Algorithmen 1 SS 2013 – Tutorium 7

9. Tutorium

Sarah Lutteropp

18. Juni 2013

Übersicht

Probeklausur

- Lest die Aufgabenstellungen genauer
- Auf **ALLE** Blätter Name + Matrikelnummer schreiben
- Hinweis auf Rückseite

7. und 8. Übungsblatt

Keine besonderen Anmerkungen.

Motivation (a,b)-Baum

Szenario

Datenmenge ist so groß, dass sie auf der Festplatte gespeichert werden muss.

Anforderung

Wir wollen die Daten so auf der Festplatte organisieren, dass möglichst wenig Paging[1] notwendig wird. (I/O-effiziente Datenstrukturen)

[1] siehe TI

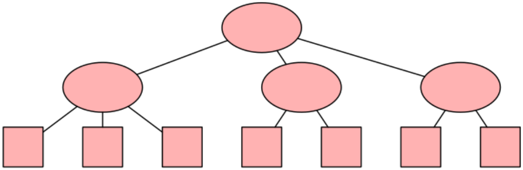
(a,b)-Bäume

Seien $a, b \in \mathbb{N}, a \geq 2, b \geq 2a - 1$.

Definition (a,b)-Baum

Ein Baum heißt (a,b)-Baum, falls gilt:

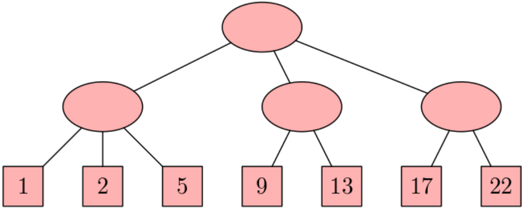
- Jeder innere Knoten hat mindestens a und höchstens b Kinder.
- Alle Blätter haben die gleiche Tiefe.
- Jeder Knoten mit m Kindern enthält genau $m - 1$ Schlüssel.



Ein (2,3)-Baum

(a,b)-Bäume als assoziatives Array

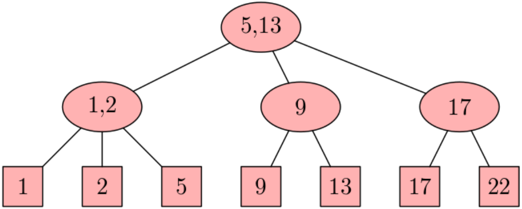
Wir speichern Schlüssel und Datenelemente nur in den Blättern:



Die Schlüssel sind von links nach rechts geordnet.

(a,b)-Bäume als assoziatives Array

Als Suchhilfe erhält ein innerer Knoten mit m Kindern genau $m-1$ Hilfsschlüssel. (Diese sind innerhalb des Knotens sortiert.)



Jetzt kann effizient nach einem Element gesucht werden (wie bei binärem Suchbaum, nur jetzt mit Mehrwege-Entscheidung).

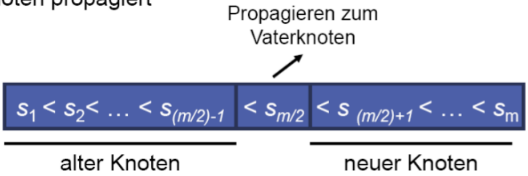
(a,b)-Bäume – Einfügen

Die Blätter enthalten die Schlüssel in aufsteigender Reihenfolge.

- Neues Blatt an richtiger Stelle einfügen
- Problem, falls Elternknoten mehr als b Kinder hat (Überlauf)
- $\Rightarrow$ In zwei Knoten mit $\lfloor (b+1)/2 \rfloor$ und $\lceil (b+1)/2 \rceil$ Kindern teilen
- Jetzt kann Vater überfüllt sein etc.

Bei Uberlauf eines Knotens → Split

- Einträge des Knotens werden auf zwei Knoten verteilt
- Das Objekt, das in der Mitte des übergelaufenen Knotens liegt, wird zum Vaterknoten propagiert



(a,b)-Bäume – Löschen

Die Blätter enthalten die Schlüssel in aufsteigender Reihenfolge.

- Blatt mit Schlüssel entfernen
- Problem, falls Elternknotenweniger als *a* Kinder hat (Unterlauf)
- $\Rightarrow$ Mit einem Geschwisterknoten vereinigen
- Dieser muss vielleicht wieder geteilt werden
- Der nächste Elternknoten kann nun wieder unterbelegt sein

B-Bäume und Datenbanken

Ein B-Baum ist ein $(m, 2m)$-Baum.
Wir wählen m so groß, dass ein Knoten soviel Platz wie eine Seite im Hintergrundspeicher benötigt (z.B. 4096 Byte).
Blätter eines Elternknotens gemeinsam speichern.
Zugriffszeit:
Nur $\mathcal{O}(\log_m(n))$ Zugriffe auf den Hintergrundspeicher.
Wurzel kann immer im RAM gehalten werden.
Falls $m \approx 500$, dann enthält ein B-Baum der Höhe 3 bereits mindestens $500 \cdot 500 \cdot 500 = 125000000$ Schlüssel und Datenelemente.
Jede Suche greift auf nur zwei Seiten zu!

Exkurs: AVL-Bäume

exkurs

Graphenrepräsentation

bar

Tiefensuche

baz

Breitensuche

moep

Vergleich Tiefensuche und Breitensuche

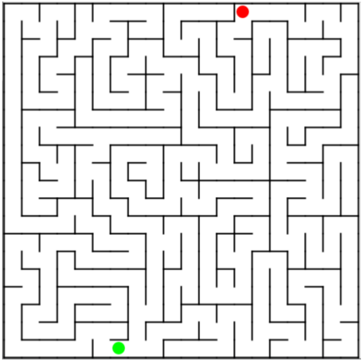
lalala

Exkurs: Iterative Tiefensuche

bla

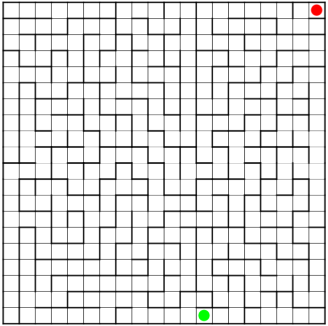
Aufgabe

Minotaurus



Lösungsidee

Minotaurus



Aufgabe in schwieriger

foo

Exkurs: Pledge-Algorithmus

bar

Kreativaufgabe

Dynamisiertes Adjazenzfeld

Gesucht ist eine Datenstruktur für gerichtete Graphen $G = (V, E)$, mit folgenden Eigenschaften:

- Stabile und eindeutige KnotenIDs
- Eindeutige KantenIDs
- Effizienter Wahlfreier Zugriff auf Knoten und Kanten
- Effiziente Navigation
- Amortisiert konstantes Einfügen von Knoten und Kanten
- Amortisiert konstantes Entfernen von Knoten und Kanten

Kreativaufgabe

Stabile und eindeutige KnotenIDs

Knoten sollen durch IDs eindeutig identifiziert werden. Diese IDs sollen Zahlen aus $\mathbb{N}_{\geq 0}$ sein. Dabei seien die KnotenIDs stabil, d.h. die ID eines Knotens ändere sich nie solange dieser Knoten existiert (nach Entfernen eines Knotens darf dessen ID jedoch neu vergeben werden).

Kreativaufgabe

Eindeutige KantenIDs

Die Kanten sollen ebenfalls durch IDs eindeutig identifiziert werden. Allerdings müssen diese nicht unbedingt Zahlen aus $\mathbb{N}_{\geq 0}$ sein und sie müssen auch nicht stabil sein.

Kreativaufgabe

Effizienter Wahlfreier Zugriff auf Knoten und Kanten

Es gibt die Operationen

- *node(u : NodeID) : Handle of Node* und
- *edge(e : EdgeID) : Handle of Edge*,

die in $\mathcal{O}(1)$ Zeit einen Handle auf das Knoten bzw. Kantenobjekt zu einer Knoten- bzw. Kanten-ID liefern.

Kreativaufgabe

Effiziente Navigation

Es gibt die Operationen

- *firstEdge(v : NodeID) : EdgeID* $\cup\{\perp\}$ und
- *nextEdge(e : EdgeID) : EdgeID* $\{\cup\perp\}$,

mit deren Hilfe wie folgt über alle ausgehenden Kanten eines Knoten *v* iteriert werden kann in einem Graph *G*:

$$
\begin{array}{l}
\textbf{for } (\ EdgeID\ \ e := graph.firstEdge(v)\ ;\ e \neq \perp\ ;\ e := nextEdge(e)\ ) \\
\quad h_e := G.edge(e) : Handle \textbf{ of } Edge \\
\quad /*\ do\ something\ */ \\
\textbf{end for}
\end{array}
$$

Sowohl *firstEdge* als auch *nextEdge* dürfen höchstens $\mathcal{O}(1)$ Zeit brauchen.

Kreativaufgabe

Amortisiert konstantes Einfügen von Knoten und Kanten

Es gibt die Operationen

- *insertNode : NodeID* und
- *insertEdge(u, v : NodeID) : EdgeID*,

die in amortisiert konstanter Zeit einen neuen Knoten bzw. eine neue Kanten von u nach v einfügen und jeweils die ID des neu erzeugten Elementes zurückliefern. Beide Operationen dürfen höchstens *amortisiert* konstante Zeit kosten.

Kreativaufgabe

Amortisiert konstantes Entfernen von Knoten und Kanten

Es gibt die Operationen

- *deleteNode(v : NodeID)* und
- *deleteEdge(e : EdgeID)*,

die einen Knoten bzw eine Kante entfernen. Der Einfachheit halber darf ein Knoten dabei nur entfernt werden, wenn bereits alle seine Kanten entfernt worden sind. Beide Operationen dürfen höchstens *amortisiert* konstante Zeit kosten.

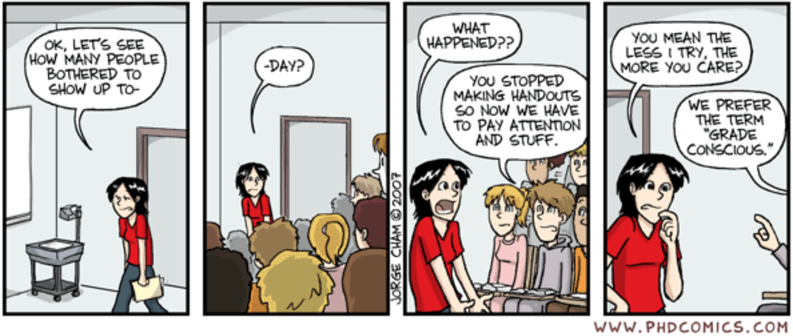
Kreativaufgabe

Dynamisiertes Adjazenzfeld

Gesucht ist eine Datenstruktur für gerichtete Graphen $G = (V, E)$, mit obigen Eigenschaften.

1. Überlegen Sie sich, wie Sie diese Datenstruktur realisieren.
2. Begründen Sie, warum die beschriebenen Operationen in Ihrer Realisierung das geforderte Laufzeitverhalten aufweisen.
3. Wieviel Speicher kann ein Graph mit Ihrer Realisierung im schlimmsten Fall belegen (abhängig von aktuellen oder zwischenzeitlichen Werten von $|V|$ und $|E|$ und das nicht nur im O-Kalkül)? Wieviel im besten Fall? Vergleichen Sie mit dem Speicherverbrauch des statischen Adjazenzfeldes aus der Vorlesung.

Bis zum nächsten Mal.



Disclaimer: Folien zu (a,b)-Bäumen zusammengeklaut aus

- `http://tcs.rwth-aachen.de/lehre/DA/SS2011/handout-2011-05-03.pdf`
- `http://www2.cs.uni-paderborn.de/cs/ag-madh/vorl/DaStrAlg01/folien/DA_6.pdf`