Prüfungsprotokoll Kurs 1663 Datenstrukturen und Algorithmen

Prüfer: Prof. Dr. Güting
Datum: 12.02.2018
Dauer: ca. 30 min
Note: 2,3

1. Erklären Sie bitte Listen, Stacks, Queues und die Unterschiede.
    1. Habe einfach und doppelt verkettete Listen erklärt, mit OPS insert, delete, find/search
    2. Stacks und Queues mit OPS insert, delete, find/search (im jeweiligen Sprachgebrauch top,pop,push/front,enqueue,dequeue), Unterschiede zwischen Stacks und Queues bzgl. LIFO/FIFO
2. Erklären Sie, was Priority Queues sind.
    1. Definition, Ordnung auf Elementen, nach denen die Menge aufgebaut ist.
    2. Wie können diese realisiert werden? - AVL-Baum, ist aber nicht effizient, daher partiell geordneter Baum (Heap)
3. Was ist ein partiell geordneter Baum / Heap?
    1. Definition, insb. Beziehung zwischen jedem Vaterknoten und seinen Söhnen
    2. Wie kann ein Heap implementiert werden?
        1. Array
    3. Wie werden die Knoten dargestellt? Wie wird der Arrayindex berechnet?
        1. Arrayindizierung, Knoten inorder in Array eintragen
        2. Berechnung: Formel aus KE (2i und 2i+1)
4. Mengenoperation: Durchschnitt.
    1. Er wollte letztendlich auf 2 geordnete Listen hinaus, die man parallel durchläuft, um eine Schnittmenge zu finden.
5. Verfahren, um Mengen zu sortieren: Divide and Conquer
    1. Wie funktioniert das Prinzip im Allgemeinen? Code aus der KE
6. MergeSort:
    1. Wie funktioniert der MergeSort?
    2. Wie berechnet sich die Laufzeit/der Aufwand des Mergesorts im O-Kalkül?
        1. Divide-Schritt, Conquer-Schritt, Merge-Schritt
           $T(n) = T(1) + 2*T(n/2) + c*n$

Prüfungsprotokoll Kurs 1663 Datenstrukturen

Prüfer: Prof. Dr. Güting
Datum: 14.09.2005
Dauer: 25 min
Note: 1,0

Dictionary

(i) Was ist ein Dictionary?
Datentyp um Mengen darzustellen mit den Operationen einfügen, löschen und finden.

(ii) Welche Datenstrukturen gibt es für die Darstellung eines dictionary?
Ich habe den Bitvektor erklärt. Es wurde nach der Laufzeit der Operationen gefragt (O(1)). Was ist der Nachteil? (beschränktes Universum, hoher Platzbedarf)
Und? Hohe Laufzeit zum Aufbau (Initialisierung).

(iii) Zählen Sie weitere Datenstrukturen auf.
Geordnete Liste, Hashing (was ist das? kurz die Idee erklärt, dann meinte Herr Güting, vielleicht kommen wir später darauf zurück, kamen wir aber nicht), binäre Suchbäume, AVL-Bäume.
Und? B-Bäume.

B-Bäume

Wie ist der B-Baum definiert?
Erst Definition Vielweg-Suchbaum angegeben. Dann Definition B-Baum.
Wie hoch ist ein B-Baum? Hier habe ich die Höhe berechnet wie im Skript.

Suchalgorithmen

Kommen wir zu Suchalgorithmen. Erklären Sie Heapsort?
Heapsort anhand der Implementierung in einem Array erklärt.
Wie ist die Laufzeit? Phase 1 sogar in $O(n)$ Zeit, Phase 2 in $O(n \log n)$ Zeit. Insgesamt Laufzeit also in $O(n \log n)$.

Plane Sweep

Was ist Plane Sweep? Erst allgemeine Idee erklärt, dann anhand eines Beispiels (Segmentschnitte). Welche Datenstrukturen werden verwendet und erklären Sie daran die Laufzeit. Habe ich wie im Skript beschrieben erklärt.

Fazit

Die Prüfung fand in einer netten und ruhigen Atmosphäre statt. Die Prüfungsprotokolle waren eine große Hilfe. Natürlich ist es wichtiger eine Sache verstanden zu haben, als Details auswendig zu können. Herr Güting ist als Prüfer zu empfehlen, aber man muss sich auch auf Fragen zur Geometrie gefasst machen.

Prüfer: Prof. Dr. Güting
Beisitzender: Dr. Spiekermann
Prüfungsfach: Datenstrukturen
Datum: 12. Jänner 2005
Prüfling: Mag. pharm. Gunther Hebein
Note: 1,0 ☺

1. Was ist ein Dictionary
    - Datentypen
        i. Bin Suchbaum (Definition, Höhe)
        ii. Welche Weiterentwicklung daraus? (AVL-Baum mit Def., Höhe, Rotation, warum nach Rotation sicher noch geordnet?)

2. Andere Mengenrepräsentationen aufzählen
    - Mengen mit Schnitt, Vereinigung, Differenz
        i. Implementierung
        ii. Algorithmen für die Operationen anhand Bitvektor (wie ist die Logik für die Operationen)
        iii. Alg. Anhand zweier geordneter Listen

3. Sie sollen ein neuartiges Fenstersystem für ein neues Betriebssystem entwickeln.
   Benutzer klickt Bildschirm, Fenster wird dadurch ausgewählt. Wie implementieren?
    - SS / SI-Baum und Stack oder queue (wenn mehrere Fenster hintereinander sind)
    - Wie viele Knoten sind in einem Level des Baumes maximal markiert?

Summa summarum ein netter Prüfer, der versucht eine gute Atmosphäre bei der Prüfung zu schaffen. Wenn man nicht weiterkommt bekommt man einen Tipp oder Prof. Gütig erklärt es einem.
Mit der Benotung bin ich sehr zufrieden, hätte eigentlich die Tage zuvor mit einer 4,35 gerechnet.

Gedächtnisprotokoll zur Vordiplomsprüfung Datenstrukturen

Inhalt: Kurs 1663/SS04
Termin: 22.12.2004
Prüfer: Prof. Güting
Beisitzer: D. Ansorge
Dauer: ca. 25 Minuten
Note: 1,3

Themen:

1. Sortieren
Wie lassen sich Sortieralgorithmen klassifizieren? – Hier bin ich auf 4 der 5 Klassifizierungen eingegangen und habe Beispiele und Aufwände angegeben. Nur die Klassifizierung „in situ“ war mir im Eifer des Gefechts entfallen, auf Nennung durch Prof. Güting konnte ich jedoch direkt die klassischen Beispiele für die Unterscheidung Heapsort/Mergesort nennen und erläutern.

Wie läuft Heapsort ab? – Aufbau des Heaps, Darstellung im Array

Warum ist der „hintere“ Teil des Arrays bereits ein „Teilheap“? – Blätter ohne Söhne, also Kriterium erfüllt.

Wie leiten Sie den Aufwand für Heapsort her? – Aufbau Phase 1 O(n*logn) + Phase 2 O(logn). Hinweis von Prof. Güting, dass Phase 1 nur einen Aufwand von O(n) hat. Ich stimmte dem zu, „es kann die Summe abgeschätztwerden zu 2...“. Meine Erwähnung dieses Sachverhalts reichte hier aus.

2. Plane Sweep
Was ist das? – Hier habe ich den Grundsätzlichen Aufbau aller Plane Sweep Algorithmen mit Plane Status- und Plan Event-Struktur beschrieben. Als Beispiel wählte ich das orthogonale Segmentschnittproblem.

Bitte beschreiben Sie mir den Algorithmus und den Aufbau der Datenstrukturen. – Menge für die Events, mod. AVL-Baum für die Status-Struktur.

Wie leiten Sie den Aufwand her? – Aufbau und Sortierung der Haltepunkte (n logn), mit n=|H|+|V|. Suche im AVL-Baum O(logn), um das Blatt zu erreichen, das dem Intervallanfang entspricht. O(t), um die Intervalle „abzuklappern“, also t-Schnitte von horizontalen und vertikalen Segmenten.

Wie sieht das denn nun aus, wenn nicht alles so schön orthogonal/stabil ist? – Hier habe ich die grundsätzliche Vorgehensweise beim allgemeinen Segmentschnitt beschrieben. Prof. Güting hätte auch noch gerne Details zur Datenstruktur gewusst (Priority Queue, balancierter Baum) aber ich stand etwas auf dem Schlauch, da ich den Begriff Priority Queue schon im Zusammenhang mit den vorherigen Fragen benutzt hatte, als auch die Zeit knapp wurde.

3. Fazit
Ich kann mich nur allen bisherigen Meinungen zu Prof. Güting anschliessen. Er ist ein ruhiger Prüfer. Er legt mehr Wert auf das Verständnis, als die exakte Formulierung. Es hilft, von sich aus Details zu erwähnen, die nicht in den Antworten auf die Fragen enthalten sein müssen und Zusammenhänge herzustellen, um damit das Verständnis der Sachverhalte zu dokumentieren.

Prüfungsprotokoll 1663

Prüfer: Prof. Dr. Güting
Beisitzer: Hr. Ansorge
Datum: 8.10. 2003
Dauer: ca. 30 Min

Fragen:

Was ist ein binärer Baum ?
Definition angeben.
Was ist die Höhe eine Baums ?
Wie hoch wird ein Baum ?
Also die max. Höhe und die min. Höhe angeben und begründen.
Wie kann man die Entartung verhindern ?
Ausbalancieren mit AVL- Bäumen.
Wie funktioniert eine Rotation ?
Ich sollte ein Beispiel zeichnen nach eigener Vorstellung zeichnen. Er wollte es wie im Kurs haben mit den Teilbäumen.
Was ist ein Segment-Baum ?
Definition angeben und allg. beschreiben.
Wieviel Einträge macht ein Intervall in einem Segment-Baum auf einem Level maximal ?
Hatte keine Ahnung. Nach längerem Hin- und Her sagte er dann 2.
Wieviel Speicherplatz verbrauchen n Intervalle die im Segment-Baum eingetragen werden ?
Wie wird nach einem Punkt im Segment-Baum gesucht ?
Man geht halt den Pfad zum Query-Punkt durch und gibt die dabei gefunden Intervalle aus.
Wie ist die Laufzeit dafür ?
O(log N + k).

Eindruck:

20 von 30min bestanden aus dem Segmentbaum ☹. Ich hatte hier eine Lücke die „bearbeitet“ wurde. Es ist natürlich mein Pech, dass er gerade diese fand. Ich empfand es sehr schade, dass die ganze Prüfung nur über Bäume ging. Man bereitet soviel vor und es kommt nur ein Thema dran. Ich hatte irgendwann vor lauter Bäumen den Wald nicht mehr gesehen.
Und dennoch: Prof. Güting hat eine sehr positive Ausstrahlung und ich fühlte mich trotz der vielen Lücken die ich Segmentbaum hatte sehr wohl in der Prüfung. Ich glaube dieses „Nachbohren“ dient dazu dem Prüfling eine Chance zur Korrektur zu geben. Ich bin durch meine Aufregung dann irgendwann mal gar nicht mehr durchgestiegen.
Die Note 2,0 fand ich unglaublich gut !!!
Also: wer „Glück“ hat und in seiner Prüfung keine Lücken aufzufinden sind, der hat mit Prof. Güting eine der besten Wahlen getroffen.

Gedächtnisprotokoll zur Diplomvorprüfung Datenstrukturen

Prüfungsinhalt:	01663 – Datenstrukturen
Prüfer:	Prof. Güting
Beisitzer:	Dr. Dieker
Zeit:	August 2001
Dauer:	ca. 30 min
Note:	1,0

- „Datenstrukturen. Ein zentrales Gebiet sind die Sortierverfahren. Nach welchen Kriterien kann man denn **Sortierverfahren klassifizieren**?“

  - intern/extern
  - methodisch
  - nach Effizienz
  - im Array oder nicht
  - allgemeine/eingeschränkte Verfahren

  In Anlehnung an die Ausführungen im Kurs (KE 4, S.153) habe ich die einzelnen Kriterien jeweils näher erläutert.

- „Was ist ein allgemeines Sortierverfahren?“

  Ein allgemeines Verfahren soll in der Lage sein, eine beliebige Menge von Werten über einem linear geordneten Wertebereich zu sortieren. Man nimmt deshalb an, dass ein allgemeines Verfahren nur eine einzige Möglichkeit hat, auf Schlüsselwerte zuzugreifen, nämlich einen Vergleichsoperator < (oder ≤) auf sie anzuwenden.

- „Was weiß man sonst noch über diese Werte?“

  Nichts.

- „Wann heißt ein Sortierverfahren stabil?“

  Ein Sortierverfahren heißt stabil, wenn sichergestellt ist, dass die Reihenfolge von Sätzen mit gleichem Schlüsselwert nicht verändert wird, also in der Ergebnisfolge die gleiche ist, wie in der Ausgangsfolge.

- „Beschreiben Sie den Algorithmus für das **einfache Sortieren durch Auswählen**!“

  Aus der noch unsortierten Teilfolge wird jeweils das Minimum entnommen und an die sortierte Teilfolge angehangen (Beschreibung des Algorithmus auf S. 154, KE 4).

- „Und wie implementiert man diesen Algorithmus *in situ*?“

  Beschreibung der Prozedur auf S. 155, KE 4.

  - Implementierung mit zwei for-Schleifen
  - äußere Schleife: der Bereich der unsortierten Teilfolge wird jeweils festgehalten; Vorbesetzung der Variablen *min* (für das Minimum) und *minindex* (für den entsprechenden Index)
  - innere Schleife: die unsortierte Teilfolge wird jeweils von Anfang bis Ende durchlaufen, um das Minimum zu finden

  Prof. Güting: „Und dann?“

  Vertauschen von S[*i*] und S[*minindex*].

- „Welche Laufzeit hat das einfache Sortieren durch Auswählen?“

  $O(n^2)$.

- „Warum $O(n^2)$? Die noch unsortierte Folge wird doch bei jedem Schleifendurchlauf kürzer, so dass man nur beim ersten Durchgang alle n Elemente betrachten muss.“

  Ja, man muss beim ersten Durchgang in der inneren Schleife n Operationen durchführen, beim zweiten Durchgang n-1, beim dritten n-2, usw. Daraus ergibt sich aber für die Laufzeit folgendes:

  $$n + (n-1) + (n-2) + ... + 1 = \sum_{i=1}^{n} i = n(n + 1) / 2 = O(n^2) \qquad \text{(KE 4, S. 155)}$$

- „Und wie geht das „schlaue“ Sortieren durch Auswählen?“

  Hiermit war der **Heapsort** gemeint. „Schlau“ deshalb, weil der Heapsort zum Auffinden des Minimums einen partiell geordneten Baum benutzt. Dadurch wird die Laufzeit von $O(n^2)$ auf $O(n \cdot \log n)$ gesenkt. Dies ist eine asymptotisch optimale Sortiermethode, denn $\Omega(n \cdot \log n)$ ist eine untere Schranke für allgemeine Sortierverfahren im worst-case.

- „Was ist ein partiell geordneter Baum?“

  Hier habe ich die formale Definition aufgeschrieben und erläutert. Eine verbale Umschreibung hätte auch genügt.

- „Wie implementiert man den partiell geordneten Baum im Array?“

  Sei ein innerer Knoten des partiellen Baumes in dem Feld S[j] abgelegt. Dann wird der rechte Sohn dieses Knotens in dem Feld S[2j] abgelegt und der linke Sohn in S[2j +1].

- „Wie wird der Heapsort im Array implementiert?“

  1.) Die n zu sortierenden Elemente werden in einen Heap eingefügt. $S[\lfloor n/2 \rfloor +1 ... n]$ (also die rechte Hälfte des Arrays) ist bereits ein Teilheap. Deshalb durchläuft man das Array nur von Index $i := \lfloor n/2 \rfloor$ bis 1 und erweitert den Teilheap S[(i+1) ... n] jeweils zu einem Teilheap S[i ... n] durch „Einsinken lassen“ von S[i].

  2.) Um den Aufbau des Heaps zu beschleunigen und *in situ* sortieren zu können, wird die sortierte Folge rückwärts am hinteren Ende des Arrays, welches den Heap enthält, eingetragen: Dazu wird dem Heap n-mal das Minimum entnommen und im hinteren Teil eingetragen (durch Vertauschen). Nach jeder Entnahme muss ein *reheap* durchgeführt werden, damit das nach vorne getauschte Element an die richtige Stelle im Heap einsinken kann.

- „Warum durchläuft die Schleife beim erstmaligen Aufbau des Heap nur die linke Hälfte des Arrays?“

  Weil die in der rechten Hälfte abgelegten Knoten keine Söhne haben. Dies ist somit bereits ein Teilheap.

- „Wie ist die Laufzeit für ein Mal einsinken?“

  O(logn).
  Ich habe außerdem noch erwähnt, dass der erstmalige Aufbau des Heap in einer Laufzeit von O(n) realisiert werden kann. Die nachfolgenden *reheaps* benötigen jeweils O(logn) also insgesamt $O(n \cdot \log n)$ Zeit.
  In diesem Zusammenhang führte Prof. Güting dann selbst aus, dass man mit dem Heapsort in $O(n + k \cdot \log n)$ Zeit die k kleinsten Elemente einer Menge in Sortierreihenfolge erhalten kann. Wenn nun $k \leq n / \log n$ ist, benötigt man hierzu nur O(n) Zeit.

- „Wie funktioniert das Einsinken?“

  Auf jeder Ebene werden zwei Entscheidungen getroffen:

  1.) Welcher Pfad ist im Heap zu verfolgen?
  Man wählt hier den kleineren der beiden Söhne des betrachteten Knotens.

2.) Muss das betrachtete Element noch tiefer sinken?
Mit dem unter 1.) ausgewählten Sohn wird nun verglichen. Ist das betrachtete Element kleiner als der Sohn (oder gleich), so befindet es sich bereits an der richtigen Stelle und muss nicht weiter „einsinken“. Ist das Element größer, dann muss es mit dem Sohn vertauscht werden und sinkt eine Ebene tiefer.

- „Dann geh'n wir mal zu einem anderen Thema. Was ist denn **Plane-Sweep**?“

  Hier habe ich am Beispiel des Segmentschnitt-Problems alles zu Plane-Sweep erzählt, was mir einfiel (KE 6, S. 246 – 250).

  - Gegeben: eine Menge horizontaler und vertikaler Liniensegmente
    Problem: Finde aller Paare sich schneidender Segmente.
    Verfahren: Eine vertikale Gerade (*Sweepline*) wird von links nach rechts durch die Ebene geschoben. Dabei „beobachtet“ man den Schnitt der Geraden mit der Objektmenge.
  - Reduktion eines k-dimensionalen Mengenproblems auf ein (k–1)-dimensionales Suchproblem
  - *Sweepline-Status-Struktur*: muss dynamisch sein
  - *Sweep-Event-Struktur*: ist bei der Behandlung orthogonaler Objekte in der Regel statisch; sie kann dynamisch sein, z.B. beim Segmentschnitt-Problem für nicht orthogonale Objekte
  - Implementierung mit einer Variante des AVL-Baumes (verkettete Blätter)
  - Laufzeit: $O(n \cdot \log n + k)$ mit n = Anzahl der Segmente und k = Anzahl der Schnittpaare; für die Laufzeitanalyse hat man also *zwei* Parameter n und k
  - Das Segmentschnittproblem ist ein Beispiel dafür, dass man als Maß für die „Komplexität“ der Eingabe nicht nur die Kardinalität der Eingabemenge, sondern ggfs. noch weitere Kriterien benutzt (hier: Anzahl der Schnittpunkte k).

  An dieser Stelle hat mich Prof. Güting dann unterbrochen und kam zu seiner letzten Frage:

- „Kann man das Segmentschnitt-Problem auch auf andere Art lösen?“

  Ja, mit **Divide-and-Conquer** (KE 6, S. 263 – 270).

  - *getrennte Darstellung* planarer geometrischer Objekte
  - rekursive Invariante
  - Beschreibung des Algorithmus und graphische Darstellung des *Merge*-Schrittes
    Aussonderung der horizontalen Segmente, die sich im *Merge*-Schritt treffen
    Implementierung mit verketteten Listen:
    - nach y-Koordinaten sortierte verkettete Listen horizontaler Segmente
    - nach unteren Intervallgrenzen sortierte verkettete Listen vertikaler Segmente
    - paralleler Listendurchlauf

Fazit:
Also, ich kann die bisherigen Prüfungsprotokolle nur bestätigen. Prof. Güting ist als Prüfer uneingeschränkt zu empfehlen. Er ist ein sehr angenehmer und ruhiger Prüfer. Bei jeder Frage hatte ich ausreichend Zeit zu überlegen und meine Antworten zu formulieren. Auch habe ich es als positiv empfunden, dass ich im zweiten Teil der Prüfung (geometrische Algorithmen) alles zum Thema erzählen konnte, was mir einfiel.

Vorbereitung:
Man sollte sich intensiv mit den im Kurs dargestellten Algorithmen und Verfahren auseinandergesetzt haben und vor allem darauf achten, dass man diese wirklich *verstanden* hat. Bloßes Auswendiglernen ist hier meines Erachtens nicht der richtige Weg.

Mit der Benotung bin ich natürlich mehr als zufrieden.

VIEL GLÜCK bei Euren Prüfungen !!!

Diplomvorprüfung Prüfungsprotokoll mündliche Prüfung

Datenstrukturen Kurs 1663

Prüfer: **Prof. Güting**

Termin: **4.10.1999**

Note: **2.0**

Fragen

Sortierverfahren

- Klassifikation von Sortierverfahren
  - intern/extern
  - methodiscch
  - nach Effizienz
  - im Array oder nicht
  - allgemeine und eingeschränkte

Einfache Sortierverfahren

- InsertionSort und SelectionSort
- Laufzeiten, und deren Herleitung aus den Algorithmen

Heapsort

- Algorithmus erklären und auf Laufzeit schliessen

Geometrische Algorithmen

- Plane-Sweep
  - Algorithmus und auf Laufzeiten schliessen erklären der Sweep-Event- und Sweepline-Status-Sruktur

Eindruck

Prof. Güting ist ein sehr ruhiger Prüfer. Er legte vor allem Wert auf die Ableitung der Laufzeiten aus den Algorithmen, das mir etwas Probleme bereitete. Die Angabe exakter Formulierungen ist dabei jedoch nicht nötig. Er formuliert die Fragen genau und lässt dem Prüfling viel Zeit zum Antworten. Die Benotung empfand ich sehr gut.

Vordiplomsprüfung Datenstrukturen 1663

Termin: Oktober 1999

Prüfer: Prof. Dr. Güting

Dauer: ca. 20-25 min

Themen:

1. als 'Standardthema': Hashing:

Hier ist Prof. Güting nicht so sehr in die Tiefe gegangen. Es genügte ihm wohl mit ein paar Stichproben einen Eindruck zu bekommen, ob und wie gut ich das Thema beherrschte (Was ist Hashing, offenes und geschlossenes Hashing: wie funktioniert es, wozu wird es verwendet, Vor- und Nachteile von Hashing).

2. als 'Vertiefungsthema': Plane-Sweep und DAC bei geometrischen Algorithmen:

Hier ging es zunächst darum die Plane-Sweep-Technik zu erklären. Ich wählte als Beispiel das Segmentschnittproblem. Anschließend sollte ich erklären, wie das Problem mit Divide and Conquer gelöst werden kann.

Soweit die Zeit es zuließ, ging es hier bis in einige Details der Datenstrukturen und Algorithmen. (Was bedeutet beispielsweise der folgende Satz: Plane-Sweep leistet eine Reduktion eines zweidimensionalen Mengenproblems auf ein eindimensionales dynamisches Suchproblem?)

Schwerpunkte: Ich habe es so erlebt, daß Professor Güting den Schwerpunkt darauf legt, ob man das Thema verstanden bzw. sich aktiv damit auseinandergesetzt hat. Ich hatte den Eindruck, daß dies Vorrang hat vor der absoluten Korrektheit im sprachlichen Ausdruck und auch eine mehr informelle Erklärungsweise OK ist. Auch wenn sich - wie bei mir sogar mehrfach - herausstellt, daß man ein Detail falsch verstanden oder vergessen bzw. mal zu kurz gedacht hat, ist das offensichtlich kein Problem.

Prüfungsatmosphäre: Ich kann mich den Urteilen meiner VorgängerInnen nur anschließen: Professor Güting ist ein ausgesprochen angenehmer und ruhiger Prüfer, der bemüht ist, eine möglichst entspannte Prüfungsatmosphäre zu schaffen. Darüber hinaus hatte ich den Eindruck, daß er an der Mitgestaltung der Prüfung durch den Prüfling ein echtes

Interesse hat. Er ließ mir einigen Spielraum, die Prüfung mitzugestalten, indem er zunächst einmal eher allgemein fragte. Ich kann Professor Güting als Prüfer uneingeschränkt empfehlen.

Kurs: Datenstrukturen 1663

Prüfer: Prof. Dr. Güting
Note: 1,0

Fragen:

1. Erläutern Sie die Begriffe "Datentyp", "abstrakter Datentyp" und "Datenstruktur" (wichtig: Unterschied Syntax, Semantik)
2. Welches sind die Dictionary-Operationen?
3. Nun gibt es einfache und komplizierte Dictionary-Implementationen. Welche einfache kennen Sie denn?
   (Habe Bitvektor angegeben und auf Vorteile und Nachteile aufmerksam gemacht. Als Nachteil habe ich auch erwähnt, daß es schwierig sein, Begriffe zu speichern, weil man eine Art Schlüssel benötige. Darüber kamen wir zum Hashing)
4. Was ist denn Hashing?
   (Habe angefangen mit der Definition, alles zu Hashfunktionen, offenem und geschlossenem Hashing und Kollisionsstrategien usw. erzählt, ohne, daß Prof Güting eine Zwischenfrage gestellt hat. Nach einiger Zeit hat er mich dann unterbrochen, um weiterzugehen.)
5. Wie kann man denn sonst noch die dictionary-Operationen implementieren?
   (Habe AVL-Baum und die Definition erwähnt und bin. Suchbaum in diesem Zusammenhang erklärt. Danach bin ich auf Rebalancierung eingegangen (ohne, daß Herr Güting danach gefragt hätte) und habe ein Bespiel für eine Rotation aufgezeichnet.)
6. Wie viele Rotationen sind denn nötig, um einen Baum wieder zu balancieren?
   (insert: eine Rotation oder Doppelrotation, delete: evtl. alle Knoten bis zur Wurzel rebalancieren)
7. Wie ist denn sichergestellt, daß der Baum nicht entartet?
   (Habe zuerst Strukturinvariante erwähnt und gesagt, daß insert und delete mit anschließender Rebalancierung diese Strukturinvariante aufrechterhalten.)
8. Und die Höhe eines AVL-Baumes?
   (Habe die Formel N(h)=N(h-1)+N(h-2)+1 erklärt und angefangen, diese zu beweisen)

<u>Fazit:</u> Aus meiner Sicht ist es wichtig, umfassend und genau auf die Fragen zu antworten und das angesprochene Thema selbstständig auszuweiten und Zusammenhänge aufzuzeigen. Z.B. habe ich zu Frage 4 alles über Hashing erzählt, was ich wusste. Nach einer Weile hat mich Prof. Güting dann unterbrochen und ist auf ein anderes Thema zu sprechen gekommen.

Prüfungsprotokoll Diplomvorprüfung
Praktische Informatik

Prüfungsinhalt:	**1663 Datenstrukturen (SS 95)**
Prüfer:	**Prof. Dr. Güting**
Beisitzer:	**Dr. Schneider**
Zeit/Ort:	**27. November 1996 - 09:30 - LG PI IV**
Dauer:	**ca. 25 min**
Prüfling:	**Thomas Salvador**
Note:	**1.0**

Fragen

- **Erläuterung der grundlegenden Begriffe im Zusammenhang mit Datenstrukturen: Algebra, abstrakter Datentyp, Datenstruktur, ...**
- Abschnitt 1.2
- **Anwendung: Verwaltung einer großen Menge von Mitarbeitern und -daten**
- i.w. insert, delete und member
- externes Dictionary
- B-Baum
- diesen definieren (ist Spezialfall eines Vielweg-Baumes, vgl. 7.1)
- insbesondere Höhe eines B-Baumes ($O(\log_{(m+1)} n)$)
- **Anwendung: Temporale Datenbank, Beispiel: Mitarbeiter, die irgendwann einmal beschäftigt waren. Unterstützt werden soll die Anfrage, welche Mitarbeiter zu einem gegebenen Zeitpunkt eingestellt waren.**
- entspricht dem Problem, Intervalle zu verwalten und die zu liefern, die eine bestimmte Query-Koordinate enthalten
- Segment-Baum
- diesen definieren, Einfügen (kanonische Knotenbedeckung), Löschen, Suchen, Komplexitäten (Platz, Zeit)

Eindruck

Die Prüfung fand im Dienstzimmer von Prof. Dr. Güting statt. Prof. Dr. Güting ist sehr freundlich und versteht es die (jedenfalls bei mir vorhandene) Nervosität zu nehmen. Prof. Dr. Güting formuliert die Fragen ausführlich und sehr verständlich, läßt einem Zeit zum Denken und Reden, bei unvollständigen Antworten fragt Prof. Dr. Güting nach.

Prof. Dr. Güting ist als Prüfer unbedingt zu empfehlen.

Protokoll der mündlichen Prüfung

Kurs: Datenstrukturen 1663

Prüfer: Prof. Dr. Güting

Note: 1,0

Fragen:

1. Erläutern Sie die Begriffe "Datentyp", "abstrakter Datentyp" und "Datenstruktur" (wichtig: Unterschied Syntax, Semantik)

2. Welches sind die Dictionary-Operationen?

3. Nun gibt es einfache und komplizierte Dictionary-Implementationen. Welche einfache kennen Sie denn?
(Habe Bitvektor angegeben und auf Vorteile und Nachteile aufmerksam gemacht. Als Nachteil habe ich auch erwähnt, daß es schwierig sein, Begriffe zu speichern, weil man eine Art Schlüssel benötige. Darüber kamen wir zum Hashing)

4. Was ist denn Hashing?
(Habe angefangen mit der Definition, alles zu Hashfunktionen, offenem und geschlossenem Hashing und Kollisionsstrategien usw. erzählt, ohne, daß Prof Güting eine Zwischenfrage gestellt hat. Nach einiger Zeit hat er mich dann unterbrochen, um weiterzugehen.)

5. Wie kann man denn sonst noch die dictionary-Operationen implementieren?
(Habe AVL-Baum und die Definition erwähnt und bin. Suchbaum in diesem Zusammenhang erklärt. Danach bin ich auf Rebalancierung eingegangen (ohne, daß Herr Güting danach gefragt hätte) und habe ein Bespiel für eine Rotation aufgezeichnet.)

6. Wie viele Rotationen sind denn nötig, um einen Baum wieder zu balancieren?
(insert: eine Rotation oder Doppelrotation, delete: evtl. alle Knoten bis zur Wurzel rebalancieren)

7. Wie ist denn sichergestellt, daß der Baum nicht entartet?
(Habe zuerst Strukturinvariante erwähnt und gesagt, daß insert und delete mit anschließender Rebalancierung diese Strukturinvariante aufrechterhalten.)

8. Und die Höhe eines AVL-Baumes?
(Habe die Formel N(h)=N(h-1)+N(h-2)+1 erklärt und angefangen, diese zu beweisen)

<u>Fazit:</u> Aus meiner Sicht ist es wichtig, umfassend und genau auf die Fragen zu antworten und das angesprochene Thema selbstständig auszuweiten und Zusammenhänge aufzuzeigen. Z.B. habe ich zu Frage 4 alles über Hashing erzählt, was ich wusste. Nach einer Weile hat mich Prof. Güting dann unterbrochen und ist auf ein anderes Thema zu sprechen gekommen.

Prüfungsprotokoll zur Prüfung Praktische Informatik
Kurs : 01663 Datenstrukturen
Datum: 09.12.98 11Uhr
Prüfer: Prof. Dr. Güting
Note: 1,3

1. Allgemeine Begriffe: Datenstruktur, Algebra, ADT, Datenstruktur. In diesem Zusammenhang wurden dann eben auch Semantik und Signatur abgefragt.
Algebra: Definition des Datentyps mit Hilfe mathematischer Funktionen.
ADT: Definition mit Hilfe von Gesetzen; Axiomen und Eigenschaften der Operationen.
Datenstruktur: Menge von Objekttypen und Operationen auf diese. Hier kam dann die Frage:
Was sind diese Objekttypen? atomare Strukturen oder andere Datenstrukturen, die bereits definiert sind oder noch zu definieren sind.
Gemeinsamkeit zwischen ADT und Algebra : Signatur. Was wird da definiert: Die benutzen Objekttypen sowie die Operationen mit Input und Output. Was steht in der Semantik : bei der Algebra werden den Objekttypen Trägermengen zugeordnet und den Operationen werden mathematische Funktionen zugeordnet, beim ADT werden die Operationen axiomatische beschrieben. Und was ist nun die Datenstruktur: die Implementierung des Datentyps auf algorithmischer Ebene. Was bedeutet das? Den Objekttypen werden Repräsentationen zugeordnet und die Operationen werden durch Algorithmen ersetzt.

2. Graphen : Was sind Graphen? Definition! Grad eines Graphen, Eingangsgrad und Ausgangsgrad.
Wie kann man diese darstellen? Adjazenzmatrix und Adjazenzliste. Platzbedarf $O(n^2)$ bzw. O(n+e>.
Welche Durchlaufarten gibt es für Graphen : Breitendurchlauf und Tiefendurchlauf. Wie funktionieren diese? Wie kann man diese realisieren : Breitendurchlauf mit Hilfe einer Queue, Tiefendurchlauf mit Hilfe eines rekursiven Algorithmus , Abbruchkriterium ein schon besuchter Knoten.

3. B-Bäume: Definition Vielweg-Suchbaum und B-Baum. Was passiert beim Einfügen? Kann zu einem Overflow führen. Was tut man dann: Splitten. Was kann beim Löschen passieren: Underflow. Was nu : Merge oder Rebalancieren. Warum sind B-Bäume zum Suchen geeignet: Pfadlänge zu allen Blättern gleich O(log<m+1)n). Wie kommt man auf diese Länge? Minimal gefüllter Baum hat m Schlüssel in jedem Knoten und m+1 Nachfolger. Mit der Ungleichung N(h) <= n < N(h+1>.

4. Praktisches Problem : Was passiert beim Löschen eines Schlüsselelementes : da sagte ich nur: den linken Schlüssel aus dem Nachfolgerknoten nehmen und als neuen Schlüssel definieren. Das ist aber nicht immer ganz richtig : es muß heißen : der nächstgrößte Schlüssel. Dieser liegt dann wohl im Blatt. Diesen löscht man aus dem Blatt und schreibt ihn als neuen Schlüssel. Dann kann man die restlichen Operationen im Blatt machen.

<u>Kommentar</u>:
Also die Prüfung war soweit okay. Prof. Dr. Güting ist von Natur aus sehr ruhig, läßt sich Zeit, um die Fragen zu stellen und läßt dem Prüfling auch Zeit, sich die Antwort zu überlegen. Ich mußte zwischendurch mal sagen : ,,OH, wie war das noch mal".

Da hat er mich überlegen lassen und als ich dann immer noch nicht auf die Antwort kam, hat er mir ein bißchen nachgeholfen. Zur Bewertung sagte er, die Prüfung hätte im gut gefallen, in den Details happerte es bei mit wohl ein bißchen, aber war in Ordnung. Daher auch eine 1,3. Hätte ich persönlich nicht mehr mit gerechnet.

Kurs: 1663 Datenstrullturen (SS 1997)
Prüfer: Prof Dr. Güting
Datum: 28.01.1998
Dauer: ca. 20 Minuten
Note: 1,0

- Nennen Sie verschiedene Kriterien zur Klassifizierung von Sortieralgorithmen.

(intern/extern, methodisch, nach Effizienz, im Array oder nicht, allgemene/eingeschränkte Verfahren; Kapitel 6, S.179/180)

- Welche verschiedenen Sortierverfahren kennen Sie?

(einfaches Sortieren durch direktes Auswählen, Divide-and-Conquer-Methoden (Mergesort und Quicksort), verfeinertes Sortieren durch Auswählen (Heapsort), Fachverteilen (Bucketsort und Radixsort); Kapitel 6)
Wie fünktioniert das einfache Sortieren durch Auswählen?
(zwei Folgen Sorted und Unsorted, entnehme Unsorted jeweils in jedem Schritt das Minimum und hänge es an Sorted an; Kapitel 6, S.180)

- Welche Laufzeit hat dieses Sortierverfahren und warum?

($O(n^2)$, beim ersten Durchlauf der Schleife werden n Operationen durchgeführt, beim zweiten Durchlauf n-1 usw.; Kapitel 6, S.181)

- Was können Sie zum Heapsort sagen?

(Implementierung als partiell geordneter Baum im Array, Aufbau des Heap mit Hilfe eines Teilheap durch ,,einsinken lassen", sortierte Folge rückwärts durch Vertauschen am Ende des Array aufbauen, das nach vorne vertauschte Element wieder korrekt ,,einsinken lassen"; Kapitel 6, S.193)

- Welche Laufzeit hat Heapsort und warum?

(O(n logn), 0(n) für Aufbau des Heap, 2 n log n für die zwei Vergleiche von reheap auf jeder Ebene des Heap-Baumes; Kapitel 6, S.198)

- Gibt es schnellere Sortierverfahren?

(keine allgemeinen, Ω(n log n); Kapitel 6, S.198)

- Wie kommt es zu dieser unteren Schranke für die Komplexität von Sortierverfahren?

Entscheidungsbaum mit n! Blättern und insgesamt 2 n! - 1 Knoten, einsetzen in die Formel für die minimale Höhe eines binären Baumes, es folgt h = log n!, abschätzen von log n!; Kapitel 6, S.198)

- Was ist ein Graph?

und $E \subseteq VxV$; Kapitel 5, S.146

- Was sind ein Pfad und ein Zyklus?

(Folge von Knoten $v_1,...,v\sim$ mit $(v,v\sim_1) \sim E$, Zyklus wenn $v_1 = v\sim$; Kapitel 5, 5.146)

- Wie implementiert man Graphen, mit welchen Aufwänden?

(Adjazezmatrix mit Laufzeit O(1) für das Auffinden einer Kante und Platzbedarf

O(n2)

Datum: 22.10.1997
Prüfer: Prof. Dr. Güting
Dauer: ca. 15 min

- Begriffe erklären: Algebra, abstrakter Datentyp, Datenstruktur
- Binärer Suchbaum
  Definition
  Einsatz (Implementierung für eine Menge mit INSERT, DELETE MEMBER)
  Höhe im günstigsten und ungünstigsten Fall
- Vielweg-Suchbäume, allgemeine Definition
- B-Bäume der Ordnung m, Definition
- Aufbau eines B-Baumes der Ordnung 2 mit vorgegebenen Zahlen (overflow- und underflow-Behandlungen erklären)
  Dies war wohl eine alte Klausuraufgabe
- Frage: Was gibt es denn sonst noch so für Bäume? (Hinweis: algorithmische Geometrie) Segment-, Intervall- und Rangebaum
- Segmentbaum
  Grundstruktur erklären
  Intervall einfügen
  Platzbedarf angeben
  Was wird in der Knotenliste als Segmentnamen abgespeichert? (z.B. Indizes, falls Segmente in einem Array abgelegt sind)

Gesamteindruck:
Die Prüfung verlief in sehr ruhiger Atmosphäre. Herr Prof. Güting war sehr freundlich. Ich hatte genug Zeit zum Überlegen. Bei Frage 6 fiel mir zunächst keine Antwort, ein und ich habe um einen Tip gebeten, den mir Prof. Güting auch gab. Der Note hat es keinen Abbruch getan. Als Prüfer ist Prof. Güting sehr zu empfehlen.

Prüfer: Prof Dr. Güting
Datum: Oktober 1997
Dauer: ca. 20 Minuten

Themen:
1) Qualität eines Algoritmus
2) Laufzeiteigenschafien
3) Definition 0-Notation
4) Komplexität des Sortierproblems
5) Heap-Sort
- ausführliche Beschreibung mit Laufzeit
- partiell geordneter Baum definieren und erklären
6) Untere Schranke für Sortierverfahren
- Entscheidungsbaum erklären
- Herleitung von log(n!) = O(n log n)

Bemerkung:

Prof Güting ist ein angenehmer und ruhiger Prüfer. Anfangs wurden die Grundlagen

des Kurses abgefragt, d.h. Definitionen und die Terminologie aus KE 1 sollten verstanden und in eigenen Worten erklärt werden können. Im weiteren Verlauf ging er nur noch auf das Sortierproblem ein, dies aber sehr ausführlich und speziell. Ich war schon etwas überrascht, da er wirklich nur einen kleinen Teil des Stoffes abgefragt hat, diesen jedoch ausführlich und speziell.

Termin: Oktober 1997
Prüfer: Prof Dr. Güting
Dauer: ca. 20 min
Note: 2,0
Themen:
1. Nach welchen Merkmalen wird die Qualität eines Algorithmus beurteilt? Korrektheit, Laufzeit, Speicherplatzbedarf, etc.
2. Wie wird die Laufzeit eines Algorithmus gemessen? (die verschiedenen Stufen erklären: Elementaroperationen, Äquivalenzklassen, etc).
3. Definition der 0-Notation, schriftlich Die Definition mit eigenen Worten erklären.
4. Was sind DAC-Algorithmen? (Divide, Conquer, Merge)
5. Beispiele für DAC-Algorithmen. (Hier kurz erwähnt, daß diese auch für die Lösung von geometrischen Problemen verwendet werden, Sortieralgorithmen: Mergesort, Quick sort).
6. Mergesort und Quicksort genau erklären. (Was muß im Mergeschritt bei Quicksort gemacht werden? Nichts, falls in situ sortiert wird.)
7. Wie wird die Laufzeit von Mergesort bestimmt? Die Formel $T(n) = a + 2\ T(n/2) +...$ wurde sehr ausführlich gefragt).

Prüfungsathmosphäre: Herr Prof. Güting ist sehr ruhig, so daß meine Nervosität verging. Bei Sachen, die ich nicht wußte, versuchte er durch Umformulierungen oder Teilfragen mir auf die Lösung zu helfen. Allerdings tat er mir nicht den Gefallen, so ein Thema dann zu verlassen. Da ich kaum Zeit hatte mich auf die Prufüng vorzubereiten, hatte ich natürlich einige Lücken. Mit der Laufzeitberechnung von Mergesort hat er eine solche entdeckt. Mit der Benotung bin ich sehr zufrieden.

Viel Glück

Prüfer: Prof. Güting
Beisitzer: Dr. Erwig
Termin: 13.06.97
Dauer: 25 min.
Note: 1,3

Was ist ein Abstrakter Datentyp?
Was ist eine Algebra?
Was ist eine Signatur?
Wie definiert man einen Datentyp?
Was ist eine Datenstruktur?
    Siehe Grundlagen aus Kapitel 1.
Welche Funktionen sind auf dem Datentyp Dictionary ausführbar?
    Einfügen, Entfernen, Test auf Enthaltensein
Welche Implementierungen eines Dictionary-Datentyps kennen Sie?
    einfache Implementierungen, Hashing, binäre Suchbäume (schlecht, wenn entartet), AVL-Bäume

Was ist Hashing?
Schlüsseltransformationsverfahren, aus Wert des Elements seine Adresse im Speicher berechnen
Behälter zur Aufnahme der Elemente
Wie kann man das realisieren?
Hashtabelle als Array + Zeiger auf Listen
Welche Arten des Hashing gibt es, was ist der Unterschied?
offenes Hashing, Buckets beliebig erweiterbar
geschlossenes Hashing, Zahl der Einträge begrenzt; schlecht geeignet für dynamische Datenstrukturen; gelöschte Elemente dürfen nicht einfach entfernt werden. sendern müssen besonders gekennzeichnet werden (Auffinden von Elementen. die bei Kollision vorbeigeleitet worden sind)
Wie ist das Laufzeitverhalten bei offenem Hashing?
insert 0(1), Einfügen am Anfang einer Liste
delete, member ,,mit Glück 0(1); im worst-case 0(n)

Seite 1
11. April 97

Prüfer: Prof. Dr. Güting
Prüfungsinhalt: Kurs 01663 - Datenstrakturen
Datum: 11.April1997, 11.00 Uhr
Dauer: 20 Minuten
Note: 1,0
Prüfungsverlauf:

- Datenstrukturen - ein besonders beliebtes Gebiet sind Sortieralgorithmen. Wie kann man denn Sortieralgorithmen klassifizieren?

- Interne/ externe Verfahren (bei sehr großen Datenbeständen ist Band- oder Plattenspeicher nötig)
- Nach Methode einfügen, Auswählen, DAC, Spezielle Verfahren mit Lautzeit 0(n))
- Im Array oder nicht, speziell in siiu (Vorteil in siiu: bei 2 Arrays nur halbe Anzahl von Sätzen im Speicher)

Dabei hatte ich das Wichtigste fast vergessen:
Herr Güting sprach noch die Komplexität an; Ich erwähnte dazu die zwei Komplexitätsklassen $O(n^2)$ und O(n log n).

- Zu den Divide-and-Conquer-Algorithmen. Wie lautet das DAC-Paradigma?

Das sollte ich anhand eines fiktiven Algorithmus DAC(S) aufschreiben:
Terminierender Zweig, falls ISI=1; sonst divide, conquer, merge (allgemein erklärt).
Um den Divide- bzw. Mergeschritt genauer erklären zu können, führte ich zusätzlich Merge-Sort als Beispiel an:
Divide: Array in der Mitte teilen = 0(1)
Merge: Paralleler Durchlauf durch zwei geordnete Teilarrays = 0(n)
Schließlich nannte ich den Satz über balancierte DAC-Algorithmen, nach dem Merge-Sort die Komplexität 0(n log n) hat (wegen merge in 0(n) Zeit).

Wie zeigt man das genau? Beweisen Sie diesen Satz.

Zuerst schrieb ich die Rekursionsformel für die Laufzeit allgemeiner, balancierter DAC-Algorithmen hin: $T(n) = \begin{cases} afallsn = 1 \\ 2 \cdot T(n/2) + f(n) + cfallsn > 1 \end{cases}$

Dann führte ich den Beweis wie im Kurstext mit Hilfe des Baumes der rekursiven Aufrufe. Alleiniges Erwähnen genügte nicht; ich sollte den kompletten Beweis führen. An einer Stelle wurde es etwas hakelig, aber ich konnte mich korrigieren. Am Rande kam auch noch die minimale Höhe vollständiger Binärbäume zur Sprache:
h =| log(n +1)**|** -1

- Nun ein praktisches Problem. Sie sind Programmierer und müssen ein Fenstersystem entwerfen. Ein Mausklick auf den Bildschirm soll nun ein bestimmtes Fenster auswählen.
Welche Datenstruktur bietet sich für ein solches Problem an?

Es handelt sich um ein 2-dimensionales Suchproblem.
Ich nannte den Segment-Intervall-Baum (SI-Baum) und beschrieb kurz seinen Aufbau:
Der SI-Baum speichert Rechtecke und liefert genau die gesuchte Operation „ finden aller Rechtecke, die einen Query-Punkt enthalten“. Speichern der x-Intervalle in einem Segment-Baum. Dies selektiert je 0(log N) Knoten, deren x-Teilintervall jeweils komplett überdeckt ist; Es sind also nur noch die y-Intervalle interessant. Für jeden Knoten wird die Liste der eingetragenen y-Intervalle dann als Intervall-Baum organisiert. Eine Darstellung der Rechtecke in einem Segrnent-Segment-Baum ist auch denkbar; Sie kostet aber mehr Speicherplatz als der SI-Baum.

- Beschreiben Sie einen der genannten Bäume genauer, sagen wir den Segment-Baum.
ch habe ein Beispiel gezeichnet und erklärt:
Ein Seginent-Baum ist eine semidynamische Datenstruktur zur Speicherung einer Menge von Intervallen über einem festen Raster X = $\{x_0,...,x_n\}$. In den Blättern sind die atomaren Intervalle $[x_0,x_1]$, $[x_1,x_2]$,...,$[x_{n-1},x_n]$ gespeichert.
Die inneren Knoten enthalten die Konkatenation der Intervalle ihrer Söhne.
Der Segment-Baum ist also ein vollständiger Baum der Höhe 0(log N).
Einfügen eines Intervalls:
Intervallname so hoch wie möglich im Baum an die betroffenen Knotenlisten anfügen. Wann immer zwei Söhne eines Vaters markiert würden, markiere statt dessen den Vater. Die Menge CN(i) der covered nodes habe ich nur erwähnt – nicht explizit hingeschrieben.
Suchen aller Intervalle, die eine Query-Koordinate enthalten:
Von der Wurzel aus läuft man entlang des Pfades zu dieser Ouery-Koordinate und gibt die Knotenlisten aller besuchten Knoten aus.

Fazit
Eine sehr angenehme Prüfung mit einem netten Prüfer. Die Atmosphäre war recht locker.
Kleinere Stockungen im Beweis wurden nicht negativ bewertet. Wenn man sich nach hilfreichen Zusatzfragen selbst korrigieren kann, ist das schon in Ordnung.
Es wurden nur zwei Themenbereiche des Kurses behandelt, diese aber einigerma-

ßen ausführlich. Benutzt man in seinen Erklärungen besondere Fachbegriffe, z.B. ,,semidynamische Datenstruktur" oder ,,in situ", so sollte man diese selbstverständlich auch erklären können da durchaus spontane Zwischenfragen folgen können. Gelingt einem dies, so kann man Pluspunkte sammeln wie ich an der Mimik von Prof Güting zu erkennen glaubte.
Mit der Benotung war ich natürlich sehr zufrieden. Trotz des kleinen Aussetzers im Beweis die 1,0 zu geben, empfand ich als überaus fair. Ich kann Herrn Prof. Güting als Prüfer uneingeschränkt empfehlen.