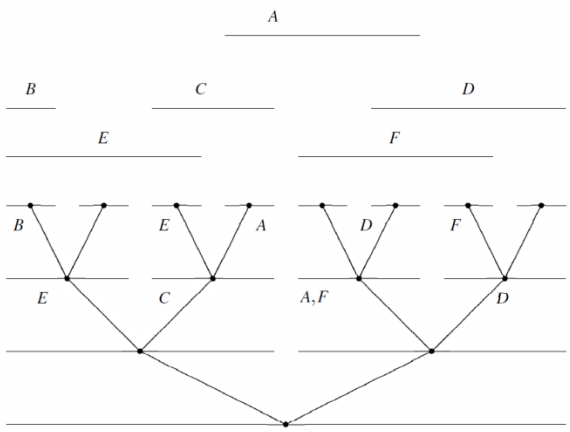
Segment trees

1 Einleitung

Segmentbäume sind eine halbdynamische Skelettstruktur: Zuerst wird eine leeres Skelett aufgebaut, in welches nachher dynamisch Intervalle (Segmente) eingefügt und entfernt werden können. Zudem erlaubt die Datenstruktur effiziente Verarbeitung von sogenannten Aufspiessanfragen; dass heisst, für einen gegebenen Punkt, in welchen Intervallen liegt dieser?

2 Struktur und Operationen

Das Skelett besteht aus einem vollständigen Binärbaum, wobei Intervalle mit Endpunkten aus einer fixen Menge {1,..,n} aufgenommen werden können. Die Blätter stehen für elementare Intervalle [i,i+1], und jeder innere Knoten im Baum repräsentiert die Vereinigung aller elementaren Intervalle in seinem Teilbaum. Die Wurzel repräsentiert also das gesamte Intervall [1,n].



Ein Intervall wird nun folgendermassen im Baum gespeichert: Für jeden Knoten, der ein Intervall repräsentiert, welches vollständig im einzufügenden Intervall liegt und am nächsten bei der Wurzel liegt, wird dort der Name des Intervalls vermerkt. Mit der Überlegung, dass die Endpunkte jedes Intervalls auf zwei Pfaden die sich nur einmal trennen (vgl. range trees) erreichbar sind, lässt sich zeigen dass der Name jedes Intervalls an höchstens *O(log n)* Knoten vorkommt. Der Speicherbedarf ist deshalb *O(n log n).* Die Listen der Namen bei jedem Knoten können in einer linearen Liste verwaltet werden.

2.1 Einfügen und Löschen

Das **Einfügen** eines Intervalls [a,b] ist einfach: Man beginnt bei der Wurzel und geht rekursiv vor. Bei jedem Knoten wird entschieden, ob das von ihm repräsentierte Intervall vollständig in [a,b] liegt. Wenn dem so ist, wird der Name in diesem Knoten gespeichert und man ist fertig. Andernfalls wird für beide Kinder überprüft, ob es gemeinsame Punkte in [a,b] und dem Intervall des Kindes gibt. Wenn dem so ist, wird die Funktion rekursiv für das Kind aufgerufen. Da höchstens logarithmisch viele Knoten betrachten werden beträgt die Laufzeit *O(log n)*.

Beim Löschen kann man grundsätzlich genauso vorgehen. Um den Intervallnamen in der verketteten Liste jedoch zu finden und entfernen sind schlimmstenfalls linear viele Schritte nötig; eine nicht akzeptabler Aufwand. Das Problem wird mit einer Sekundärstruktur für die Intervallnamen gelöst: In einem Wörterbuch, welches Einfügen und Entfernen in logarithmischer Zeit erlaubt (z.B. ein balancierter Binärbaum) werden alle Namen der Intervalle gespeichert. Jeder Eintrag zeigt auf eine lineare Liste mit Zei-

gern zu jedem Vorkommen des Namens in der Grundstruktur. Damit ist auch das Entfernen in logarithmischer Zeit möglich.

2.2 Aufspiessanfragen (stabbing query)

Um eine Aufspiessanfrage für einen gegeben Punkt x zu beantwortet, kann wiederum rekursiv vorgegangen werden:

```
procedure report(p: Knoten; x: Punkt)
    gebe alle Intervalle von p aus
    if (p ist Blatt) fertig;
    else
       if (p hat linken Sohn s mit Intervall, das x enthält)
           report(s,x);
       if (p hat linken Sohn s mit Intervall, das x enthält)
           report(s,x);
end
```