \infty \to H(u) = \infty)$

konsistent gdw. für jeden Knoten $u \in V$ und alle Folgeknoten v von u gilt $h(u) \leq w(u, v) + h(v)$
($w(u, v)$ Kosten des Übergangs von u nach v)

nicht-überschätzend gdw. für jeden Knoten $u \in V$ gilt $h(u) \leq H(u)$

Aus nicht-überschätzend folgt sicher und zielerkennend. (ÜA)
Aus zielerkennend und konsistent folgt nicht-überschätzend. (ÜA)

Was bisher geschah

- Daten, Information, Wissen
- Wissensrepräsentation und -verarbeitung

Repräsentation:

- Zustandsübergangssystem:
  Graph mit markierten Knoten
  (Zustände und deren Eigenschaften)
- Startzustand
- Eigenschaften der Zielzustände

Lösung: Pfad vom Start- zu einem Zielzustand

Verarbeitung: Suche im Graphen

uninformiert: Breiten-, Tiefen-, Gleiche-Kosten-Suche

informiert: Heuristik

Bidirektionale Suche (nach Lösungswegen, Plänen)

- simultane Suche ab Startknoten und ab Zielknoten
  Vorwärtssuche mit L_{xs}, L_{as}, Rückwärtssuche mit L_{xg}, L_{ag}
- Lösung (Pfad $p(s, g)$ von Start s zu Ziel g) gefunden, wenn ein Zustand u von s und g erreichbar ist (also bei beiden Suchen entdeckt wurde)
  Lösung $p(s, g) = p(s, u) \circ p(g, u)^{-1}$
- Bidirektionale Suche endet, wenn sich die „Grenzen“ der durch die Suche bisher entdeckten Mengen überschneiden
  $((L_{xs} \cup L_{as}) \cap (L_{xg} \cup L_{ag}) \neq \emptyset)$

Speicherbedarf geringer als bei Breitensuche

- nur sinnvoll, wenn eindeutiger (gesuchter) Zielzustand bekannt ist, z.B. bei Kannibalen-Missionare-Rätsel, Navigation
- Erweiterung auf endliche Mengen explizit gegebener Zustände möglich (Betrachtung von Zustandsmengen in Suchknoten)
- meist ungeeignet, wenn Zielzustände durch zu erfüllende Bedingung definiert sind
  (z.B. Spiele mit Zielbedingung wie Schach-Matt, kein Zug möglich), mehreren Zielzuständen (gleicher oder verschiedener Güte)

Besten-Suche

(best-first-search)

Allgemeines Suchverfahren mit Bewertungsfunktion
$f : V \to \mathbb{R}_{\geq 0} \cup \{\infty\}$
mit folgender Strategie zur Auswahl der in jedem Schritt zu expandierenden Knoten:

- Knoten werden aufsteigend nach Bewertung $f(u)$ expandiert,
- Expansion des Knotens u mit dem geringsten Wert $f(u)$ zuerst
- Verwaltung von L_a als priority queue

Beispiel: Suche eines kürzesten Weges zwischen Orten A und B

- Bewertungsfunktion $f(u)$: bisherige Kosten bis zum Ort u (ohne Schätzfunktion, uniforme Kostensuche, Dijkstra)
- Bewertungsfunktion $f(u)$:
  Luftlinienentfernung des Ortes u von B (nur Schätzfunktion)

Besten-Suche – Eigenschaften

zwei Methoden:

1. Knoten mit großen Werten möglichst spät expandieren
2. Knoten mit großen Werten nicht expandieren

- Bestensuche mit einer beliebigen Bewertungsfunktion ist nicht immer optimal.
- Bestensuche nach Methode 1 (fair) ist vollständig.
- Bestensuche nach Methode 2 ist nicht immer vollständig.

Greedy-Suche (kleinste geschätzte Restkosten)

Idee: Suche zuerst in Teilbäumen der noch nicht besuchten Knoten mit den geringsten (geschätzten) noch aufzuwendenden Kosten

Heuristische Funktion $h : V \to \mathbb{R}_{\geq 0} \cup \{\infty\}$

$h(v)$ ist Abschätzung des von Knoten v aus den noch notwendigen Kosten zum Erreichen eines Zielzustandes

Greedy-Suche:
Besten-Suche mit Bewertungsfunktion $f : V \to \mathbb{R}_{\geq 0} \cup \{\infty\}$,
wobei für jeden Knoten $v \in V$ gilt

$$f(v) = h(v)$$

Eigenschaften der Greedy-Suche:

- optimal?
- vollständig?

Bisherige Kosten

Kostenfunktion $k : V \to \mathbb{R}_{\geq 0}$

$k(u)$ Kosten des besten (bisher bekannten) Pfades vom Startzustand zum Zustand u

Kostenfunktion $k : V \to \mathbb{R}_{\geq 0}$ heißt

streng monoton wachsend , falls für alle Knoten u und alle Nachfolger v von u gilt $k(u) < k(v)$

Beispiele für Kostenfunktionen:

- Tiefe des Knotens im Suchbaum,
- maximale Entfernung vom Startknoten

A*-Suche (kleinste Gesamtkosten)

Idee: Suche (zuerst / nur) in Teilbäumen der noch nicht besuchten Knoten mit dem geringsten Wert der Schätzfunktion
(Summe von bisherigen und geschätzen zukünftigen Kosten)
Funktionen

- $k : V \to \mathbb{R}_{\geq 0}$ – geringste bisher bekannte Kosten von einem Startzustand zu v
- $h : V \to \mathbb{R}_{\geq 0}$ – geschätzte Kosten von v zu einem Endzustandr Lösung

A*-Suche: Besten-Suche mit Schätzfunktion $f : V \to \mathbb{R}_{\geq 0}$, wobei für jeden Knoten $v \in V$ gilt

$$f(v) = k(v) + h(v)$$

Eigenschaften der A^*-Suche: Wann optimal? Wann vollständig?

IDA*-Suche: Kombination von

- schrittweiser Vertiefung (iterative deepening)
- A*-Suche

Anwendungen

Planungsprobleme und kombinatorische Suchprobleme, z.B.

- Routenplanung
- Navigation (z.B. autonomer Fahrzeuge)
- TSP in Graphen
- Verlegen von Leitungen
- Schaltkreis-Layout
- Scheduling
- Produktionsplanung

Zwei-Personen-Spiele

Zwei-Personen-(Brett)spiel

- aktueller Spielzustand immer für beide Spieler sichtbar (vollständige Information)
- einer gewinnt, der andere verliert (Nullsummenspiel)

Repräsentation (Spielbaum):

- Menge von Zuständen (Min- und Max-Zustände)
- Startzustand
- Endzustände (ohne Fortsetzung)
- Nachfolgermenge $S(v)$ = Menge von Zuständen (nach zulässigen Zügen)
- Bewertungsfunktion: Menge der Endzustände $\rightarrow \mathbb{Z}$
  - positiv: Spieler (1, Max, beginnt) gewinnt
  - negativ: Gegner (0, Min) gewinnt

Beispiel Nim (Variante)

- n Münzen auf einem Stapel
- Spielzug: Teilen eines Stapels in zwei nichtleere Stapel ungleicher Größe
- Sobald ein Spieler keinen Zug mehr ausführen kann, hat er verloren (und der andere gewonnen).

neutrales Spiel: für jeden Spieler sind dieselben Spielzüge möglich

(eine mögliche) Modellierung als Zustandsübergangssystem:

Zustände: $S : \mathbb{N} \to \mathbb{N}$ (Multimenge)
Münzanzahl $\mapsto$ Anzahl der Stapel mit dieser Zahl an Münzen

Startzustand: $S(n) = 1 \wedge \forall i \neq n : S(i) = 0$

Endzustände: kein Zug möglich

Übergänge: (erlaubte Züge) für $x = x_1 + x_2 \wedge x_1 \neq x_2 \wedge x_1 x_2 \neq 0$:
$S \to S'$ mit
$S'(x) = S(x) - 1$
$\wedge S'(x_1) = S(x_1) + 1 \wedge S'(x_2) = S(x_2) + 1$
$\wedge \forall i \in \mathbb{N} \setminus \{x, x_1, x_2\} : S'(i) = S(i)$

Minimax-Werte in vollständigen Spielbäumen

- vollständiger Spielbaum $B = (V, E)$
- Bewertung der Endzustände (Blätter im Spielbaum) bekannt
- Fortsetzung der Bewertungsfunktion von den Blättern auf alle Knoten im Spielbaum $b : V \to \mathbb{Z}$

rekursive Berechnung (Minimax-Algorithmus) des Wertes eines Knotens v im Spielbaum:

$$m(v) = \begin{cases} b(v) & \text{falls } v \text{ Endzustand} \\ \max\{m(u) \mid u \in S(v)\} & \text{falls } v \text{ Max-Knoten} \\ \min\{m(u) \mid u \in S(v)\} & \text{falls } v \text{ Min-Knoten} \end{cases}$$

Beispiele (Tafel):

- Spielbaum,
- Nim mit $n = 7$

Spielstrategie für Spieler 1 (Max):
Zug wählen, der zum Zustand mit höchstem Minimax-Wert führt

Minimax-Werte mit Heuristik

bei unvollständigem Spielbaum: Kombination von

- heuristischer Knotenbewertung
- Berechnung der Minimax-Werte

Beispiele (Tafel): Tic-Tac-Toe
kein neutrales Spiel: jeder Spieler darf nur sein Symbol setzen
(wie z.B. auch Gomoku, Schach, Go)
mit Schätzfunktion für den Spieler am Zug:
Differenz der Anzahlen der noch nicht blockierten Gewinntripel

auch dabei Spielstrategie für Spieler 1 (Max):
Zug wählen, der zum Zustand mit höchstem Minimax-Wert führt

α-β-Suche

Berechnung der Minimax-Werte in Teilbaum beschränkter Tiefe durch

- Tiefensuche mit zusätzlicher Verwaltung eines Intervalles $[\alpha(u), \beta(u)]$ für jeden Knoten u
- Minimax-Wert $w(u)$ liegt immer im Intervall $[\alpha(u), \beta(u)]$
- zu Beginn für alle Knoten u: $[-\infty, \infty]$ ($\alpha(u) = -\infty$, $\beta(u) = \infty$)
- schrittweise Annäherung der Intervallgrenzen an $w(u)$

α-β-Pruning:
Bei Berechnung des Minimax-Wertes des Knotens u Berechnungen für alle Enkel von u auslassen (abtrennen), wenn diese die Intervallgrenzen $\alpha(u)$ und $\beta(u)$ nicht verbessern können
Abtrennen jedes Kindes v eines

Min-Knotens u, falls $\beta(u) \leq \alpha(v)$
(Min kann in u durch Wahl eines zuvor untersuchten Kindes v' den geringeren Minimax-Wert $\beta(u)$ erreichen als durch Wahl von v)

Max-Knotens u, falls $\alpha(u) \geq \beta(v)$
(Max kann in u durch Wahl eines zuvor untersuchten Kindes v' den höheren Minimax-Wert $\alpha(u)$ erreichen als durch Wahl von v)

Automatische Berechnung heuristischer Funktionen

Ziel: automatische Bewertung von Spielzügen, d.h.
Bewertung von Knoten v (Spielsituationen) im Spielbaum
Beispiel: Monte-Carlo-Baum-Suche MCTS
Idee: Berechnung des Wertes für v aus simulierten Spielen

- simuliertes Spiel i: Folge von (zufälligen) Zügen bis Spielende
- Bewertung des Knotens v im Spielbaum durch
  - n simulierte Spiele, beginnend in v
  - $\forall i \in \{1, \ldots, n\}$: R_i = Ergebnis des Spieles i
  - Berechnung des Wertes von v aus Ergebnissen $\{R_1, \ldots, R_n\}$ z.B. Mittel, Gewinnwahrscheinlichkeit

übliche Modifikationen:

- Integration von spezifischem Wissen (z.B. Standard-Antworten, Eröffnungsbibliotheken) statt ausschließlich zufälliger Züge
- Spiele nur bis festgelegter Anzahl von Zügen
- Backpropagation: Update der Werte auf Pfad zur Wurzel
- Speichern und laufendes Anpassen schon berechneter Bewertungen von Spielsituationen über mehrere Spiele (Lernen)

Beispiele: Monte-Carlo Go (1993), AlphaGo (erfolgreich 2016/17)