Gedächtnisprotokoll zur Diplomprüfung THEORETISCHE INFORMATIK

Fächer: Effiziente Algorithmen (Hromkovic, SoSe 2002)
Applied Automata Theory (Thomas, SoSe 2002)
Compilerbau (Indermark, WS 2002/03)
Prüfer: Hromkovic (Effiziente Algorithmen, Applied Automata Theory), Indermark (Compilerbau)
Datum: 11.4.2003
Note: 1.0

Effiziente Algorithmen

- Dynamische Programmierung:
  - Idee beschreiben.
  - Gemeinsamkeiten und Unterschiede zu Divide-And-Conquer benennen.
  - Floyd-Algorithmus erklären.
- NP-Vollständigkeit:
  - Alle Reduktion nennen, die ich kenne und auch beweisen kann.
  - Satz von Cook:
    * Idee beschreiben.
    * Wie beschreibt man eine Konfiguration der NTM?
      $\rightarrow C(i, j, t), H(i, t), S(i, t)$ angegeben.
    * Die Formel $U(x_1, ..., x_n)$ aufschreiben.
    * Bedeutung der Komponenten von $\Phi = A \wedge B \wedge C \wedge D \wedge E \wedge F \wedge G$ angeben und die Formel D hinschreiben.
- PTAS für SKP:
  - Algorithmus aufschreiben.
  - Approximationsgüte herleiten (ab ca. der Hälfte der Herleitung war's Herrn Hromkovic genug).

Applied Automata Theory

- Minimierung von DFAs:
  - Idee beschreiben und erklären, wie der Kanonische DFA aufgebaut ist.
  - Nerode Kongruenz.
  - Was sind trennbare Zustände?
  - Minimierungsalgorithmus erklären.
  - Warum arbeitet der Algorithmus korrekt und bricht auch wirklich ab?

- Baumautomaten:
  - Motivation für Baumautomaten
    $\rightarrow$ Beispiel angeben.
  - Welche zwei Ansätze gibt es?
    $\rightarrow$ Bottom-Up und Top-Down.
  - BU-Aufbau erklären.
  - Erkennen $\uparrow$ DFAs und $\uparrow$ NFAs die gleichen Sprachen?
    $\rightarrow$ Ja (Potenzmengenkonstruktion).

Compilerbau

- Reguläre Ausdrücke:
  - Angenommen man erweitert die regulären Ausdrücke um die Negation, wie kann dann $\neg\alpha$ erkannt werden?
    $\rightarrow$ α über NFA-Methode $(!)$ erst in NFA und dann in DFA überführen. Beim DFA werden dann Endzustandsmenge und Nicht-Endzustandsmenge vertauscht.
  - Komplexität der NFA- und DFA-Methode.
- Erweitertes Matching-Problem:
  - Definieren.
  - Gibt es mehrere lm-Zerlegungen?
  - flm-Zerlegung:
    * Skizzieren.
    * Backtrack-Automat beschreiben.
    * Wie kann man bei minimiertem Automaten schnell testen, ob ein Zustand produktiv ist?
      $\rightarrow$ Es gibt nur eine Senke im minimalen DFA. Alle anderen Zustände sind produktiv.
- BU-Parser:
  - Wie kann man einen LR(0)-Parser bauen?
    $\rightarrow$ LR(0)-Mengen berechnen oder Potenzmengenkonstruktion.
  - Potenzmengenkonstruktion:
    * Zustandsmenge definieren.
    * Anfangszustand angeben.
    * Am Beispiel der Grammatik $S' \rightarrow S, S \rightarrow AB, A \rightarrow a, B \rightarrow b$ zeigen, welche Transitionen möglich sind.
    * Was entspricht einem Zustand des DFAs?
      $\rightarrow$ Genau eine LR(0)-Menge.
    * Was stellt die δ-Funktion dar?
      $\rightarrow$ goto-Funktion.

- Semantische Analyse:
  - Am Beispiel von $A \rightarrow BCD$ (mit je einem inheriten und synthetischen Attribut) grafisch erklären, welche Abhängigkeiten möglich sind und welche nicht.
  - Wie sieht das bei LAGs aus?
  - Anhand der Zeichnung beschreiben, was Zirkularitäten sind und wie sie entstehen können.
  - Wie sieht die Attributberechnung bei LAGs aus?
    $\rightarrow$ Tiefensuche mit zwei Knotenbesuchen (erst inherite Attribute, dann synthetische).

Fazit: Beide Prüfer sind sehr zu empfehlen. Ich war gerade am Anfang sehr nervös und hatte das Gefühl, daß es bei der Probeprüfung zwei Wochen vorher (siehe nächste Seite) besser lief. Man sollte sich nicht zu sehr irritieren lassen, wenn man bei einigen Sachen nicht sofort auf die richtige Antwort kommt. Beide Prüfer helfen einem dann auch ganz gut weiter oder formulieren die Frage nochmal anders. In Effiziente Algorithmen ist es wichtig, die Algorithmen inkl. Herleitung von Laufzeit, Approximationsgüte etc. zu kennen. Außerdem sollte man die Reduktionen drauf haben. Applied Automata Theory wurde von den drei Fächern sicherlich am oberflächlichsten behandelt. Herr Indermark legt sehr großen Wert darauf, daß man die Zusammenhänge seiner Vorlesung wirklich verstanden hat. Die wesentlichen Definitionen sollte man schon auch kennen, aber letztendlich kommt es auf das Gesamtverständnis an. Bei seiner ersten Frage habe ich mich erstmal etwas schwer getan, da sie ja keine reine Reproduktion der Vorlesung darstellt. Ich habe ihm erzählt, daß man aus einem DFA leicht einen DFA für die Negation bauen kann, daß das aber bei NFAs nicht funktioniert. Also hatte ich erstmal keine Idee wie ich (nach Thompson) einen NFA für $\neg\alpha$ bauen soll. Irgendwie kam ich dann aber drauf, nachdem Herr Indermark auch so einige Male mit dem Zaunpfahl gewunken hatte.

Gedächtnisprotokoll zur Testprüfung Effiziente Algorithmen (Hromkovic)

- Divide And Conquer:
  - Idee beschreiben.
  - Ein beliebiges Beispiel angeben: Habe das Multiplizieren großer Zahlen beschrieben und erzählt wie man von $O(n^2)$ auf $O(n^{1.59})$ kommt.
- NP-Vollständigkeit:
  - Reduktion VC $\leq_p$ gHK erklären (eher grafisch als formal). Den Beweis mußte ich nicht machen.
- Lokale Suche:
  - LSS(*Neigh*) beschreiben.
  - Wo funktioniert es gut?
    $\rightarrow$ MST.
  - Wo funktioniert es schlecht?
    $\rightarrow$ Pathologischen Problemfall von TSP komplett erklärt.
- Anwendbare exponentielle Algorithmen:
  - Idee für Erfüllbarkeitsprobleme beschrieben ($O(2^n) \rightarrow O(c^n)$ mit $c < 2$).
  - Konstruktion der Formel $F(l = 0$ bzw. $l = 1)$ für ein Literal l.
  - Divide-And-Conquer-Algorithmus angeben.
  - Rekurrenzgleichung aufschreiben und analysieren (Wo kommt die Komplexitätsminderung her?).