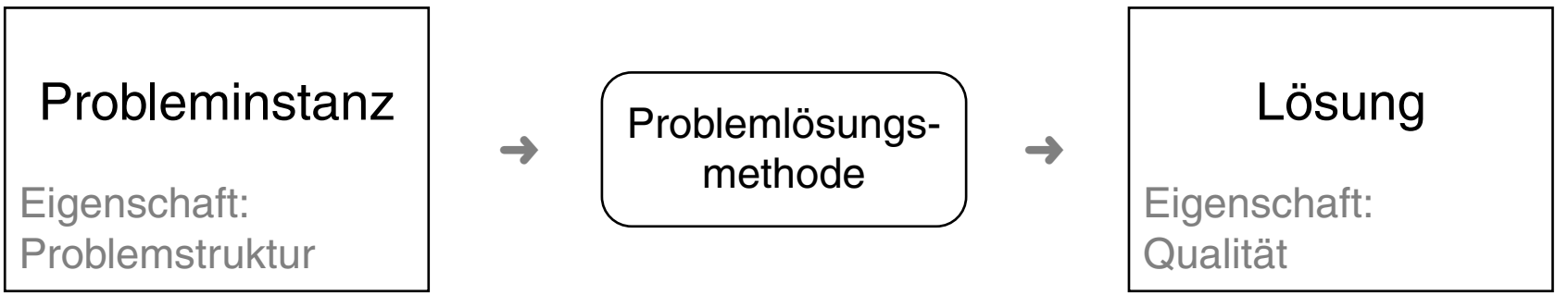
Kapitel ADS:II

II. Algorithm Engineering

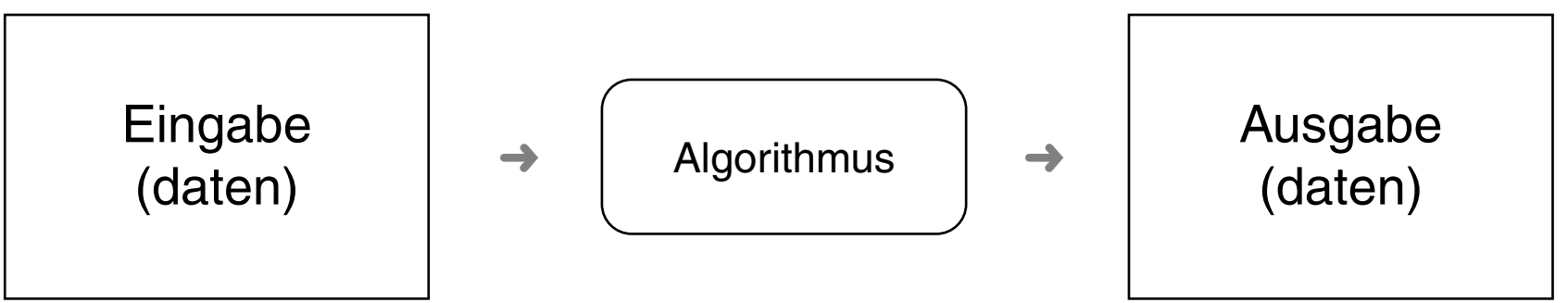
- Problemlösen
- Phasen des Algorithm Engineering
- Exkurs: Programmiersprachen
- Pseudocode
- Rekursive Algorithmen
- Algorithmenanalyse
- Algorithmenimplementierung
- Algorithmenevaluierung

Problemlösen



System in der realen Welt

Modell im Computer (Repräsentation)



Problemlösen

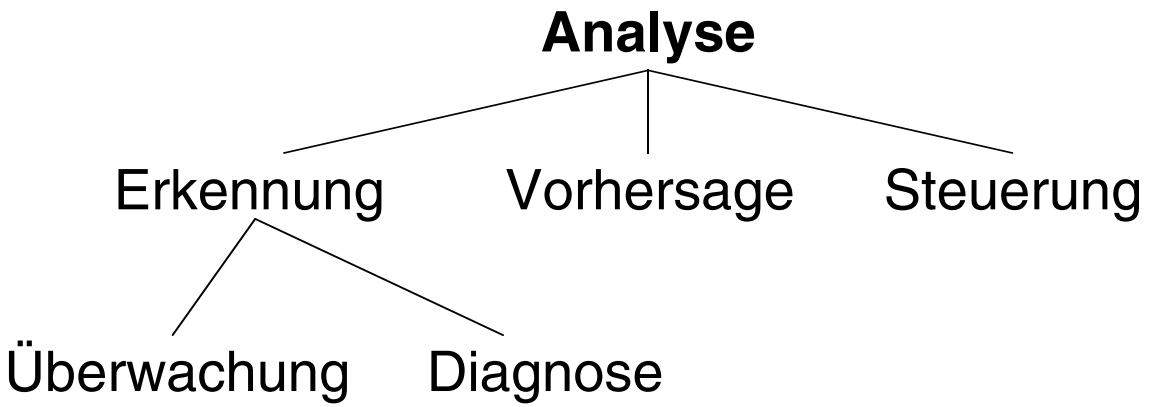
Problemklassen [Clancey 1985]

Analyse

Synthese

Problemlösen

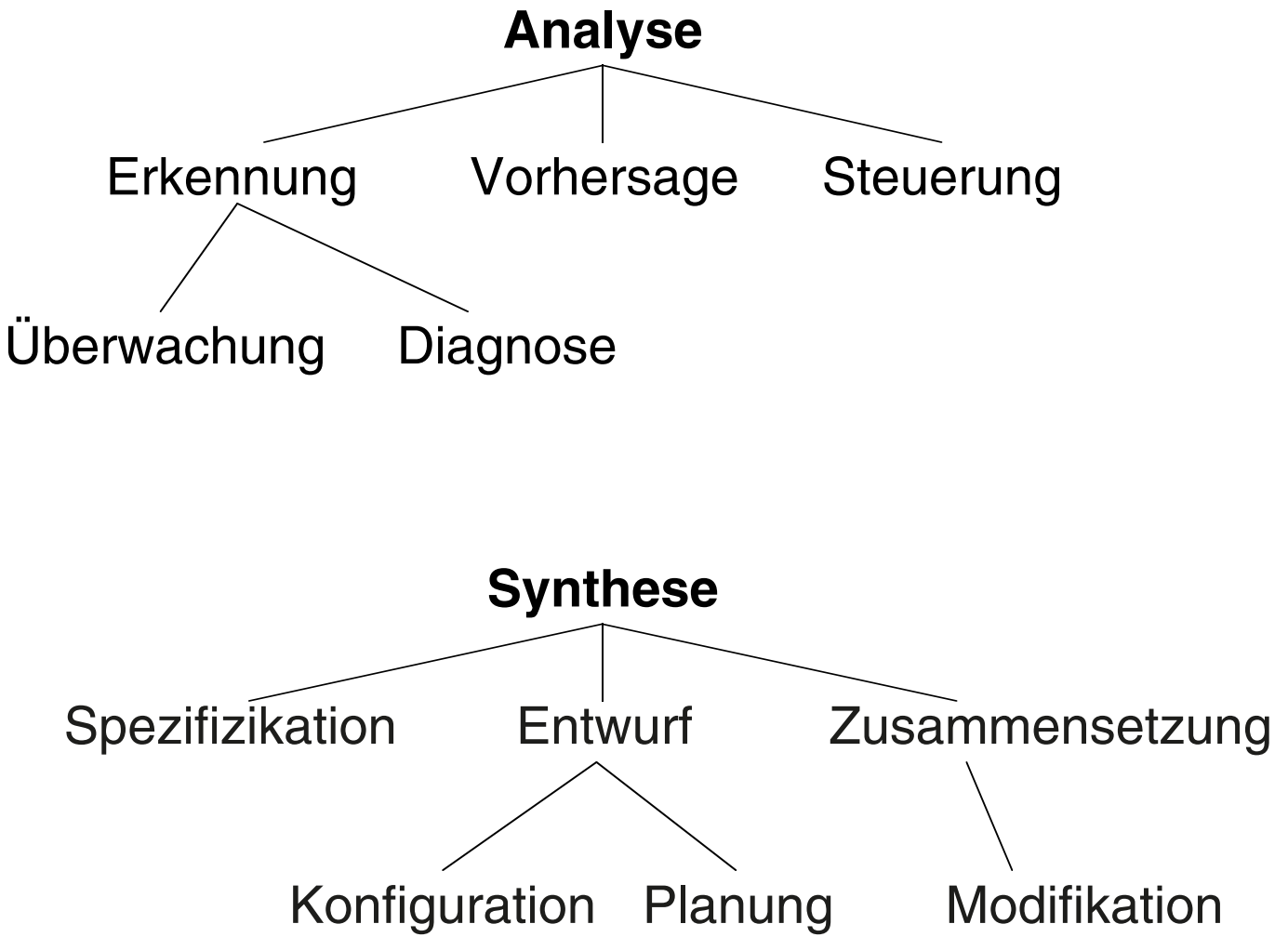
Problemklassen [Clancey 1985]



Synthese

Problemlösen

Problemklassen [Clancey 1985]



Bemerkungen:

- Eine Analyse (griech. „Auflösung“) ist eine systematische Untersuchung, bei der das untersuchte Objekt in seine Elemente (Bestandteile) zerlegt wird.
- Als Synthese (griech. „Zusammensetzung“) bezeichnet man die Vereinigung von zwei oder mehr Elementen (Bestandteilen) zu einer neuen Einheit.
- Clancey nimmt eine System-orientierte Sicht ein, bei der ein System definiert wird als Komplex interagierender Objekte, das einen Prozess mit Eingabe und Ausgabe realisiert.

  Die obigen Problemklassen drücken aus, was man mit einem System machen oder wie man es beeinflussen kann.
- Eine Analyseproblem zu lösen, bedeutet, Fragen bezüglich eines Systems zu beantworten.
- Die Analyseproblemklassen können unter der Systemsicht als Probleme verstanden werden, deren Lösung je eines der drei Dinge Eingabe, Ausgabe, oder System beinhaltet:
  - Identifizierung: Abbildung von Eingabe-Ausgabe-Paaren auf ein (un)bekanntes System.
  - Vorhersage: Ermittlung der Ausgabe eines (un)bekannten Systems für eine Eingabe.
  - Steuerung: Ermittlung von Eingaben für ein (un)bekanntes System, die zu erwünschten Ausgaben führen.
  - Überwachung: Erkennung unerwarteten Verhaltens eines (un)bekannten Systems.
  - Diagnose: Erklärung unerwarteten Verhaltens eines bekannten Systems.

Bemerkungen: (Fortsetzung)

- Ein Syntehseproblem zu lösen, bedeutet, Fragen bezüglich *einer Menge* von Systemen zu beantworten.
- Die Syntheseproblemklassen betreffen die Generierung eines neuen Systems:
  - Spezifikation: Sammlung von Anforderungen und Einschränkungen an ein System.
  - Entwurf: Konzeptuelle Beschreibung des Systems bezüglich räumlicher und zeitlicher Interaktionen (Konfiguration). Erstellung von Anweisungen zur Zusammensetzung (Planung).
  - Zusammensetzung: Umsetzung der Planung zur physischen Realisierung des Systems.
  - Modifikation: Anpassung des Systems aufgrund von Fehlverhalten oder Anforderungsänderungen.

Problemlösen

Problemlösungsstrategien

1. Generate and test / blind search

   Betrachte einen Lösungskandidaten nach dem anderen, bis eine adäquate Lösung gefunden ist und/oder alle Kandidaten betrachtet wurden.

   Beispiele: Verschlüsselung knacken; Suche in unsortierten Objekten

2. Hill climbing / local search

   Verbessere schrittweise einen initialen Lösungskandidaten bestmöglich, bis keine Verbesserung mehr möglich ist.

   Beispiele: Wechselgeld mit minimaler Zahl von Münzen herausgeben; Planung des Verlegens von Telefon-/Stromkabeln mit minimalen Kosten.

3. Informed search

   Verbessere schrittweise alternative Lösungskandiaten, und zwar zunächst den, der voraussichtlich am schnellsten zur optimalen Lösung führt.

   Beispiele: Planung von Routen, Spielzügen, Chiplayouts, Roboterarmbahnen

Problemlösen

Problemlösungsstrategien

1. Generate and test / blind search

   Betrachte einen Lösungskandidaten nach dem anderen, bis eine adäquate Lösung gefunden ist und/oder alle Kandidaten betrachtet wurden.

   Beispiele: Verschlüsselung knacken; Suche in unsortierten Objekten

2. Hill climbing / local search

   Verbessere schrittweise einen initialen Lösungskandidaten bestmöglich, bis keine Verbesserung mehr möglich ist.

   Beispiele: Wechselgeld mit minimaler Zahl von Münzen herausgeben; Planung des Verlegens von Telefon-/Stromkabeln mit minimalen Kosten.

3. Informed search

   Verbessere schrittweise alternative Lösungskandiaten, und zwar zunächst den, der voraussichtlich am schnellsten zur optimalen Lösung führt.

   Beispiele: Planung von Routen, Spielzügen, Chiplayouts, Roboterarmbahnen

Problemlösen

Problemlösungsstrategien

1. Generate and test / blind search

   Betrachte einen Lösungskandidaten nach dem anderen, bis eine adäquate Lösung gefunden ist und/oder alle Kandidaten betrachtet wurden.

   Beispiele: Verschlüsselung knacken; Suche in unsortierten Objekten

2. Hill climbing / local search

   Verbessere schrittweise einen initialen Lösungskandidaten bestmöglich, bis keine Verbesserung mehr möglich ist.

   Beispiele: Wechselgeld mit minimaler Zahl von Münzen herausgeben; Planung des Verlegens von Telefon-/Stromkabeln mit minimalen Kosten.

3. Informed search

   Verbessere schrittweise alternative Lösungskandiaten, und zwar zunächst den, der voraussichtlich am schnellsten zur optimalen Lösung führt.

   Beispiele: Planung von Routen, Spielzügen, Chiplayouts, Roboterarmbahnen

Problemlösen

Problemlösungsstrategien

1. Generate and test / blind search

   Betrachte einen Lösungskandidaten nach dem anderen, bis eine adäquate Lösung gefunden ist und/oder alle Kandidaten betrachtet wurden.

   Beispiele: Verschlüsselung knacken; Suche in unsortierten Objekten

2. Hill climbing / local search

   Verbessere schrittweise einen initialen Lösungskandidaten bestmöglich, bis keine Verbesserung mehr möglich ist.

   Beispiele: Wechselgeld mit minimaler Zahl von Münzen herausgeben; Planung des Verlegens von Telefon-/Stromkabeln mit minimalen Kosten.

3. Informed search

   Verbessere schrittweise alternative Lösungskandiaten, und zwar zunächst den, der voraussichtlich am schnellsten zur optimalen Lösung führt.

   Beispiele: Planung von Routen, Spielzügen, Chiplayouts, Roboterarmbahnen

Problemlösen

Problemlösungsstrategien (Fortsetzung)

4. Divide and conquer

   Zerlege die Probleminstanz in Teilprobleme, die leicht zu lösen sind, und setze aus den Lösungen der Teilprobleme eine Gesamtlösung zusammen.

   Beispiele: Sortieren; Reiseplanung; Briefzustellung via Verteilzentren

5. Retrieve and adapt

   Wähle einen Lösungskandidaten unter den Lösungen bekannter Probleminstanzen und passe ihn der vorliegenden Instanz an.

   Beispiele: Präzedenzfälle für Urteile; Hotline-Systeme; Maschinenbau

6. Random select and test

   Wähle Lösungskandidaten zufällig und prüfe, ob eine adäquate Lösung erzielt wurde.

   Beispiele: Evolution; Stunden-/Ablaufplanung mit Bedingungen; Kalibrierung

Problemlösen

Problemlösungsstrategien (Fortsetzung)

4. Divide and conquer

   Zerlege die Probleminstanz in Teilprobleme, die leicht zu lösen sind, und setze aus den Lösungen der Teilprobleme eine Gesamtlösung zusammen.

   Beispiele: Sortieren; Reiseplanung; Briefzustellung via Verteilzentren

5. Retrieve and adapt

   Wähle einen Lösungskandidaten unter den Lösungen bekannter Probleminstanzen und passe ihn der vorliegenden Instanz an.

   Beispiele: Präzedenzfälle für Urteile; Hotline-Systeme; Maschinenbau

6. Random select and test

   Wähle Lösungskandidaten zufällig und prüfe, ob eine adäquate Lösung erzielt wurde.

   Beispiele: Evolution; Stunden-/Ablaufplanung mit Bedingungen; Kalibrierung

Problemlösen

Problemlösungsstrategien (Fortsetzung)

4. Divide and conquer

   Zerlege die Probleminstanz in Teilprobleme, die leicht zu lösen sind, und setze aus den Lösungen der Teilprobleme eine Gesamtlösung zusammen.

   Beispiele: Sortieren; Reiseplanung; Briefzustellung via Verteilzentren

5. Retrieve and adapt

   Wähle einen Lösungskandidaten unter den Lösungen bekannter Probleminstanzen und passe ihn der vorliegenden Instanz an.

   Beispiele: Präzedenzfälle für Urteile; Hotline-Systeme; Maschinenbau

6. Random select and test

   Wähle Lösungskandidaten zufällig und prüfe, ob eine adäquate Lösung erzielt wurde.

   Beispiele: Evolution; Stunden-/Ablaufplanung mit Bedingungen; Kalibrierung

Problemlösen

Problemlösungsstrategien (Fortsetzung)

4. Divide and conquer

   Zerlege die Probleminstanz in Teilprobleme, die leicht zu lösen sind, und setze aus den Lösungen der Teilprobleme eine Gesamtlösung zusammen.

   Beispiele: Sortieren; Reiseplanung; Briefzustellung via Verteilzentren

5. Retrieve and adapt

   Wähle einen Lösungskandidaten unter den Lösungen bekannter Probleminstanzen und passe ihn der vorliegenden Instanz an.

   Beispiele: Präzedenzfälle für Urteile; Hotline-Systeme; Maschinenbau

6. Random select and test

   Wähle Lösungskandidaten zufällig und prüfe, ob eine adäquate Lösung erzielt wurde.

   Beispiele: Evolution; Stunden-/Ablaufplanung mit Bedingungen; Kalibrierung

Bemerkungen:

- Ein Lösungskandidat kann entweder eine von vornherein vollständige Lösung sein, die es zu optimieren gilt, oder aber eine Teillösung die es (optimal) zu vervollständigen gilt.
- Jede der Problemlösungsstrategien ist nur unter Ausnutzung von Struktureigenschaften eines Problems (der Problemstruktur) einsetzbar:
  - Generate and test / blind search: Der Lösungsraum kann systematisch aufgezählt werden.
  - Hill climbing / local search: Es gibt ein Maß dafür, welche Veränderung einen Lösungskandidaten (wie stark) verbessert oder verschlechtert.
  - Informed search: Der Abstand eines Lösungskandidaten zur gewünschten Lösung kann abgeschätzt werden.
  - Divide and conquer: Das Problem lässt sich in Teilprobleme zerlegen, wobei die Zusammensetzung optimaler Teillösungen zu einer optimalen Gesamtlösung führt.
  - Retrieve and adapt: Es existiert ein Weg, die Lösungen bekannter Probleme in Lösungen neuer Probleme zu verwandeln.
  - Random select and text: Der Raum möglicher Lösungen ist groß und seine Struktur unbekannt; es gibt jedoch hinreichend viele adäquate Lösungskandidaten, so dass eine Zufallsauswahl oft funktioniert.

Problemlösen

Problemlösungsstrategien (Fortsetzung)

Algorithmen sind ein Mittel dafür, Problemlösungsstrategien in einer für die Ausführung durch einen Computer geeigneten Weise zu formalisieren.

Voraussetzung dafür ist eine genügendes Verständnis des jeweiligen Problems sowie der für seine Lösung erforderlichen Schritte.

Problemlösen

Problemlösungsstrategien (Fortsetzung)

Algorithmen sind ein Mittel dafür, Problemlösungsstrategien in einer für die Ausführung durch einen Computer geeigneten Weise zu formalisieren.

Voraussetzung dafür ist eine genügendes Verständnis des jeweiligen Problems sowie der für seine Lösung erforderlichen Schritte.

Beispiele für Probleme, die sich einer algorithmischen Lösung entziehen:

- Identifikation von Buchstaben in Bildern
- Vorhersage von Kaufentscheidungen
- Risikoanalyse bei Kreditvergaben
- Auto fahren

Problemlösen

Problemlösungsstrategien (Fortsetzung)

Algorithmen sind ein Mittel dafür, Problemlösungsstrategien in einer für die Ausführung durch einen Computer geeigneten Weise zu formalisieren.

Voraussetzung dafür ist eine genügendes Verständnis des jeweiligen Problems sowie der für seine Lösung erforderlichen Schritte.

Beispiele für Probleme, die sich einer algorithmischen Lösung entziehen:

- Identifikation von Buchstaben in Bildern
- Vorhersage von Kaufentscheidungen
- Risikoanalyse bei Kreditvergaben
- Auto fahren

Der Mensch lernt, solche Probleme zunächst anhand von Trainingsbeispielen und daraufhin aus gesammelten Erfahrungen zu lösen.

Sind Computer zu dieser Art der Problemlösung in der Lage?

Problemlösen

Maschinelles Lernen

Basierend auf einer Menge von Beispielen von Probleminstanzen und ihren jeweiligen Lösungen, erlerne ein Modell, um Lösungskandidaten für zuvor ungesehene Instanzen zu bestimmen.

Problemlösen

Maschinelles Lernen

Basierend auf einer Menge von Beispielen von Probleminstanzen und ihren jeweiligen Lösungen, erlerne ein Modell, um Lösungskandidaten für zuvor ungesehene Instanzen zu bestimmen.

Der Unterschied zwischen Problemlösungsstrategien und maschinellem Lernen ist der zugrundeliegende Inferenzmechanismus:

- Deduktives Schließen:

$$\textit{Instanz} \Rightarrow \textit{Loesung}$$

„Schluss vom Allgemeinen [Gesetz formuliert als Algorithmus] auf das Besondere [Lösung].“

- Induktives Schließen:

$$\textit{Instanz} \rightarrow \textit{Loesung}$$

„Schluss vom Besonderen [Beispiele] auf das Allgemeine [Gesetz kodiert als Modell].“

Bemerkungen:

- Eine formal definiertes Problem und ein Algorithmus, der es löst, bilden ein Kalkül mit dem Lösungen aus Probleminstanzen deduktiv ableitbar werden. Unter der Voraussetzung der Korrektheit des Algorithmus entspricht die Ableitung einer Lösung für eine gegebene Instanz dem Beweis ihrer Richtigkeit. Beispiel:

  Alle Menschen sind sterblich.
  Sokrates ist ein Mensch.

  $\Rightarrow$ Sokrates ist sterblich.

- Abhängig von ihrer „Repräsentativität“ für das betrachtete Problem erlaubt die Analyse einer Menge von Beispielen von Instanzen und Lösungen ein auf Analogiebildung beruhendes Modell zur Lösung aufzustellen, das sich an den Charakteristiken der Instanzen orientiert. Beispiele:

  Sokrates ist sterblich.
  Sokrates ist ein Mensch.

  $\rightarrow$ Alle Menschen sind sterblich.

  Bodo ist ein Dackel.
  Bodo ist ein Hund.

  $\rightarrow$ Alle Hunde sind Dackel.