



Vortrag

---

# Suchverfahren der Künstlichen Intelligenz

Sven Schmidt (Technische Informatik)



# Suchverfahren der Künstlichen Intelligenz

---

- **Grundlagen**
  - **Zustandsraumrepräsentation**
  - Generische Suche
  - Bewertung von Suchstrategien
- Uninformierte Suchverfahren
  - Breitensuche
  - Gleiche-Kosten-Suche
  - Tiefensuche
  - Schrittweise vertiefende Suche
- Heuristische Suchverfahren
  - Heuristische Schätzfunktionen
  - Bergsteigen



# Suche in der Künstlichen Intelligenz

---

- Probleme der Künstlichen Intelligenz werden in Probleme der Suche überführt



# Zustandsraumrepräsentation

---

- Zustände
- Zustandsübergangsooperationen



# Zustände

---

- Informationsstand des Suchschrittes
- Startzustand : Information zum Anfang der Suche (Ausgangssituation)
- Endzustand : Information die zum Beenden der Suche führt (Lösung des Problems)
- Verursacht Kosten : Speicher



# Möglicher Startzustand

---

<b>1</b>	<b>2</b>	<b>3</b>
	<b>5</b>	<b>4</b>
<b>8</b>	<b>7</b>	<b>6</b>



# Zielzustand

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>4</b>	<b>5</b>	<b>6</b>
<b>7</b>	<b>8</b>	



# Zustandsübergangsoperatoren

---

- Übergang von einem Zustand zu seinem Folgezustand
- Meist durch Berechnung
- Verursacht Kosten : Zeit und Speicher



# Beispiel : Schiebepuzzle

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>7</b>		<b>4</b>
<b>5</b>	<b>8</b>	<b>6</b>



# Beispiel : Schiebepuzzle

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>7</b>		<b>4</b>
<b>5</b>	<b>8</b>	<b>6</b>



# Zustandsraumrepräsentation

---

- Üblich : Darstellung als Baum
- $d$  = Tiefe des Baumes
- $b$  = Verzweigungsgrad des Baumes (Anzahl der Nachfolgeknoten)

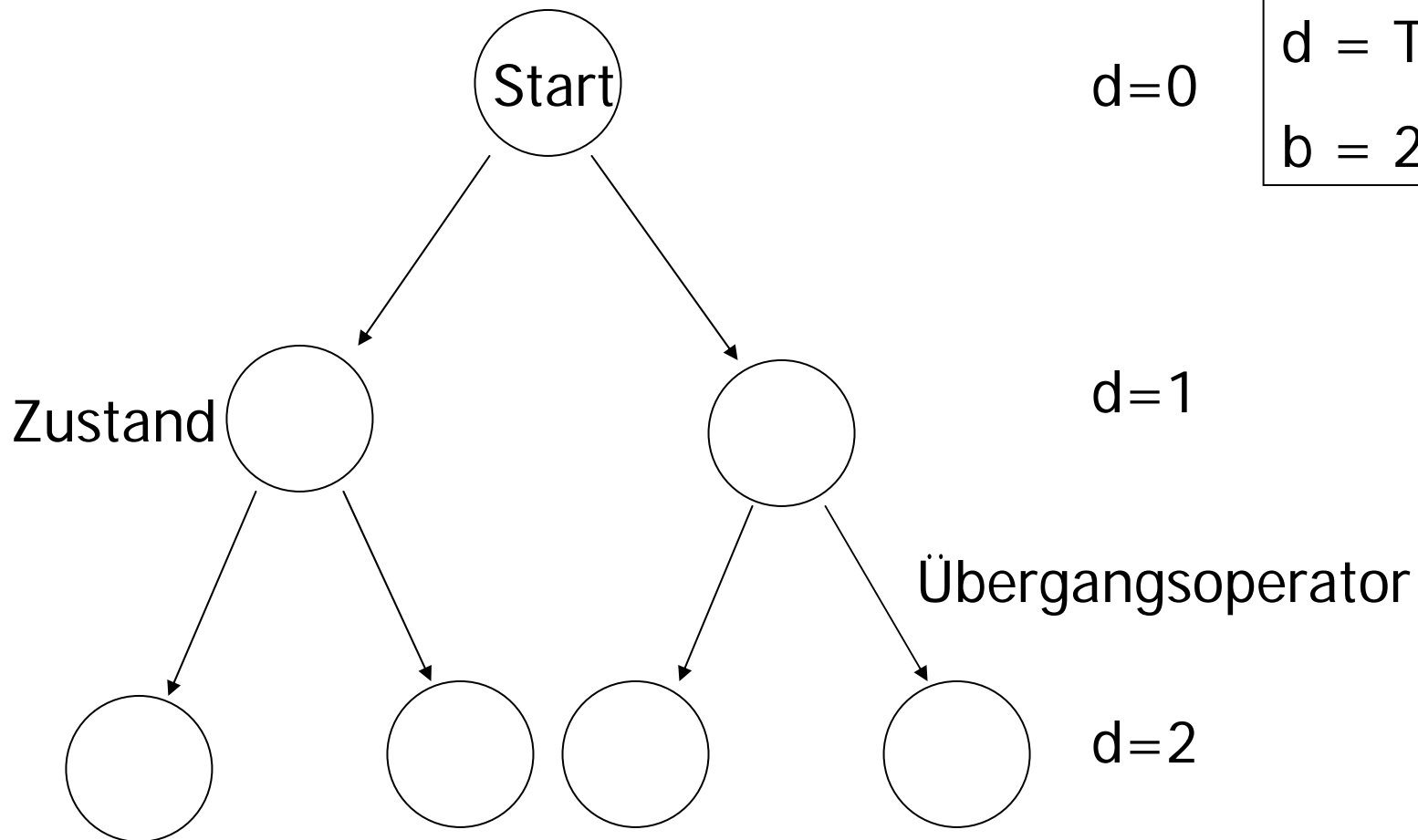


# Zustandsraumrepräsentation

---

- Üblich : Darstellung als Baum
- $d$  = Tiefe des Baumes
- $b$  = Verzweigungsgrad des Baumes (Anzahl der Nachfolgeknoten)
- Ziel: Durch Suchverfahren mit möglichst wenig Ressourcenaufwand möglichst kurze Lösungspfade finden

# Zustandsraumrepräsentation





# Expansion

---

- Berechnung des nächsten Zustandes durch die Übergangsfunktion
- In einer Liste : Vaterknoten vernichten und durch seine Nachfolgeknoten ersetzen
- Zeit sowie Speicherplatz werden verbraucht



# Suchverfahren der Künstlichen Intelligenz

---

- **Grundlagen**
  - Zustandsraumrepräsentation
  - **Generische Suche**
  - Bewertung von Suchstrategien
- Uninformierte Suchverfahren
  - Breitensuche
  - Gleiche-Kosten-Suche
  - Tiefensuche
  - Schrittweise vertiefende Suche
- Heuristische Suchverfahren
  - Heuristische Schätzfunktionen
  - Bergsteigen



# Erkennung Start-/ Zielzustand

---

- Durch algorithmisch überprüfbare Eigenschaften charakterisiert
- **Explizit** vorgegebene Zustandsmenge : Aufzählung der jeweiligen Elemente die den Start-/ Zielknoten identifizieren
- **Implizit** vorgegebene Zustandsmenge : Aufzählung nicht möglich aber errechenbar (zusätzliche Zeitverzögerung in der Suche)



# Generische Suche

---

- Allgemeines Verfahren der Suche
- Durch Spezialisierung kann jedes beliebige Suchverfahren gewonnen werden

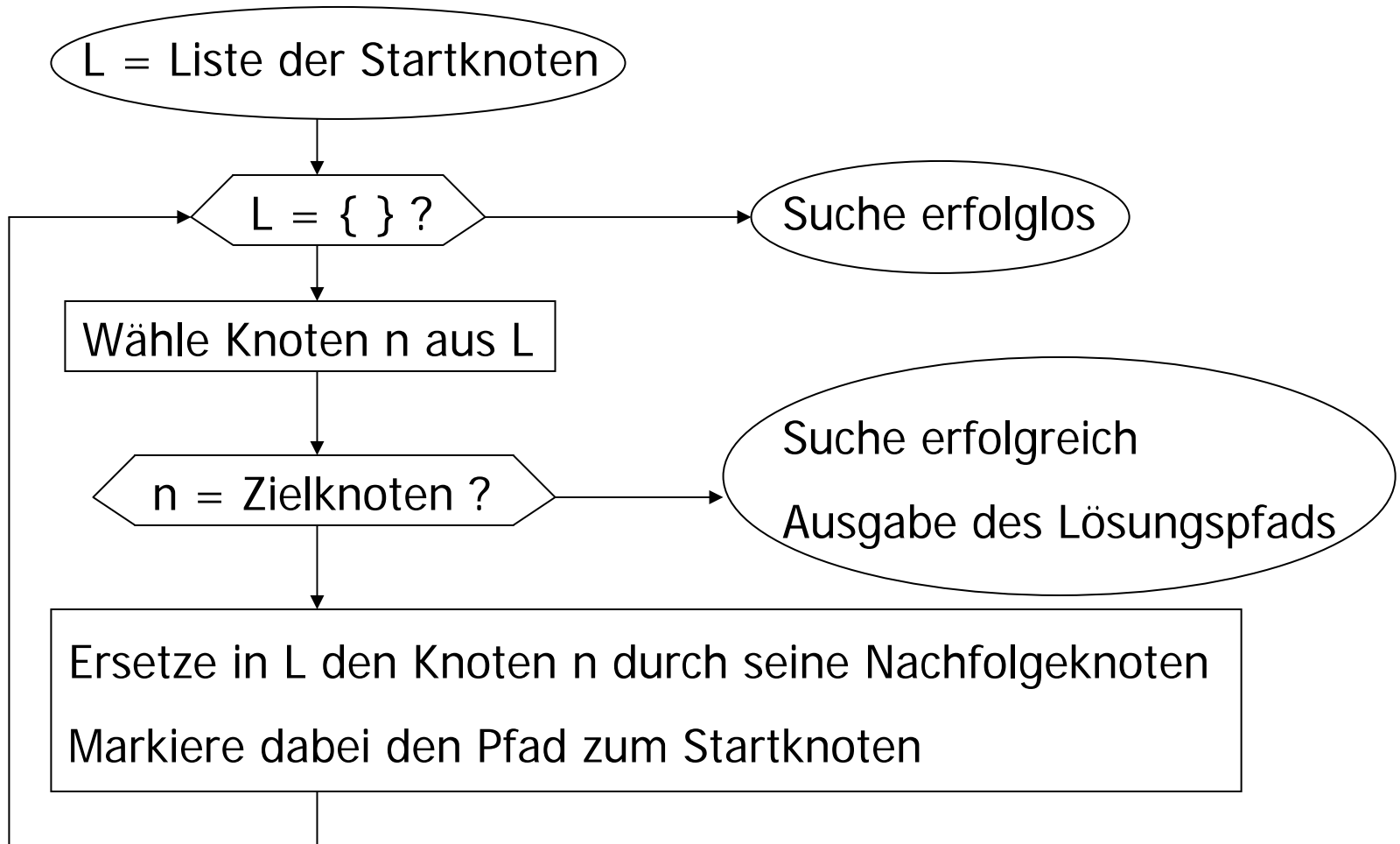


# Generische Suche

---

- L = Liste der Zustände die noch nicht auf Zieleigenschaften untersucht wurden (AGENDA)
- Wenn L = lineare Liste, dann wird Knoten weiter vorne als erstes überprüft
- Einfügen und Auswählen der Nachfolgezustände (Knoten) in L bestimmt Suchstrategie

# Generische Suche





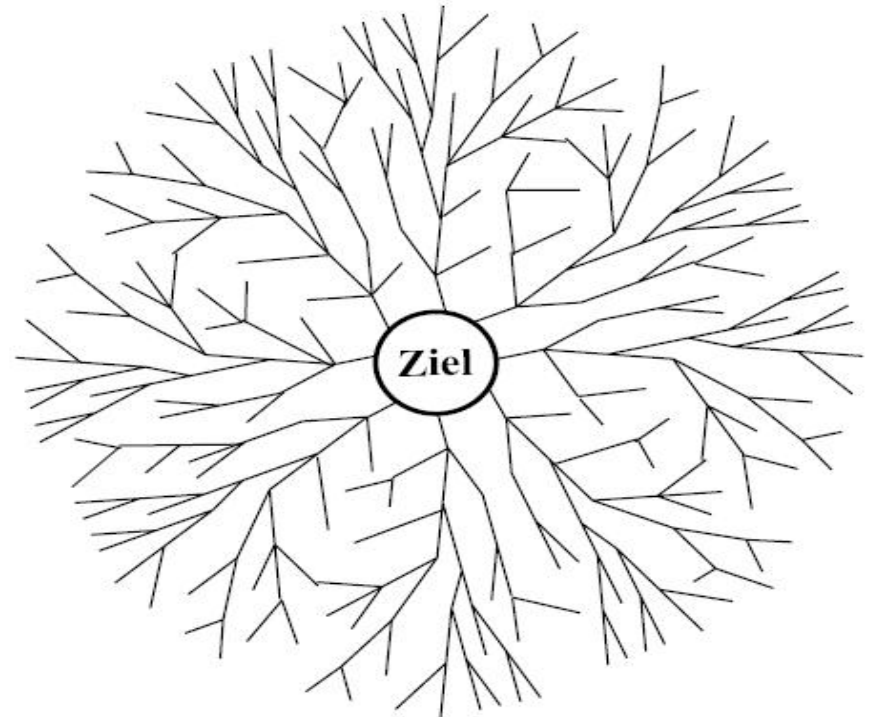
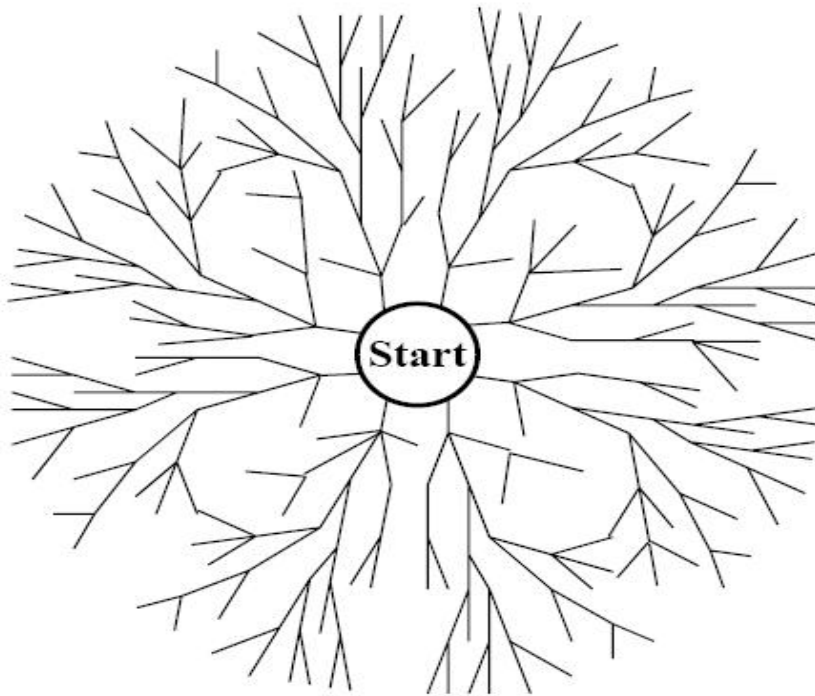
# Vorwärtssuche – Rückwärtssuche

---

- Rückwärtssuche
  - Vom Zielknoten aus werden alle möglichen Vorgängerknoten generiert, bis Startknoten gefunden wird
- Bedingung für Rückwärtssuche:
  - Zustandsoperatoren sind leicht umkehrbar
  - Start-/Zielzustände sind mit wenig Aufwand aufzählbar



# Vorwärts- und Rückwärtssuche





# Probleme bei zyklischen Pfaden

---

- Verschiedene Knoten können gleichen Zustand repräsentieren
- Gefahr das Suche nicht terminiert!
- Abhilfe:
  - „Closed-List“ C auf der alle überprüften Knoten gelangen
  - Erhöhter Speicher- und Zeitaufwand!



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>8</b>	<b>5</b>	<b>4</b>
<b>7</b>		<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>8</b>	<b>5</b>	<b>4</b>
	<b>7</b>	<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
	<b>5</b>	<b>4</b>
<b>8</b>	<b>7</b>	<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>5</b>		<b>4</b>
<b>8</b>	<b>7</b>	<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>5</b>	<b>7</b>	<b>4</b>
<b>8</b>		<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>5</b>	<b>7</b>	<b>4</b>
	<b>8</b>	<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
	<b>7</b>	<b>4</b>
<b>5</b>	<b>8</b>	<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>7</b>		<b>4</b>
<b>5</b>	<b>8</b>	<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>7</b>	<b>8</b>	<b>4</b>
<b>5</b>		<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>7</b>	<b>8</b>	<b>4</b>
	<b>5</b>	<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
	<b>8</b>	<b>4</b>
<b>7</b>	<b>5</b>	<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>8</b>		<b>4</b>
<b>7</b>	<b>5</b>	<b>6</b>



# Beispiel : zyklische Pfade

---

<b>1</b>	<b>2</b>	<b>3</b>
<b>8</b>	<b>5</b>	<b>4</b>
<b>7</b>		<b>6</b>

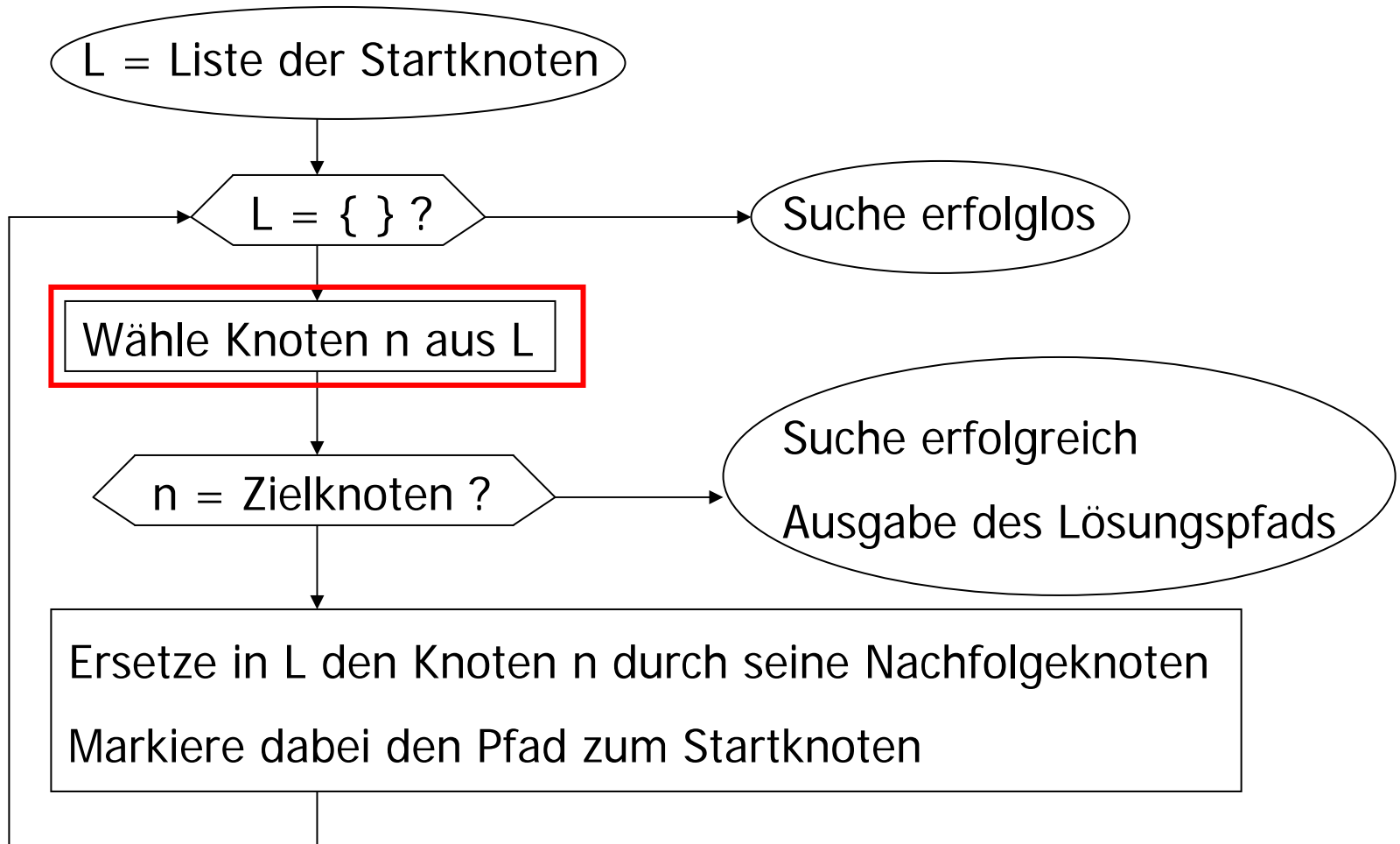


# Suchverfahren der Künstlichen Intelligenz

---

- **Grundlagen**
  - Zustandsraumrepräsentation
  - Generische Suche
  - **Bewertung von Suchstrategien**
- Uninformierte Suchverfahren
  - Breitensuche
  - Gleiche-Kosten-Suche
  - Tiefensuche
  - Schrittweise vertiefende Suche
- Heuristische Suchverfahren
  - Heuristische Schätzfunktionen
  - Bergsteigen

# Bewertung von Suchstrategien





# Bewertung von Suchstrategien

---

- Bestimmt welcher Teil und in welcher Reihenfolge der Suchbaum durchsucht wird
- Vollständiges Suchverfahren :
  - ALLE Knoten des Baums werden expandiert bis zum Ergebnis (Suche Erfolgreich / Erfolglos)
- Unfaire Suchverfahren :
  - Knoten werden zuerst in der Tiefe gesucht
  - Kann zu Problemen führen wenn Suchpfade unendlich lang



# Bewertung von Suchstrategien

---

- $\text{Space}(X)$  :
  - Alle Gleichzeitig in der Agenda  $L$  gespeicherten Knoten
- $\text{Time}(X)$  :
  - Zeit die beim untersuchen der Zustände auf die Zieleigenschaft aufgewendet wird



# Bewertung von Suchstrategien

---

- Bei nachfolgenden Betrachtungen gelten folgende Einschränkungen um Suchverfahren zu bewerten :
  - Suchbaum ist Uniform : konstanter Verzweigungsgrad, einheitliche Tiefe  $d$
  - Zielknoten liegt mit gleicher Wahrscheinlichkeit in der Maximaltiefe  $d$  (ganz unten im Suchbaum)

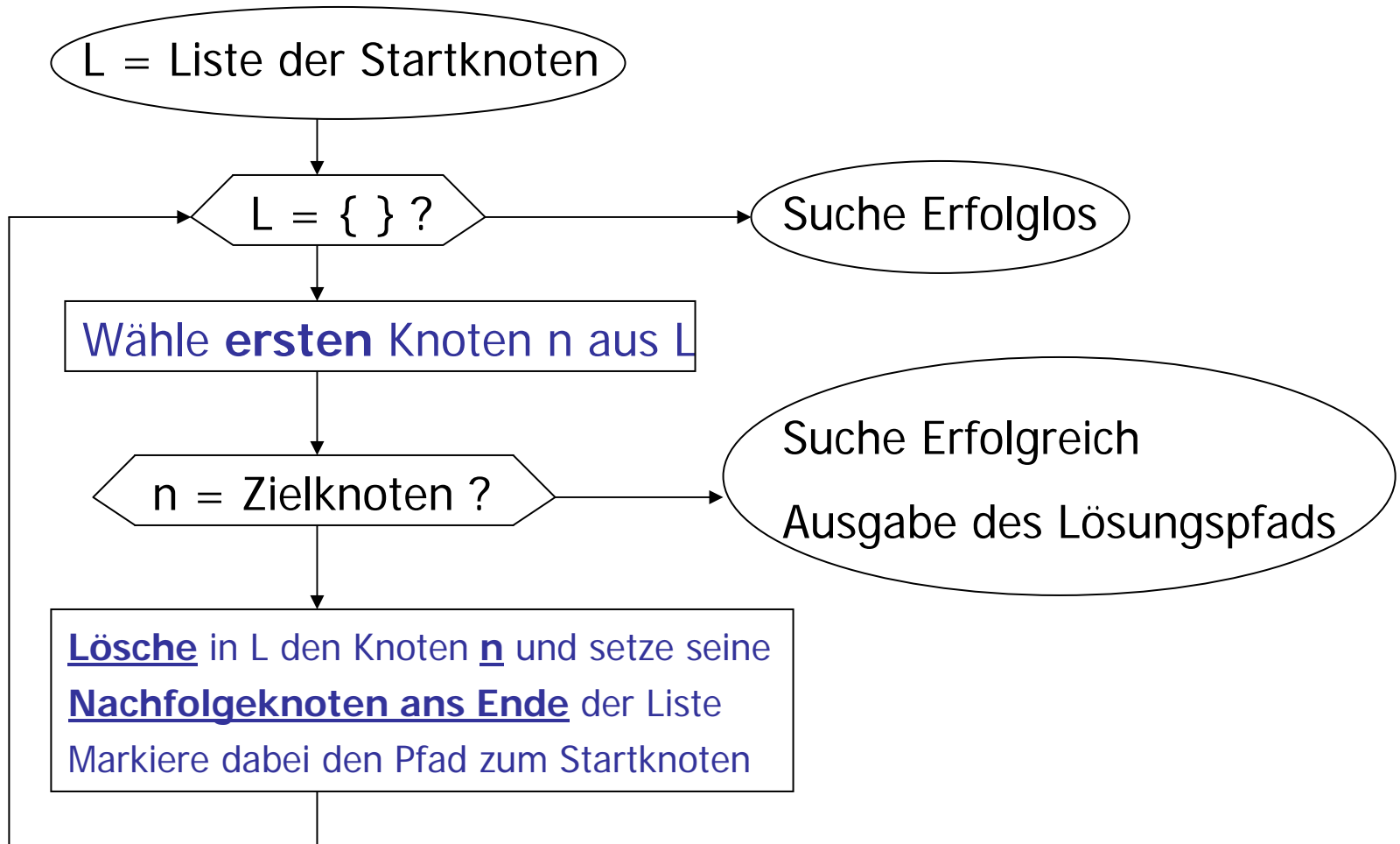


# Suchverfahren der Künstliche Intelligenz

---

- Grundlagen
  - Zustandsraumrepräsentation
  - Generische Suche
  - Bewertung von Suchstrategien
- **Uninformierte Suchverfahren**
  - **Breitensuche**
  - Gleiche-Kosten-Suche
  - Tiefensuche
  - Schrittweise vertiefende Suche
- Heuristische Suchverfahren
  - Heuristische Schätzfunktionen
  - Bergsteigen

# Breitensuche



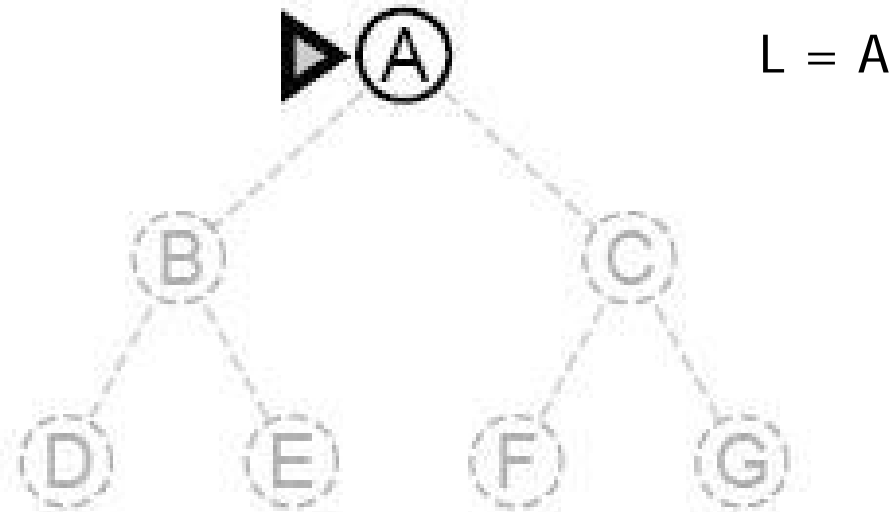


# Breitensuche

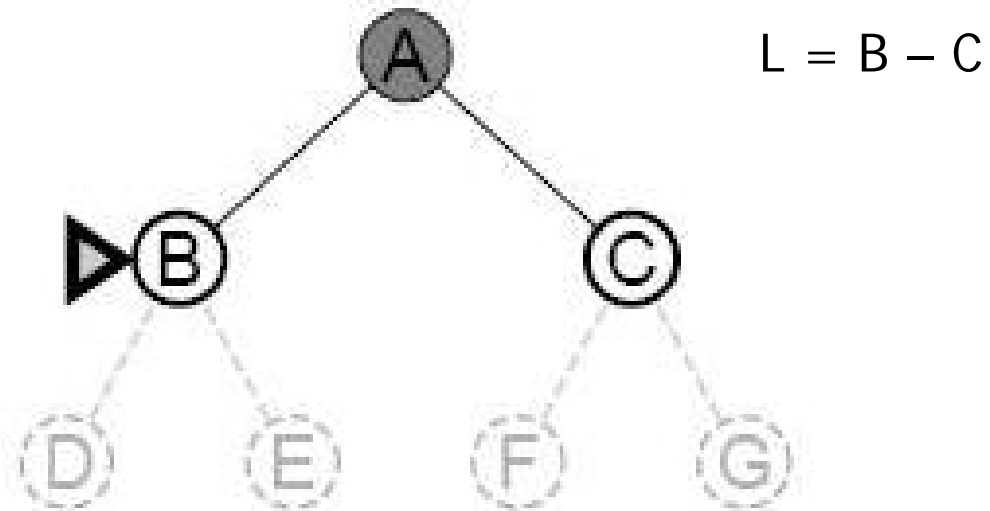
---

- Nachfolgeknoten werden an das Ende der Agenda L gesetzt, zu überprüfender Knoten n wird vom Anfang entnommen
- Suchbaum wird Schicht für Schicht durchsucht
- Knoten der nächsten Ebene werden erst durchsucht, wenn Knoten der vorangegangenen Ebene überprüft wurden

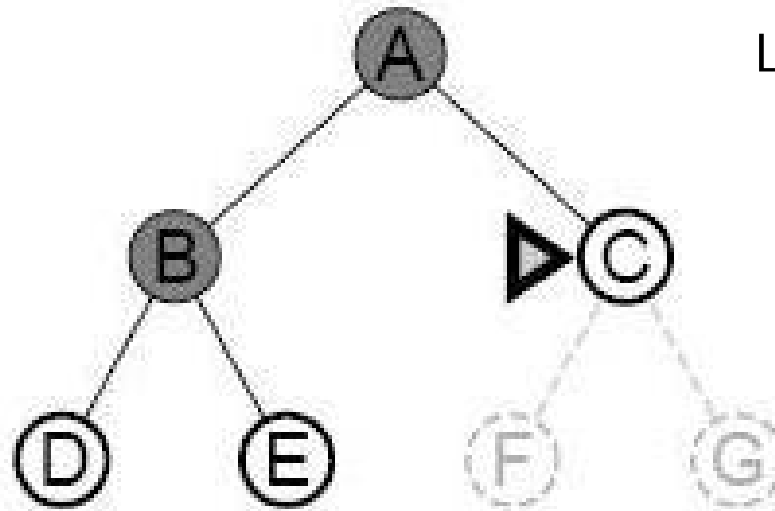
# Breitensuche - Suchbaum



# Breitensuche - Suchbaum

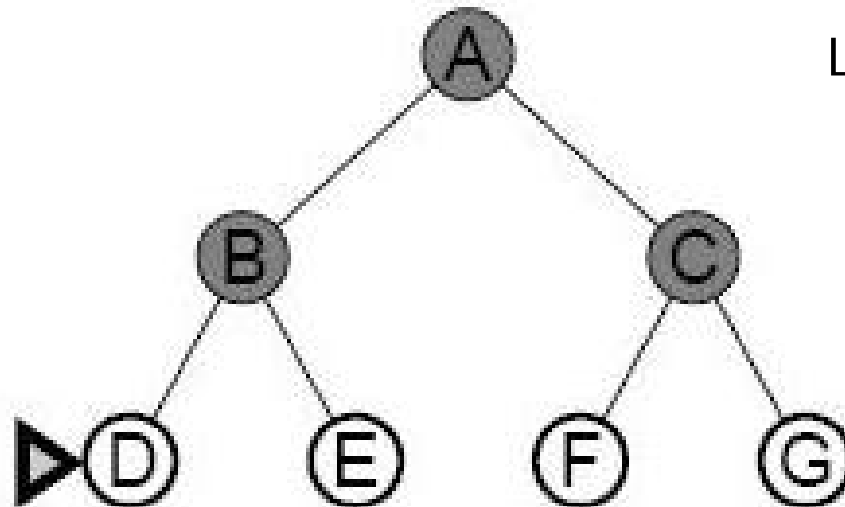


# Breitensuche - Suchbaum



L = C - D - E

# Breitensuche - Suchbaum



L = D - E - F - G



# Breitensuche - Speicheraufwand

---

- Bevor erster Knoten aus einer Ebene überprüft wird, stehen alle Knoten aus derselben Ebene in Agenda L
- $\text{Space}(B) = b^d$  Exponentiell !!
- Die Knoten der Ebene davor wurden alle durch ihre Nachfolgeknoten ersetzt!



# Breitensuche - Zeitaufwand

---

- Im besten Fall ist der erste Knoten der nächsten Ebene der Zielknoten (best case)
- Im schlimmsten Fall der Letzte (worst case)
- Es müssen Alle Knoten der oberen Ebene ( $d-1$ ) überprüft worden sein



# Breitensuche - Zeitaufwand

---

$$\sum_0^{(d-1)} b^k = \frac{(b^d - 1)}{(b - 1)}$$

- Alle Knoten der vorangegangenen Ebene

- Best case : Knoten der vorherigen Ebene + Zielknoten

$$\frac{(b^d - 1)}{(b - 1)} + 1 = \frac{(b^d - 1 + (b - 1))}{(b - 1)} = \frac{(b^d + b - 2)}{(b - 1)} = O(b^{(d-1)})$$



# Breitensuche - Zeitaufwand

- Worst case : Knoten der vorherigen Ebene + Alle Knoten der aktuellen Ebene

$$\frac{(b^d - 1)}{(b - 1)} + b^d = \frac{(b^d - 1)}{(b - 1)} + \frac{b^d * (b - 1)}{(b - 1)} = \frac{(b^{(d+1)} - 1)}{(b - 1)} = O(b^d)$$

- Mittlerer Zeitaufwand : (Bestcase + Worstcase) / 2 (Exponentiell !!)

$$TIME(Breitensuche) = \frac{(b^{(d+1)} + b^d + b - 3)}{(2 * (b - 1))} = O(b^d)$$



# Breitensuche : Beispiel

Tiefe	Knoten	Zeit	Speicher
2	1100	1 sek	1 MB
4	111100	11 sek	100 MB
6	$10^7$	19 min	10 GB
8	$10^9$	31 std	1 TeraB
10	$10^{11}$	129 T.	100 TeraB
12	$10^{13}$	35 J.	100 PetaB
14	$10^{15}$	3523 J.	1 ExaB

$b = 10,$

10.000 Knoten/sek.,

1000 Byte/Knoten



# Sucheverfahren der Künstlichen Intelligenz

---

- Grundlagen
  - Zustandsraumrepräsentation
  - Generische Suche
  - Bewertung von Suchstrategien
- **Uninformierte Suchverfahren**
  - Breitensuche
  - **Gleiche-Kosten-Suche**
  - Tiefensuche
  - Schrittweise vertiefende Suche
- Heuristische Suchverfahren
  - Heuristische Schätzfunktionen
  - Bergsteigen



# Gleiche-Kosten-Suche

---

- Kosten der Übergangsfunktionen von einem Zustand in den nächsten müssen bekannt sein
- Expandiert wird der Knoten, der die kleinsten Operator-Kosten aufweist
- Kostenfunktion durch Aufsummieren der Kosten :

$$g(n_k) = \sum_0^{k-1} c(n_i \rightarrow n_{(i+1)})$$

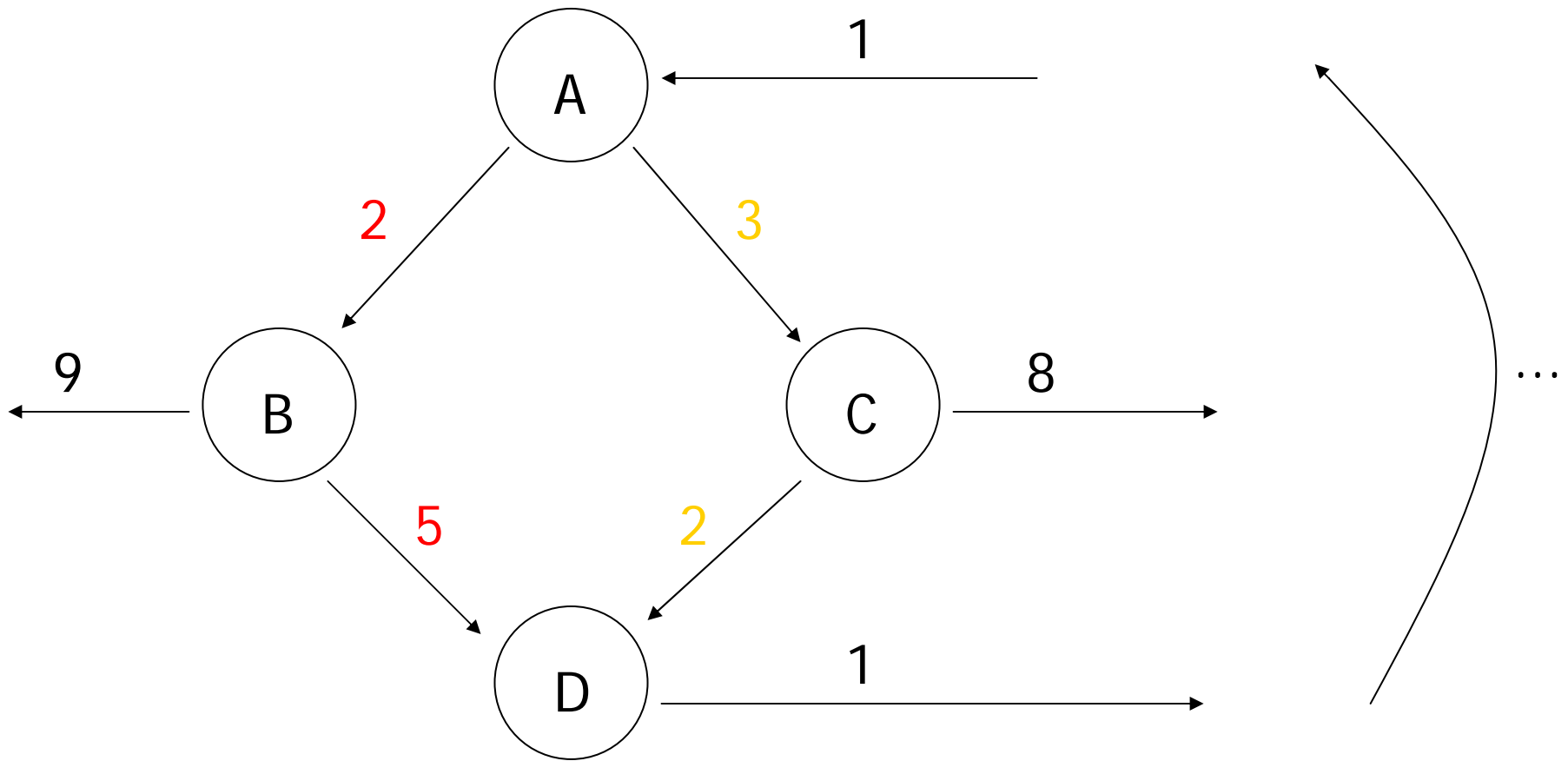


# Gleiche-Kosten-Suche

---

- Bei Suchbäumen mit Zyklen :
  - Vor Einfügen eines Knotens in Agenda L wird geprüft, ob sich dieser Knoten bereits in einem Pfad mit höheren Kosten befindet
  - Ist dies der Fall, so wird der Pfad entfernt und durch den Knoten ersetzt
  - Pfad wird in Knoten selbst gespeichert !
  - Somit ist der kostenoptimale Pfad in L gespeichert

# Gleiche-Kosten-Suche





# Suchverfahren der Künstlichen Intelligenz

---

- Grundlagen
  - Zustandsraumrepräsentation
  - Generische Suche
  - Bewertung von Suchstrategien
- **Uninformierte Suchverfahren**
  - Breitensuche
  - Gleiche-Kosten-Suche
  - **Tiefensuche**
  - Schrittweise vertiefende Suche
- Heuristische Suchverfahren
  - Heuristische Schätzfunktionen
  - Bergsteigen

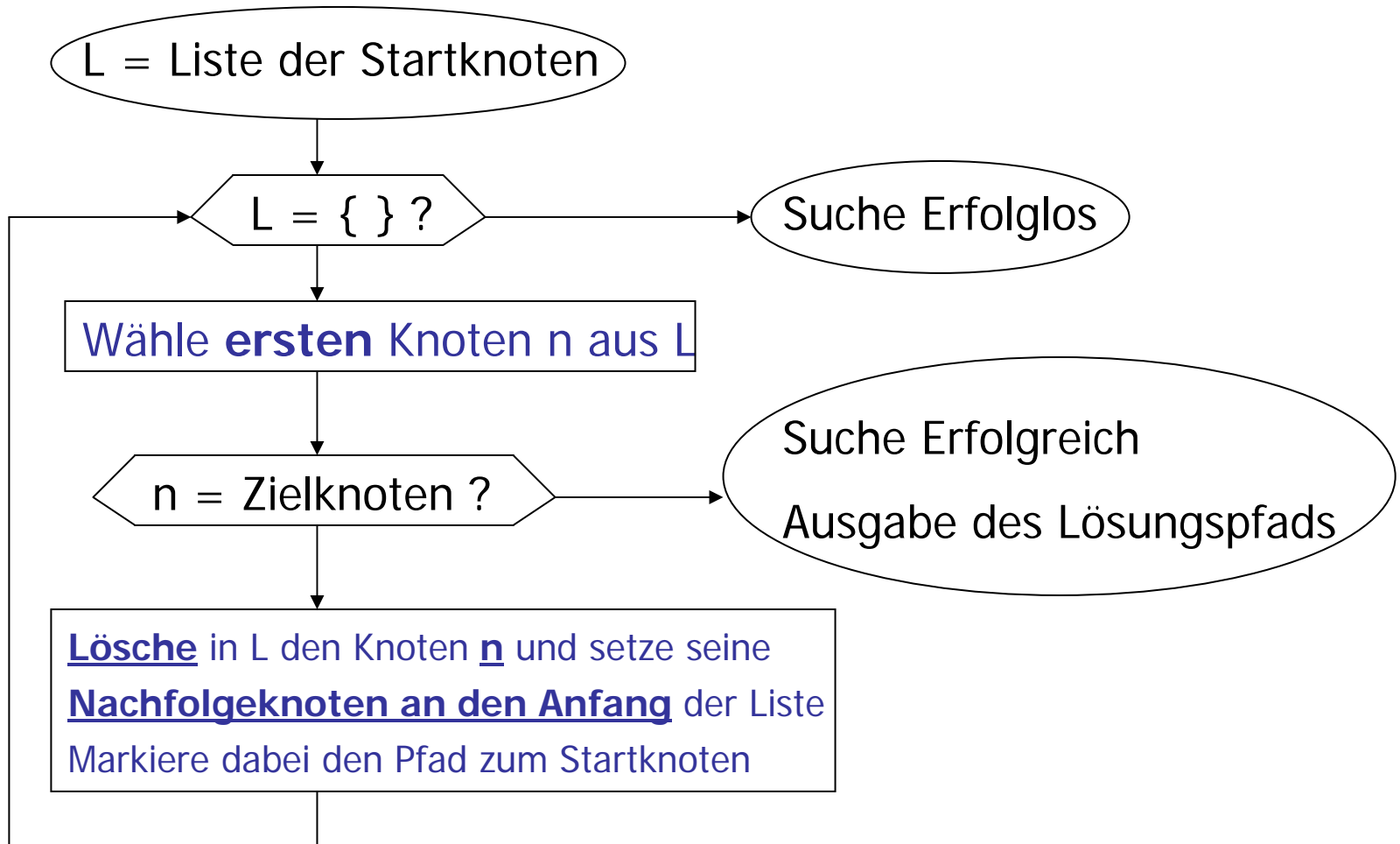


# Tiefensuche

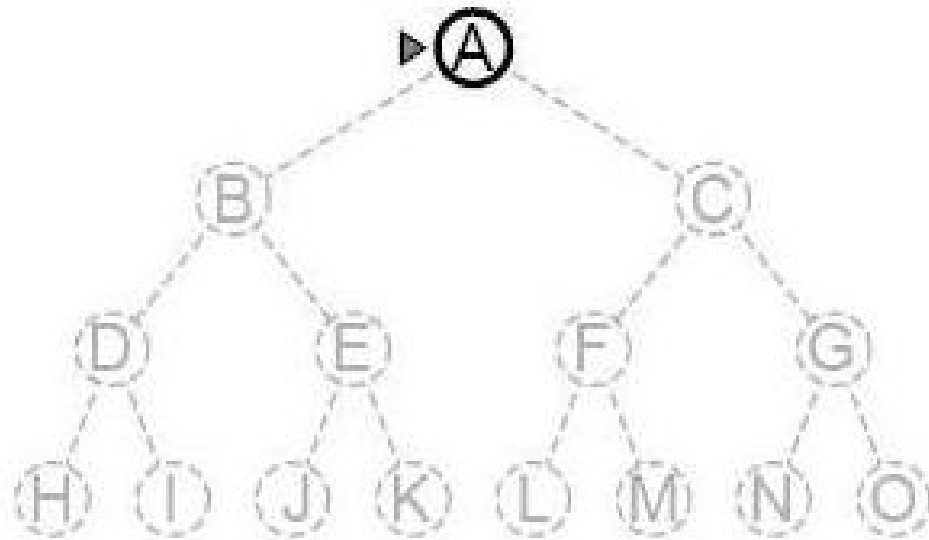
---

- Nachfolgeknoten werden am Anfang der Liste eingefügt
- Bevor Geschwister expandiert werden, müssen alle Kinder überprüft worden sein

# Tiefensuche



# Tiefensuche

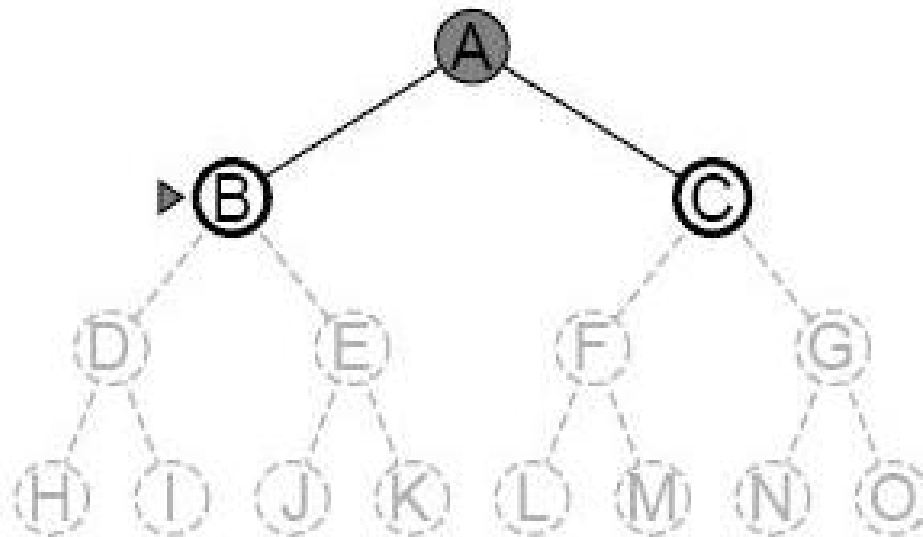


L = A



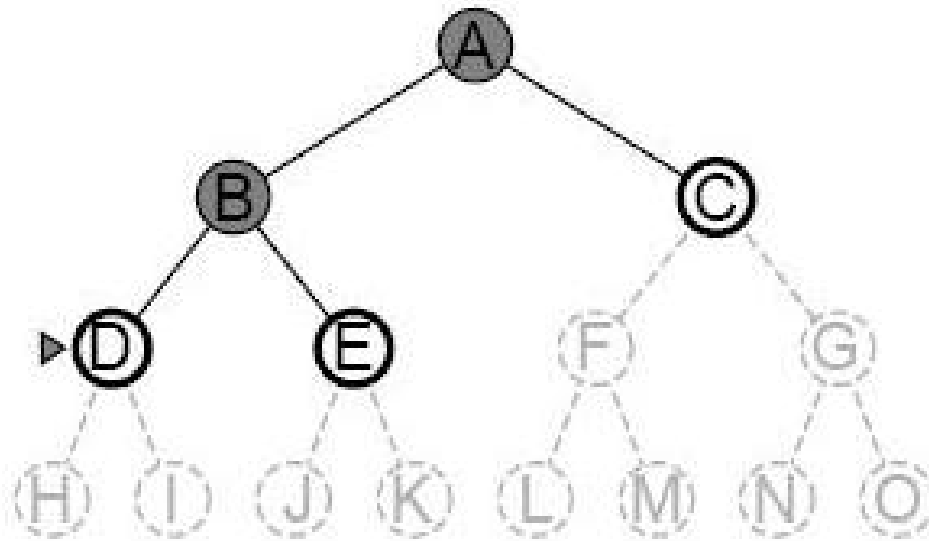
# Tiefensuche

---



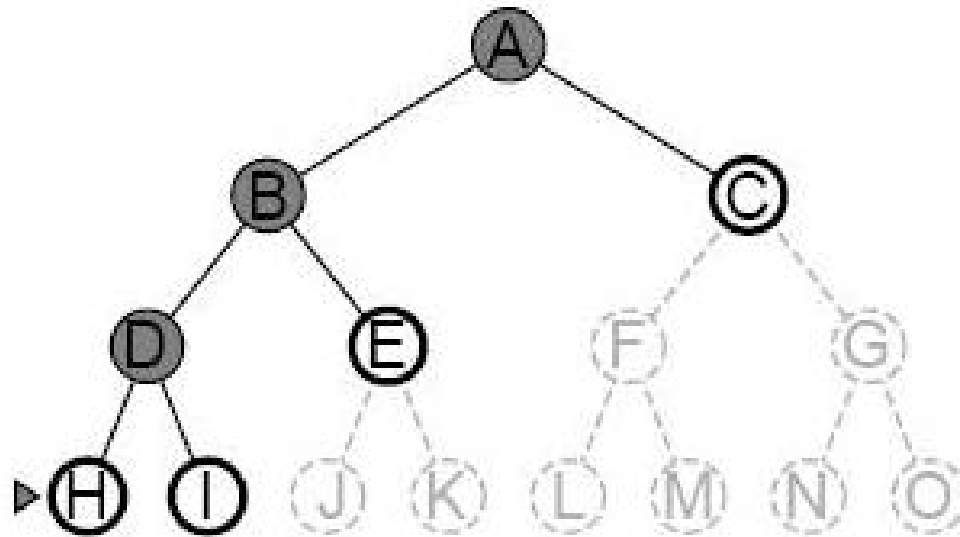
$$L = B - C$$

# Tiefensuche



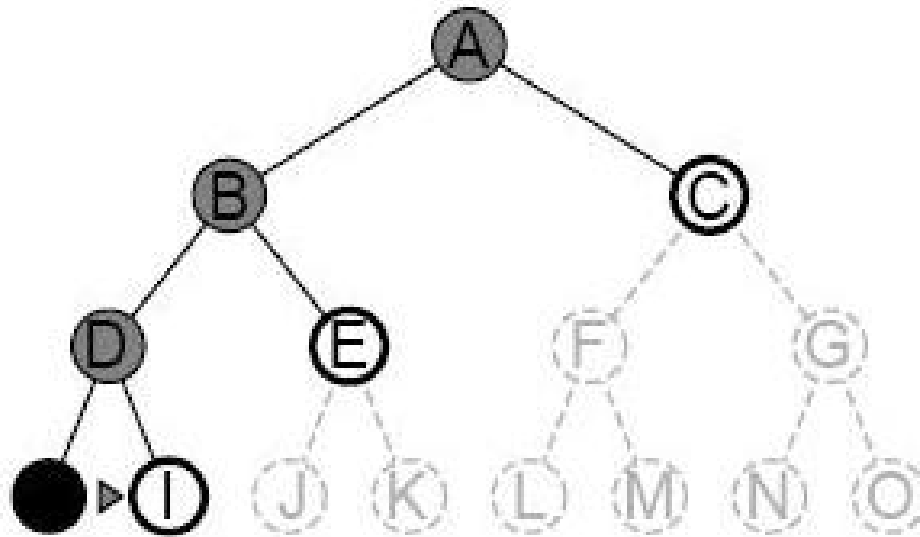
L = D - E - C

# Tiefensuche



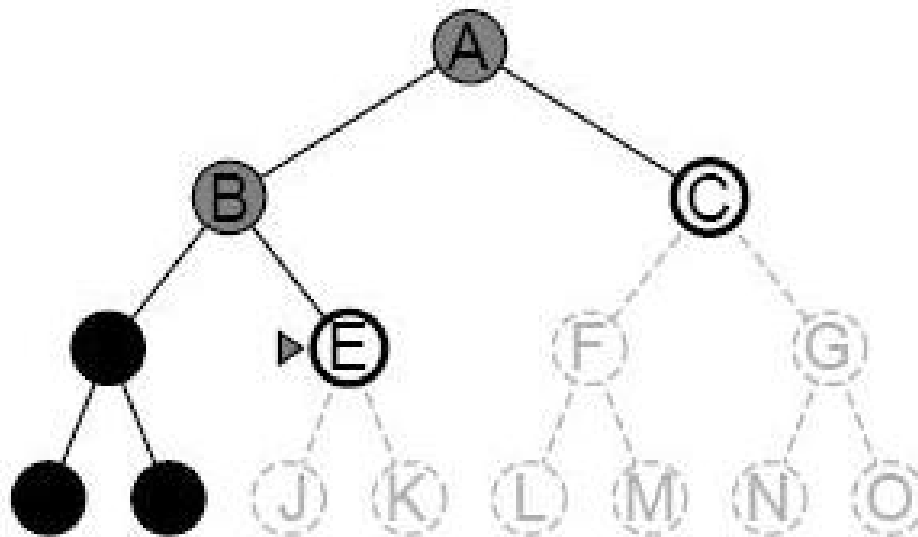
L = H - I - E - C

# Tiefensuche



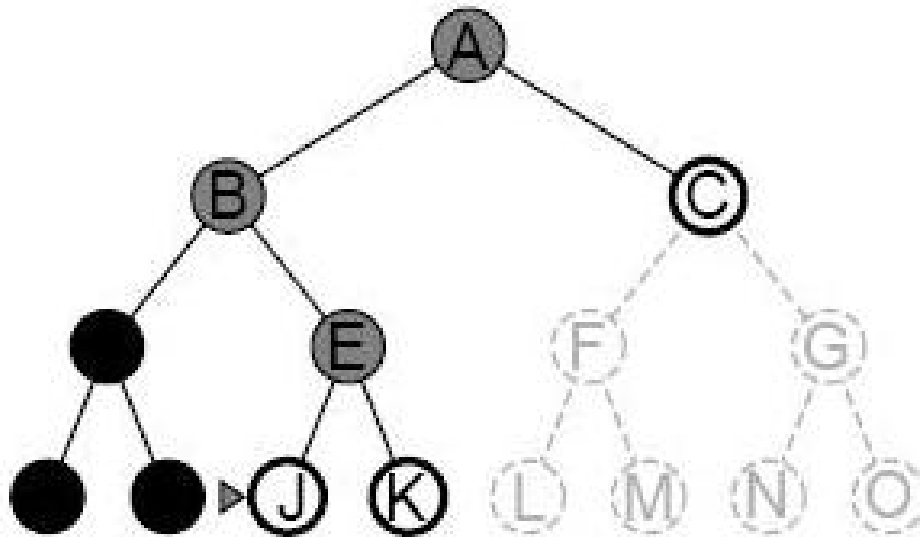
L = I - E - C

# Tiefensuche



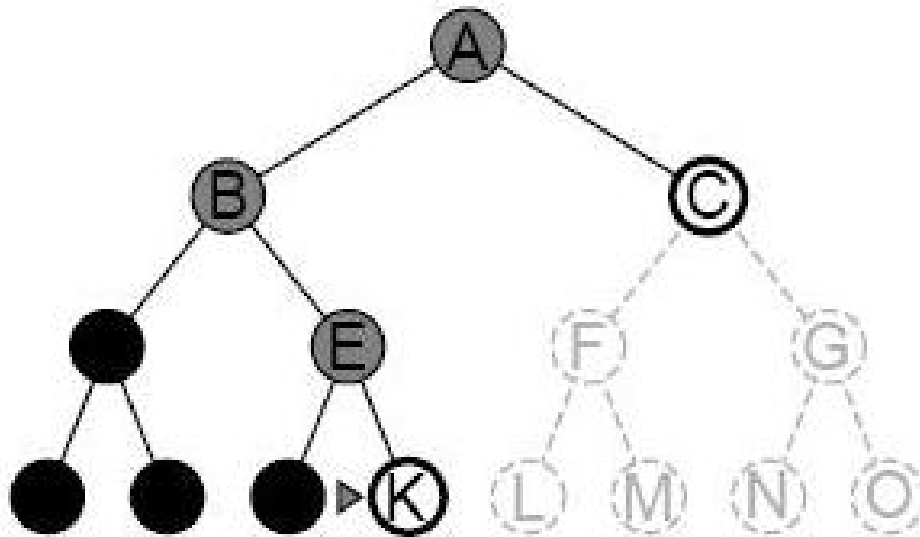
$$L = E - C$$

# Tiefensuche



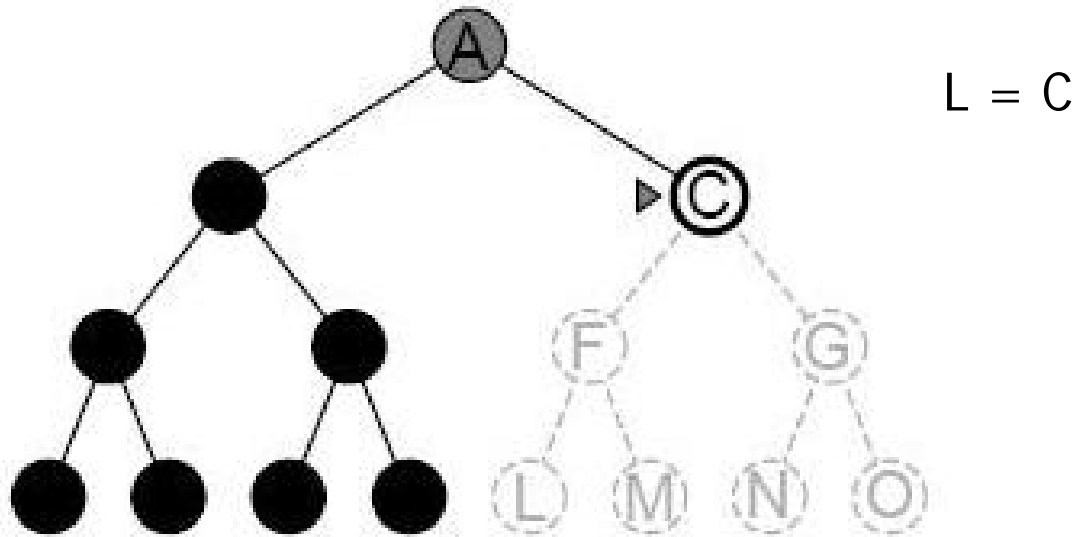
$L = J - K - C$

# Tiefensuche

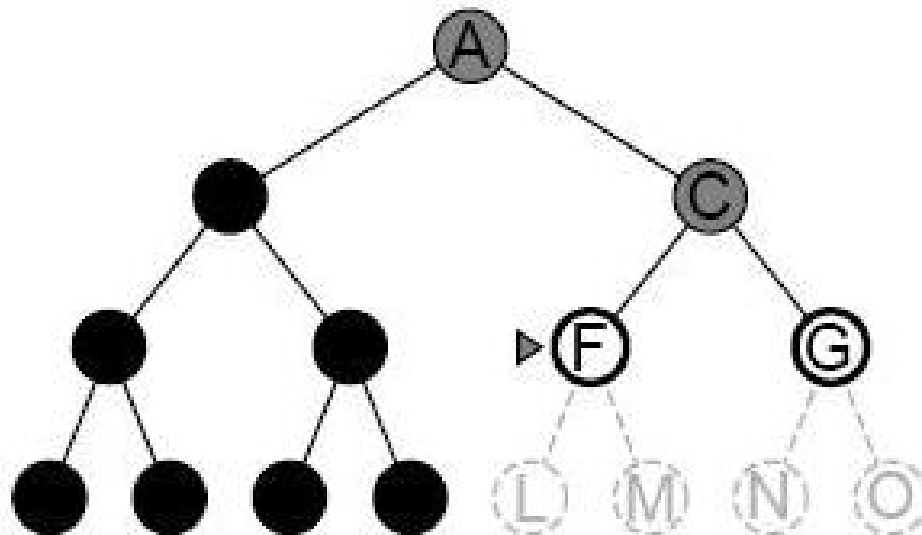


$$L = K - C$$

# Tiefensuche

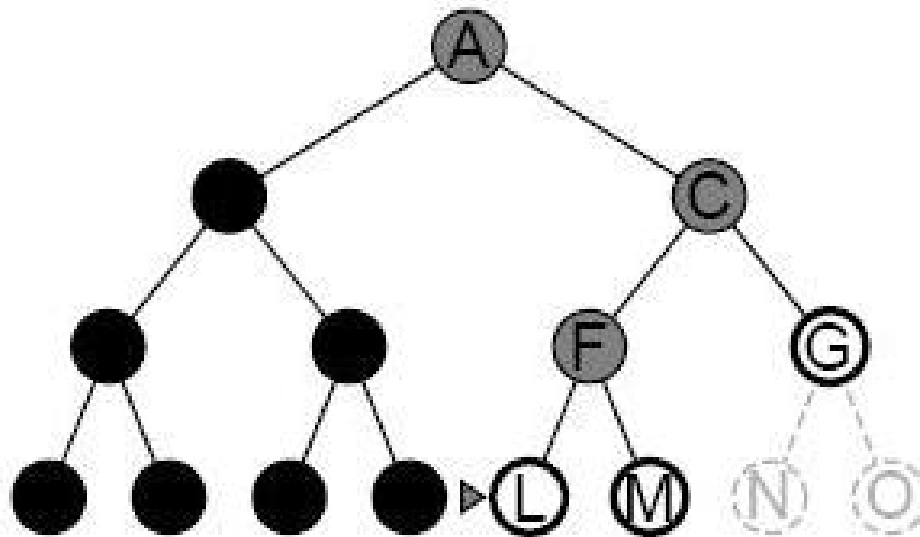


# Tiefensuche



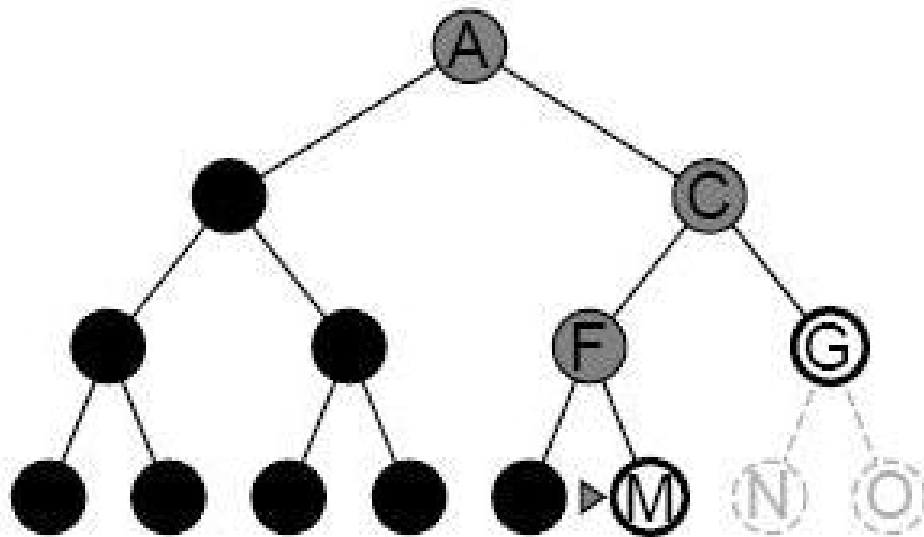
$$L = F - G$$

# Tiefensuche



$L = L - M - G$

# Tiefensuche



$$L = M - G$$



# Tiefensuche - Speicheraufwand

- L = Alle nicht expandierten Geschwisterknoten, die auf dem Suchpfad lagen
  - Sie werden erst beim Aufstieg expandiert

$$\text{Space}(T) = d(b-1) + 1 = O(b \cdot d)$$

- Linear zur Tiefe d !!!



# Tiefensuche - Zeitaufwand

---

- Da Tiefensuche zuerst in untere Ebenen absteigt, wird Zielknoten als erstes im linken Teil des Baums gefunden (best case)
- Ist Zielknoten im rechten unteren Teil des Baums, so müssen vorher alle anderen Knoten durchsucht werden (worst case)



# Tiefensuche - Zeitaufwand

---

- Best case :

$$\text{Time}(T) = d(b-1) + 1 = O(b \cdot d) \quad \text{Linear!!}$$

- Worst case :

$$\text{Time}(T) = \sum_0^d b^k = \frac{(b^{(d+1)} - 1)}{(b-1)} = O(b^d)$$

- Mittlerer Zeitaufwand :

$$\text{Time}(T) = \frac{(b^{(d+1)} + b \cdot d + b - d - 2)}{(2 \cdot (b-1))} = O(b^d)$$



# Suchverfahren der Künstlichen Intelligenz

---

- Grundlagen
  - Zustandsraumrepräsentation
  - Generische Suche
  - Bewertung von Suchstrategien
- **Uninformierte Suchverfahren**
  - Breitensuche
  - Gleiche-Kosten-Suche
  - Tiefensuche
  - **Schrittweise vertiefende Suche**
- Heuristische Suchverfahren
  - Heuristische Schätzfunktionen
  - Bergsteigen

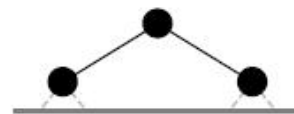
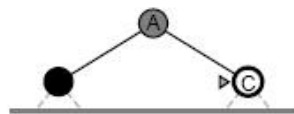
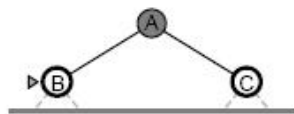
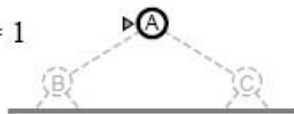


# Schrittweise vertiefende Suche

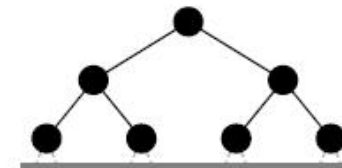
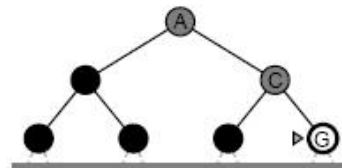
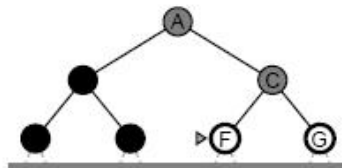
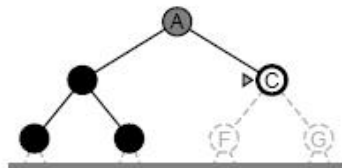
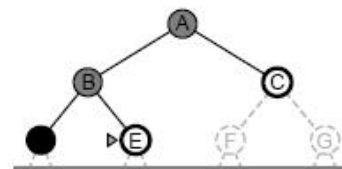
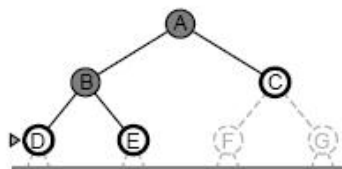
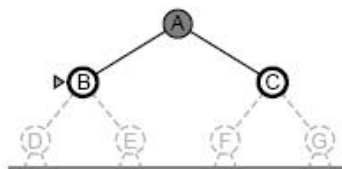
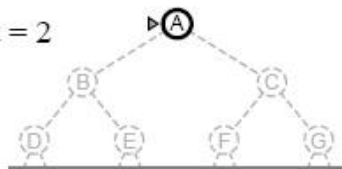
---

- Wie Tiefensuche, allerdings nur bis zu einer maximalen Suchtiefe  $c$
- Wenn keine Lösung gefunden wurde, dann erneutes Suchen vom Wurzelknoten an, bis Tiefe  $c + 1$

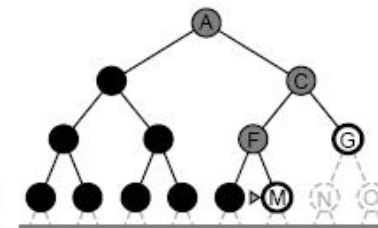
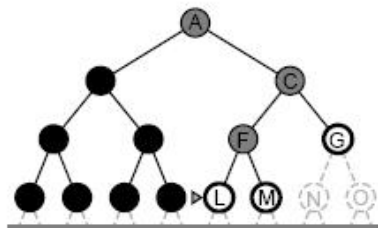
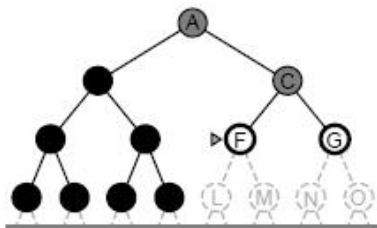
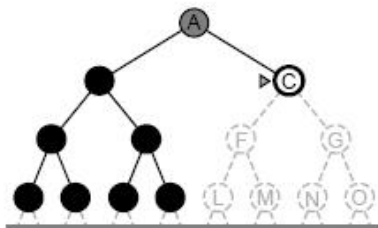
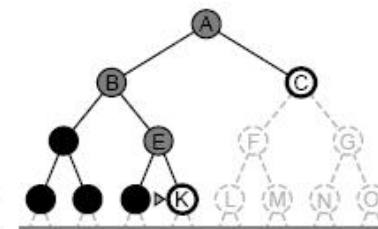
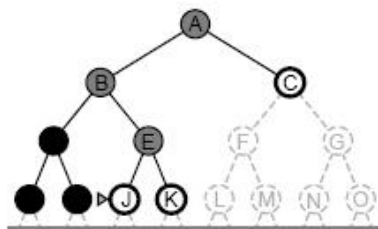
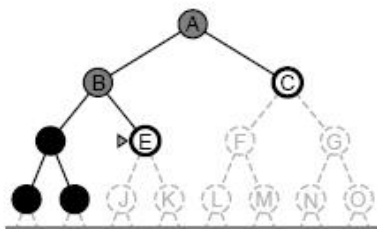
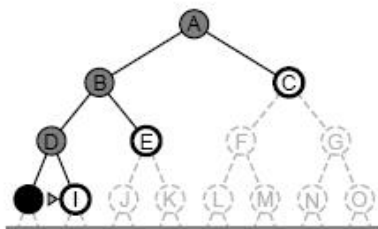
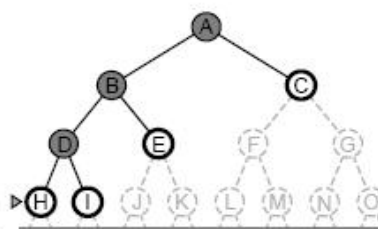
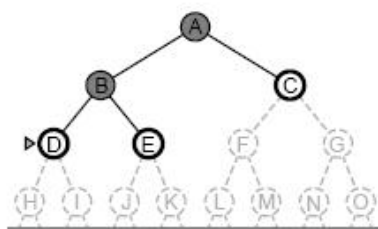
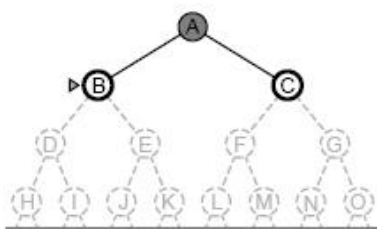
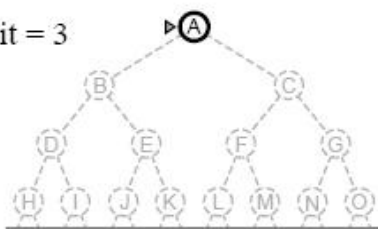
Limit = 1



Limit = 2



Limit = 3





# Schrittweise vertiefende Suche

---

- Mehraufwand durch erneutes Durchsuchen von der Wurzel an für jeden Iterationsschritt
- Da meiste Arbeit in der untersten Ebene, ist für große d der Mehraufwand kaum ein Problem
- Nachteil der Tiefensuche durch Begrenzung der Tiefe aufgehoben

# Uninformierte Suchverfahren

Kriterium	Breiten- suche	Einheitliche Kosten	Tiefen- suche	Beschränk- te Tiefen- suche	Iterative Ver- tiefung	Bidirek- tional (falls möglich)
Vollständig?	Ja <sup>a</sup>	Ja <sup>a, b</sup>	Nein	Nein	Ja <sup>a</sup>	Ja <sup>a, d</sup>
Zeit	$O(b^{d+1})$	$O(b^{\lceil c*/\epsilon \rceil})$	$O(b^m)$	$O(b^l)$	$O(b^d)$	$O(b^{d/2})$
Speicher	$O(b^{d+1})$	$O(b^{\lceil c*/\epsilon \rceil})$	$O(bm)$	$O(bl)$	$O(bd)$	$O(b^{d/2})$
Optimal?	Ja <sup>c</sup>	Ja	Nein	Nein	Ja <sup>c</sup>	Ja <sup>c, d</sup>



# Suchverfahren der Künstlichen Intelligenz

---

- Grundlagen
  - Zustandsraumrepräsentation
  - Generische Suche
  - Bewertung von Suchstrategien
- Uninformierte Suchverfahren
  - Breitensuche
  - Gleiche-Kosten-Suche
  - Tiefensuche
  - Schrittweise vertiefende Suche
- **Heuristische Suchverfahren**
  - **Heuristische Schätzfunktionen**
  - Bergsteigen



# Heuristische Schätzfunktionen

---

- Algorithmus festlegen, der kürzesten Pfad zum Zielknoten abschätzt (Auswahl des Knotens  $n$ )
- Zustände werden bewertet, nicht wie bei Gleiche-Kosten-Suche die Übergangsfunktionen
- Funktion  $h()$  = Schätzfunktion
  - Nicht zu aufwendig
  - Aber genau genug, um Suchfunktion nicht in die Irre zu führen
- $h()$  liefert positiven Wert!  
Je kleiner der Wert, desto näher der Zielknoten



# Heuristische Schätzfunktion

## Beispiel : Schiebepuzzle

---

- 2 Schätzfunktionen:
  - $h1()$  = Alle Steine werden gezählt, die nicht auf ihrem Platz sind
  - $h2()$  = Abstand jedes Steins zur Zielposition wird aufsummiert (Manhattan-Distanz)
- Aufwand :  $h1() < h2()$
- Nutzen ??

# Heuristische Schätzfunktion

## Beispiel : Schiebepuzzle

Bewertung eines möglichen Nachfolgezustandes

<b>1</b>	<b>2</b>	<b>3</b>
	<b>7</b>	<b>4</b>
<b>5</b>	<b>8</b>	<b>6</b>

■  $h1() = 4$

■  $h2() = 2+2+1+2=7$

<b>1</b>	<b>2</b>	<b>3</b>
<b>4</b>	<b>5</b>	<b>6</b>
<b>7</b>	<b>8</b>	

# Heuristische Schätzfunktionen

## Beispiel : Schiebepuzzle

$d$	Search Cost			Effective Branching Factor		
	IDS	$A^*(h_1)$	$A^*(h_2)$	IDS	$A^*(h_1)$	$A^*(h_2)$
2	10	6	6	2.45	1.79	1.79
4	112	13	12	2.87	1.48	1.45
6	680	20	18	2.73	1.34	1.30
8	6384	39	25	2.80	1.33	1.24
10	47127	93	39	2.79	1.38	1.22
12	364404	227	73	2.78	1.42	1.24
14	3473941	539	113	2.83	1.44	1.23
16	–	1301	211	–	1.45	1.25
18	–	3056	363	–	1.46	1.26
20	–	7276	676	–	1.47	1.27
22	–	18094	1219	–	1.48	1.28
24	–	39135	1641	–	1.48	1.26

Empirische Evaluation (Durchschnitt über 100 Beispiele,  $d$ : Lösungstiefe)



# Suchverfahren der Künstlichen Intelligenz

---

- Grundlagen
  - Zustandsraumrepräsentation
  - Generische Suche
  - Bewertung von Suchstrategien
- Uninformierte Suchverfahren
  - Breitensuche
  - Gleiche-Kosten-Suche
  - Tiefensuche
  - Schrittweise vertiefende Suche
- **Heuristische Suchverfahren**
  - Heuristische Schätzfunktionen
  - **Bergsteigen**

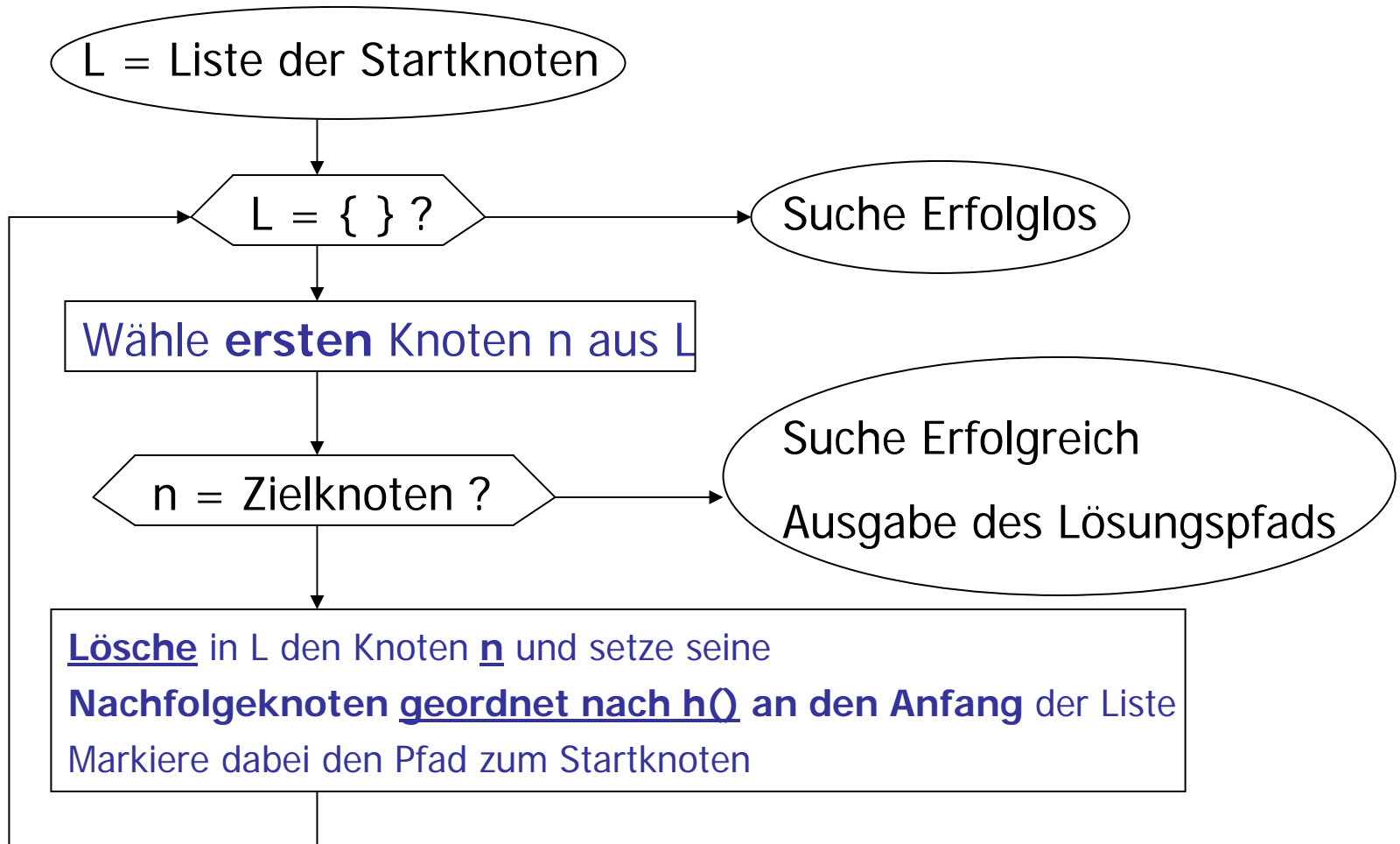


# Bersteigen

---

- Tiefensuche
- Nachfolger des Knotens  $n$  werden nach Heuristik **geordnet vorne** in Agenda  $L$  eingetragen

# Bergsteigen



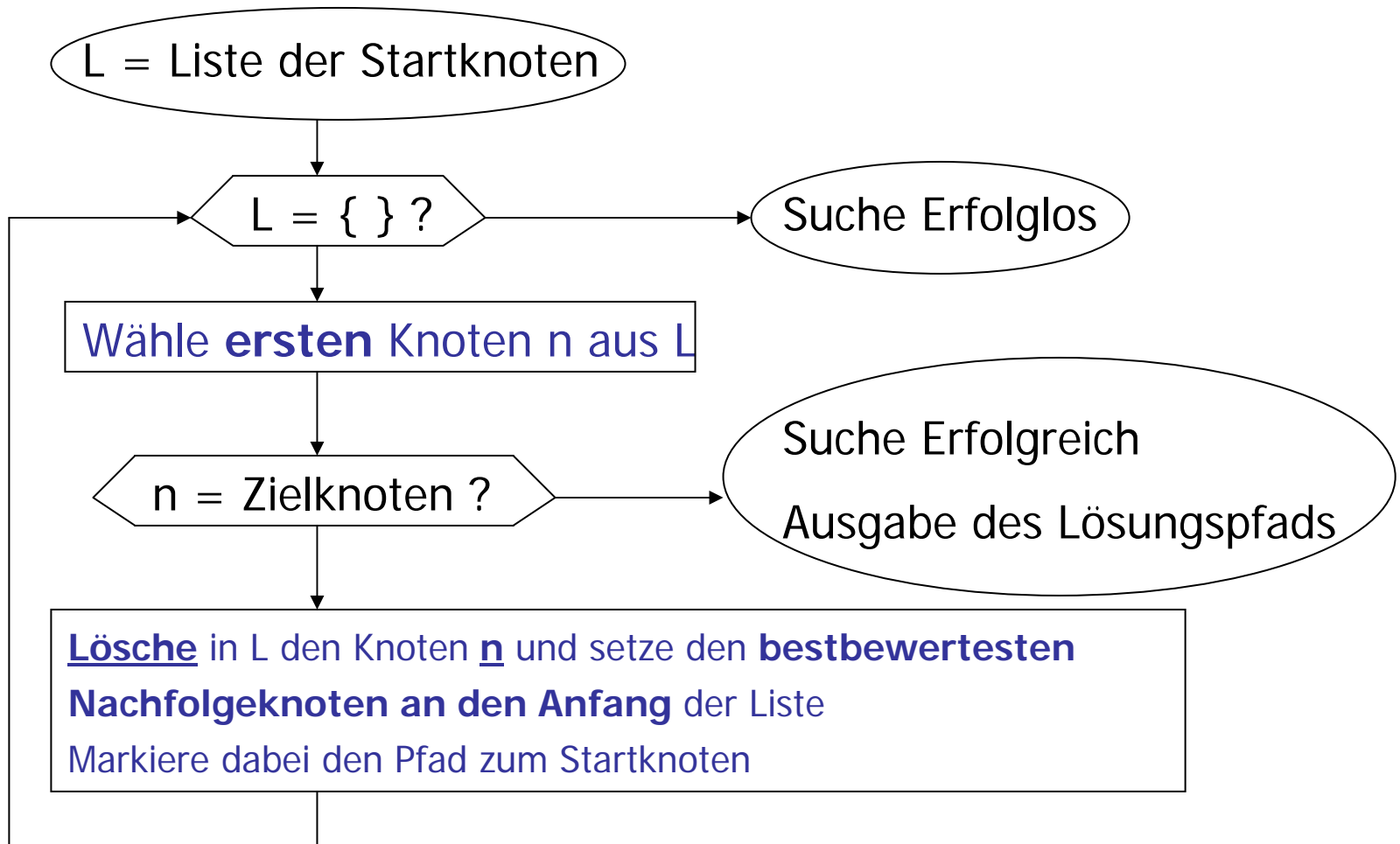


# Optimistisches Bergsteigen

---

- Tiefensuche
- Setzt nur den **bestbewertesten** Nachfolger an den Anfang der Agenda L
- Dadurch kein Aufsteigen im Suchbaum möglich
- Gefährlich, da heuristische Schätzfunktion keinen Fehler machen darf

# Optimistisches Bergsteigen





# Optimistisches Bergsteigen

---

- Bei konstantem Ausgangsverzweigungsgrad :  
 $\text{Space}(\text{opt.B}) = O(b)$
- Problem: lokale Minima und Ebenen beenden die Suche vorzeitig und erfolglos!



# Randomisiertes Bergsteigen

---

- Wenn optimistisches Bergsteigen vorzeitig erfolglos beendet wird:
  - Neue Suche starten mit zufällig gewähltem Anfangspunkt
  - Merken der Besten Endergebnisse jeder Suche
  - Nach gewisser Zeit (abhängig von lokalen Minima) wird globales Minimum gefunden (Lösung des Problems)



# Abschließendes Beispiel : Tic-Tac-Toe

---

<b>X</b>		
	<b>O</b>	
<b>O</b>		<b>X</b>

Ausgangssituation



# Abschließendes Beispiel : Tic-Tac-Toe

---

X	X	
	O	
O		X

X		X
	O	
O		X

X		
X	O	
O		X

X		
	O	X
O		X

X		
	O	
O	X	X

Eigene

Folgezustände



# Abschließendes Beispiel : Tic-Tac-Toe

- Heuristische Schätzfunktion

Kosten	Situation des nächsten Zuges
+1*n	Eigener ALLEIN liegender Stein in Horizont./Vertikal.Diagonal
+5*n	Eigene ZWEI hintereinander liegende Steine in Horizont./Vertikal.Diagonal
+20*n	Gewinnsituation
-1*n	Fremder ALLEIN liegender Stein in Horizont./Vertikal.Diagonal
-5*n	Fremde ZWEI hintereinander liegende Steine in Horizont./Vertikal.Diagonal
-20*n	Verlustsituation

# Abschließendes Beispiel : Tic-Tac-Toe

	5	1	1	1	5	
5	X	X				
		O				
1	O			X		
		1	1			1

$$K = 14 - 10$$

$$K = 4$$



# Ende des Vortrags

---

- Vielen Dank für das Zuhören !



# Quellen

---

## Literatur:

- Handbuch der Künstlichen Intelligenz: Kaptiel 4.1 – 4.2

## Internet:

- <http://www.ki.informatik.uni-frankfurt.de/lehre/WS2002/KI/skript/KI-2Suche.pdf>
- <http://www.techfak.uni-bielefeld.de/~jung/01ki2/Termin01klein.pdf>
- [http://www.pearson-studium.de/media\\_remote/katalog/bsp/3827370892bsp.pdf](http://www.pearson-studium.de/media_remote/katalog/bsp/3827370892bsp.pdf)
- <http://nakula.rvs.uni-bielefeld.de/~mirco/download/KI-Zusammenfassung.pdf>
- <http://www.iicm.edu/greif/images/node1.html>
- <http://www.informatik.uni-ulm.de/ki/Edu/Vorlesungen/GdKI/WS0304/skript/04-Infsuche.pdf>
- <http://fstolzenburg.hs-harz.de/ki/folien/heuristik.pdf>