Suchen in dynamischen Mengen

Ideen und Konzepte der Informatik

Kurt Mehlhorn

max planck institut
informatik

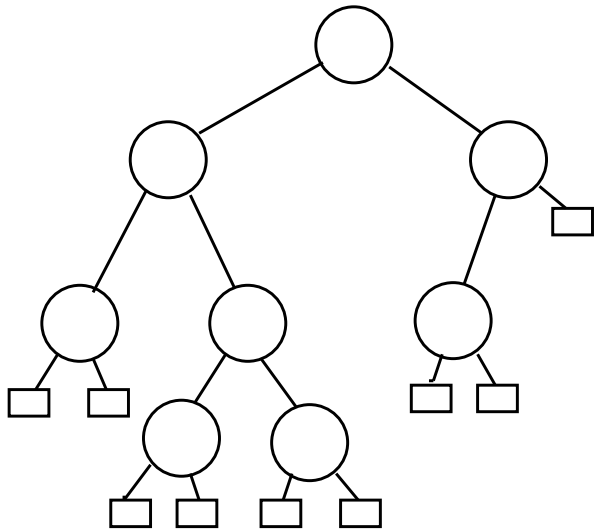
SIC Saarland
Informatics Campus

Suchen in Dynamischen Mengen

- statische Menge: ändert sich nicht im Laufe der Zeit.
- dynamische Menge: ändert sich durch Einfügen und Streichen.
  - Menge der an einem Funkmast angemeldeten Telefone
  - Mitarbeiter einer Firma
  - Kontoinhaber bei einer Bank und ihr Kontostand
- Binärsuche ist gut für die Suche in statischen Mengen; sie ist ungeeignet für die Suche in dynamischen Mengen.
- Für dynamische Mengen braucht man **Suchbäume**.

Geordneter binärer Wurzelbaum

- Wurzel = Knoten ohne Elternknoten
- Blatt = Knoten ohne Kinder (Rechteck)
- Knoten haben einen Elternknoten (außer Wurzel) und zwei Kinder (außer Blättern), linkes Kind und rechtes Kind.
- Kanten sind gerichtet von den Eltern zu den Kindern.
- innere Knoten = Knoten, die keine Blätter sind (Kreis).
- Unterbaum mit Wurzel v = v und dessen Nachkommen
- linker Unterbaum zu v = Unterbaum mit Wurzel *linkesKind*(v)



7 innere Knoten

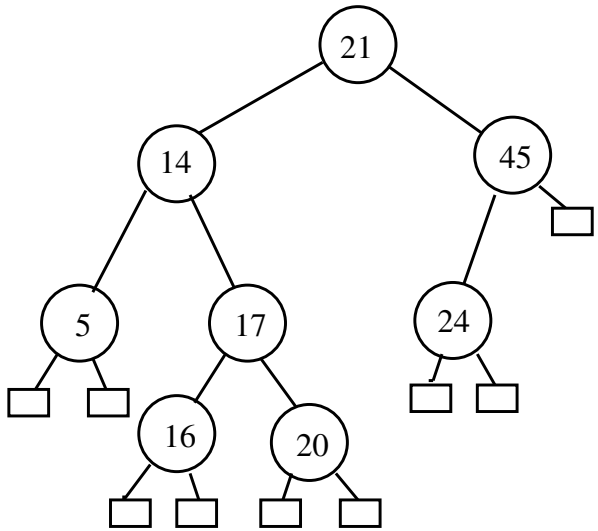
8 Blätter

Suchbaum

- Suchbaum für die Menge $\{5, 14, 16, 17, 20, 21, 24, 45\}$.
- Die inneren Knoten sind mit den Elementen der Menge beschriftet: Schlüssel, Key des Knoten.

  Schlüssel im linken Unterbaum von v < Schüssel in v < Schlüssel im rechten Unterbaum von v.
- Suche nach einem Element e beginnt in der Wurzel.

$v \leftarrow$ Wurzel des Baumes;
while v ist kein Blatt **do**
  if $e = Schluessel(v)$
    HALT(e wurde gefunden);
  if $e < Schluessel(v)$
    $v \leftarrow linkesKind(v)$
  else
    $v \leftarrow rechtesKind(v)$
HALT(e steht nicht im Baum);

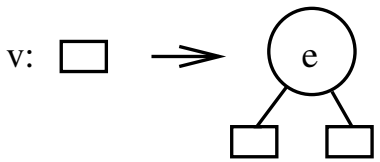


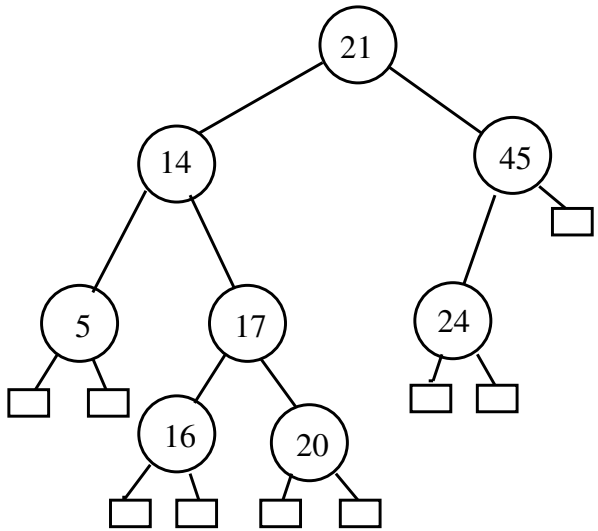
Aufwand der Suche = Tiefe des Baumes (längster Weg von Wurzel zu einem Blatt); kann man logarithmisch in der Größe der Menge halten.

Einfügen eines Elements

- Suchbaum für die Menge $\{5, 14, 16, 17, 20, 21, 24, 45\}$.
  Schlüssel im linken Unterbaum von v < Schüssel in v < Schlüssel im rechten Unterbaum von v.
- Einfügen von $e = 18$: Suche nach e ist erfolglos und endet in einem Blatt. Wir ersetzen das Blatt durch einen inneren Knoten mit Schlüssel e und zwei Kindern.

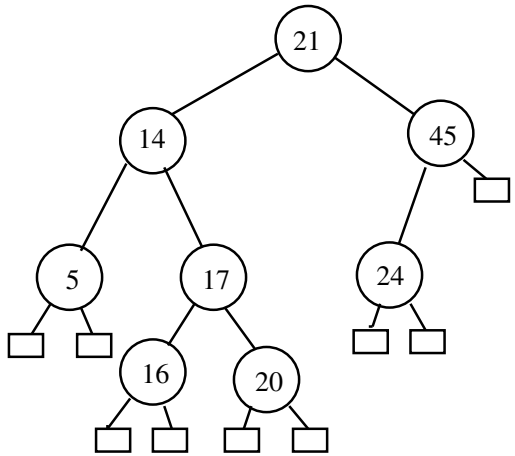
$v \leftarrow$ Wurzel des Baumes;
while v ist kein Blatt **do**
  if $e = Schluessel(v)$
    HALT(e wurde gefunden);
  if $e < Schluessel(v)$
    $v \leftarrow linkesKind(v)$
  else
    $v \leftarrow rechtesKind(v)$
/* v ist ein Blatt */





Realisierung im Rechner

- Wir nummerieren die Knoten durch (beliebig)
- Bauen eine Tabelle mit drei Spalten: Schlüssel, linkes Kind, rechtes Kind.
- In Zeile *i* steht die Information zum Knoten mit der Nummer *i*.
- Unter linkes/rechtes Kind steht Nummer des Kindes. Wenn Kind ein Blatt ist, dann steht -1.



	Key	*lKind*	*rKind*
0			
1			
2			
3			
4			
5			
6			
7			
8			

- Einfügen von *e*: füge zur Tabelle eine neue Zeile dazu, schreibe *e* in die Schlüsselspalte, -1 in die Kindspalten und trage die Nummer der Zeile im Elternknoten ein.

Zusammenfassung

- Binärsuche für statische Mengen, Suchbäume für dynamische Mengen.
- Streichen in Suchbäumen: siehe Aufgaben.
- Zeit für Binärsuche: logarithmisch in der Größe der Menge.
- Zeit für Suche im Suchbaum: Tiefe des Baumes.
- $\log$ Größe $\leq$ Tiefe $\leq$ Größe.
- Man kann Tiefe logarithmisch in der Größe halten (balancierte Bäume); siehe Wikipedia.