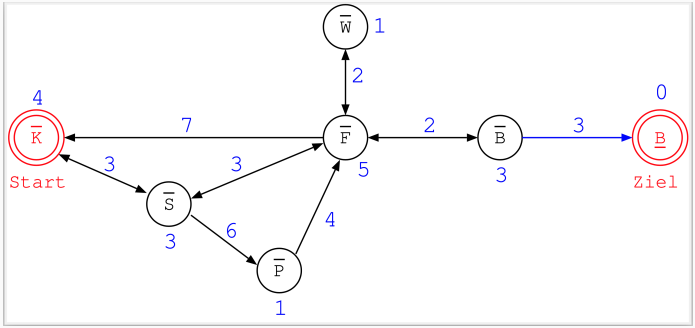
Suche mit Breitensuche

Carsten Gips (HSBI)

Hole das Buch



Uninformierte (“blinde”) Suche:

Keine Informationen über die Kosten eines Pfades: Nur die Pfadlänge (Anzahl der Schritte) zählt.

Breitensuche (*BS*, *BFS*)

1. Füge Startknoten in leere Datenstruktur (Stack, Queue, ...) ein
2. Entnehme Knoten aus der Datenstruktur:
   - Knoten ist gesuchtes Element: Abbruch, melde "*gefunden*"
   - Markiere aktuellen Knoten, und
   - Expandiere alle Nachfolger des Knotens und füge alle unmarkierten Nachfolger, die noch nicht in der Datenstruktur sind, in die Datenstruktur ein
3. Falls die Datenstruktur leer ist: Abbruch, melde "*nicht gefunden*"
4. Gehe zu Schritt 2

=> Was passiert, wenn wir eine **Queue** einsetzen?

Tafelbeispiel Breitensuche (Queue, Suchbaum, markierte Knoten)

Eigenschaften Breitensuche vs. Tiefensuche

	Tiefensuche	**Breitensuche**
Vollständigkeit	nein[1]	ja[2]
Optimalität	nein	ja
Zeitkomplexität	$O(b^m)$	$O(b^{d+1})$
Speicherkomplexität	$O(bm)$	$O(b^{d+1})$

b: Verzweigungsfaktor, **d**: Ebene d. höchsten Lösungsknotens, **m**: Länge d. längsten Pfades

[1] gilt für Tree-Search-Variante; vollständig in Graph-Search-Variante bei endlichem Suchraum
[2] falls b endlich

Praxisvergleich Breitensuche vs. Tiefensuche

Breitensuche: Annahme: $b = 10$, 10.000 Knoten/s, 1.000 Byte/Knoten

Tiefe	**exp. Knoten**	**Zeit**	**Speicher**
2	10^3	0.1 s	1 MB
4	10^5	10 s	100 MB
6	10^7	20 min	10 GB
8	10^9	30 h	1 TB
10	10^{11}	130 d	100 TB

Praxisvergleich Breitensuche vs. Tiefensuche

Breitensuche: Annahme: $b = 10$, 10.000 Knoten/s, 1.000 Byte/Knoten

Tiefe	exp. Knoten	Zeit	Speicher
2	10^3	0.1 s	1 MB
4	10^5	10 s	100 MB
6	10^7	20 min	10 GB
8	10^9	30 h	1 TB
10	10^{11}	130 d	100 TB

Tiefensuche: Annahme: längster Pfad (Tiefe) $m = 1000$

=> Speicherbedarf **ca. 10 MB**

Wrap-Up

- Uninformierte Suchverfahren
  - Keine weiteren Pfadkosten (nur Anzahl der Schritte)
  - Breitensuche: Verfolge alle Pfade (baue den Suchbaum ebenenweise auf)

LICENSE