Lehrstuhl für Informatik I
— Informationsstrukturen und
wissensbasierte Systeme —
Universität Würzburg

Prof. Dr. H. Noltemeier

Würzburg, 15. Juli 1998

Übungen zur Vorlesung
Praktische Informatik II (SS98)

Musterlösung zum 6. Übungsblatt

Lösung zu Aufgabe 20: (Volle Persistenz)

Mittels der „fat node"-Methode sollte zur folgenden Sequenz von Operationen eine Versionenliste VL und eine (voll) persistente Datenstruktur gebildet werden (dabei ist die letzte Komponente jeweils der Zeitstempel der aufgerufenen Version).

1. Generierung eines Startknotens (Karosserie / grün / 0).
2. Ausführung der folgenden Operationen:
   (a) Insert(Sportgetriebe / 5-Gang, 1)
   (b) Insert(Klimaanlage / —, 2)
   (c) Insert(Sitzheizung / nur Fahrer, 3)
   (d) Insert(Fensterheber / mechanisch, 3)
   (e) Insert(Schiebedach / elektrisch, 1)
   (f) Update(Sitzheizung / beide Vordersitze, 4)
   (g) Update(Fensterheber / elektrisch, 6)
   (h) Insert(Alarmanlage / fernbedienbar, 7)
   (i) Delete(Sportgetriebe, 5)
   (j) Insert(Automatikgetriebe / 4-Gang, 10)
   (k) Access(Klimaanlage, 9)
   (l) Access(Sitzheizung, 7)
   (m) Delete(Klimaanlage, 5)

Wir erhalten folgendes Ergebnis:

- Knoten der Datenstruktur:
  (Ein neuer Knoten besitzt folgenden Aufbau:
  [Zeitstempel | Begriff B | Ausführungsoption A | Pointer P])

  v_1: [1 | Karosserie | grün | *nil*] ⟶ [2 | P | v_2] ⟶ [6 | P | v_6] ⟶
  [10 | P | v_3] ⟶ [4 | P | v_2]

  v_2: [2 | Sportgetriebe | 5-Gang | *nil*] ⟶ [3 | P | v_3] ⟶ [12 | P | v_5]
  ⟶ [10 | P | v_3]

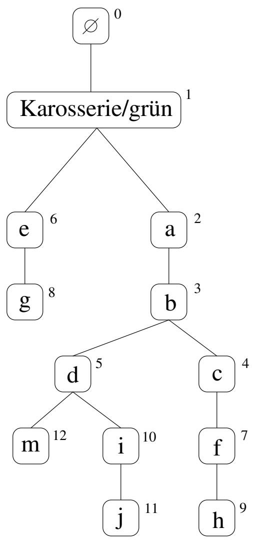
  v_3: [3 | Klimaanlage | — | *nil*] ⟶ [4 | P | v_4] ⟶ [5 | P | v_5]

v_4:	4	Sitzheizung	nur Fahrer	*nil*	$\longrightarrow$	7	A	beide Vordersitze	$\longrightarrow$	9	P	v_7

v_5:	5	Fensterheber	mechanisch	*nil*	$\longrightarrow$	11	P	v_8	$\longrightarrow$	4	P	*nil*

v_6:	6	Schiebedach	elektrisch	*nil*
v_7:	9	Alarmanlage	fernbedienbar	*nil*
v_8:	11	Automatikgetriebe	4-Gang	*nil*

- Versionenbaum:



- Access-Array:

Version Nr.	Wurzel	Version Nr.	Wurzel	Version Nr.	Wurzel
0	*nil*	5	v_1	10	v_1
1	v_1	6	v_1	11	v_1
2	v_1	7	v_1	12	v_1
3	v_1	8	v_1		
4	v_1	9	v_1		

- Versionenliste:

0, 1, 6, 8, 2, 3, 5, 12, 10, 11, 4, 7, 9

Zugriffe:

- ACCESS(Klimaanlage, 9)
  Suche endet erfolgreich bei v_3; „—“ wird zurückgeliefert.

- ACCESS(Sitzheizung, 7)
  Suche endet erfolgreich bei v_4;„beide Vordersitze“ wird zurückgeliefert.

Hinweis: Schritt (g) UPDATE(Fensterheber / elektrisch, 6) schlägt fehl, da in Version 6 kein Fensterheber vorhanden ist. Wir gehen dennoch davon aus, daß eine Version 8 in die Versionenliste aufgenommen wird, da andernfalls auch Schritt (j) fehlschlägt.

Lösung zu Aufgabe 21: (Optimale Serialisierung)

Gegeben seien Listen A_i $(i = 1, \ldots, L)$, die bereits alle monoton wachsend sortiert sind und zu einer sortierten Gesamtliste sortiert werden sollen. Die Anzahlen der Listenelemente seien jeweils $b_1, \ldots, b_L$. Das „Zusammenfassen" gehe jeweils paarweise vor, z.B. das Mischen von A_i und A_k zu einer sortierten Liste A_{ik} kann in $b_i + b_k$ elementaren Operationen erfolgen. Die Reihenfolge der Misch-Operationen kann folglich durch einen binären Wurzelbaum repräsentiert werden, dessen Blätter genau die Listen $A_1, \ldots, A_L$ repräsentieren und dessen Nicht-Blätter jeweils zwei Söhne aufweisen.

Wir wollen im folgenden betrachten, in welcher Reihenfolge die Paare von Listen A_i, A_k gewählt werden sollen, so daß die Gesamtzahl aller elementaren Misch-Operationen so klein wie möglich wird.

(a) **Zu zeigen:** Ist $T_{L,2}$ ein binärer Wurzelbaum der obigen Art mit L Blättern und sind die Blätterhöhen h_i $(i = 1, \ldots, L)$, so benötigt man insgesamt $\sum_{i=1}^{L} h_i \cdot b_i$ elementare Mischoperationen.

Beweis: Vollständige Induktion über die Höhe des binären Wurzelbaumes, der die Reihenfolge der Mischoperationen repräsentiert.

(IA) $h = 1$:
Die Wurzel hat als Nicht-Blatt einen linken und einen rechten Sohn, die ihrerseits keine Söhne mehr besitzen und die Listen A_1 und A_2 repräsentieren. Für die Ausführung dieses Mischplanes benötigt man

$$b_1 + b_2 = \sum_{i=1}^{2} 1 \cdot b_i = \sum_{i=1}^{L} h_i \cdot b_i$$

elementare Mischoperationen.

(IV) Sei die Behauptung gezeigt für ein $h \in \mathbb{N}$.

(IS) $h \to h + 1$:
Wir betrachten den letzten Schritt des Mischens der Listen. Dieser entspricht der Wurzel des Baumes.
Sei $I_l \subset \{1, \ldots, L\}$ die Menge der Listenindizes, deren zugehörigen Blätter sich im linken Teilbaum befinden, und sei $I_r = \{1, \ldots, L\} - I_l$ die Menge der Listenindizes, deren zugehörigen Blätter sich im rechten Teilbaum befinden.
Nach (IV) benötigen wir für das Mischen der Listen im linken Teilbaum

$$\sum_{i \in I_l} (h_i - 1) \cdot b_i = \sum_{i \in I_l} h_i \cdot b_i - \sum_{i \in I_l} b_i$$

elementare Operationen und für das Mischen der Listen im rechten Teilbaum

$$\sum_{i \in I_r} (h_i - 1) \cdot b_i = \sum_{i \in I_r} h_i \cdot b_i - \sum_{i \in I_r} b_i$$

elementare Operationen.

Im letzten Schritt sind zwei Listen der Längen $\sum_{i\in I_l} b_i$ und $\sum_{i\in I_r} b_i$ zu Mischen. Da dies in $\sum_{i\in I_l\cup I_r} b_i$ Schritten zu erledigen ist, werden insgesamt

$$\sum_{i\in I_l}(h_i-1)\cdot b_i+\sum_{i\in I_r}(h_i-1)\cdot b_i+\sum_{i\in I_l\cup I_r} b_i=\sum_{i=1}^{L} h_i\cdot b_i$$

elementare Operationen benötigt.

Damit ist die Behauptung gezeigt. □

(b) Ein optimaler „Misch-Plan“, repräsentiert durch ein $T_{opt}\in\mathbb{T}_{L,2}$ (Menge aller binären Bäume mit L Blättern) erfüllt mit $b:=(b_1,\ldots,b_L)$

$$\tilde{L}_{(T_{opt};b)}:=\sum_{i=1}^{L}\tilde{h}_i\cdot b_i=\min_{T\in\mathbb{T}_{L,2}}\sum_{i=1}^{L} h_i\cdot b_i$$

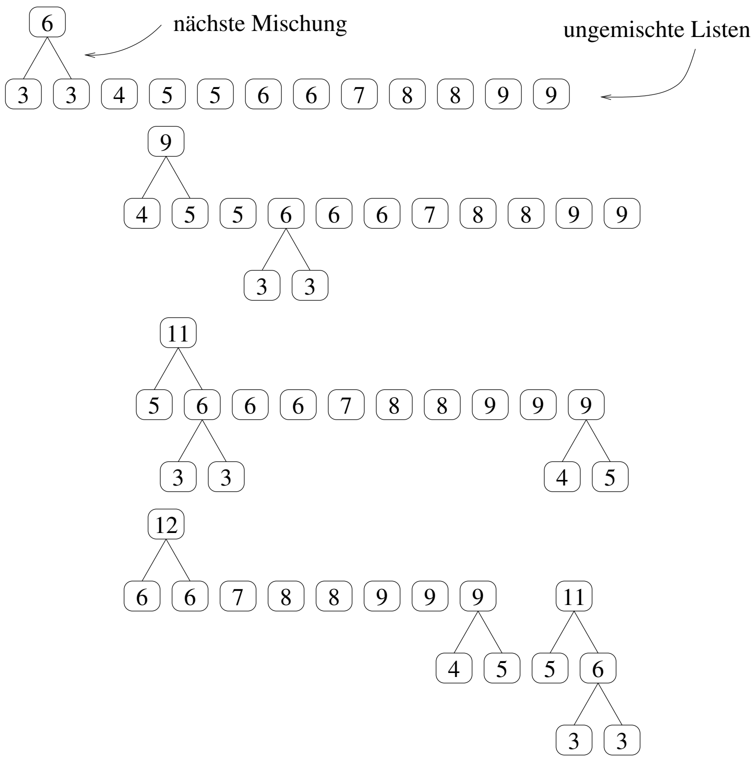
wobei $\tilde{h}_i$ die Höhe von A_i in T_{opt} ist; oder gleichwertig:

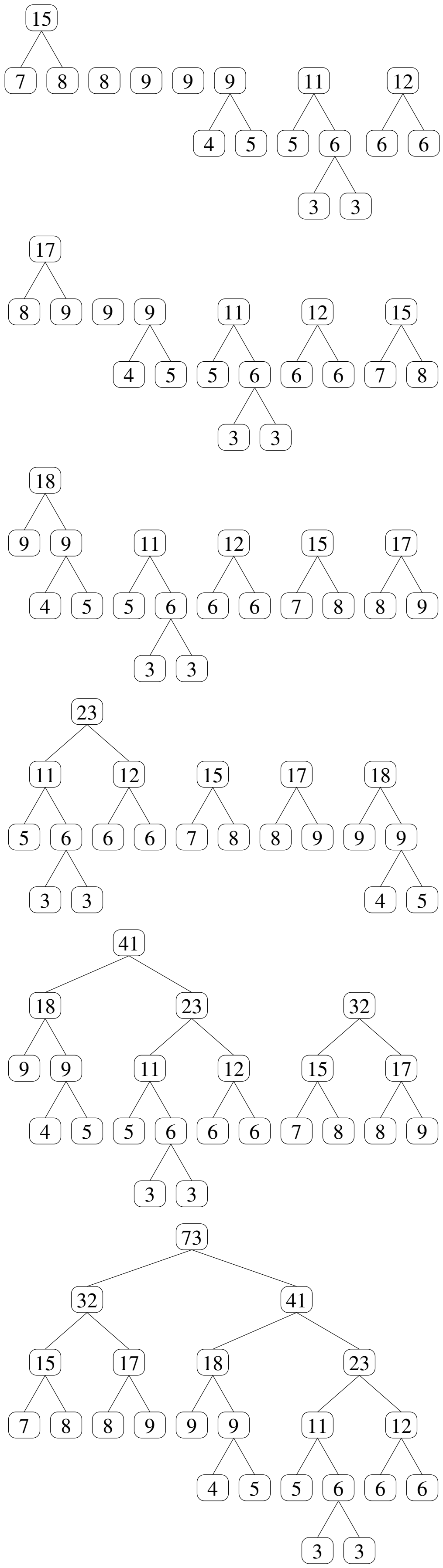
$$\tilde{L}_{(T_{opt};b)}/B=\min_{T\in\mathbb{T}_{L,2}}\sum_{i=1}^{L} h_i\cdot b_i/B$$

(=minimaler Erwartungswert der Blätterhöhe in $\mathbb{T}_{L,2}$ zur Wahrscheinlichkeitsverteilung $P:=(b_1/B,\ldots,b_L/B)$ auf den Blättern mit $B:=\sum_{i=1}^{L} b_i$)

Zu $L=12$ und $b=(5,3,7,9,8,4,6,8,9,5,3,6)$ war ein derartiger optimaler Misch-Plan anzugeben.

Die Konstruktion dieses Baumes erfolgt analog zur Konstruktion einer Huffman-Codierung. Kürzere Listen entsprechen dabei Symbolen geringerer Häufigkeit, d.h. es werden jeweils die beiden kürzesten noch zu mischenden Listen als nächstes betrachtet.





Für den so erhaltenen Baum T gilt $\tilde{L}_{(T;b)} = \tilde{L}_{(T_{opt};b)} = 257$.

Lösung zu Aufgabe 22: (Suchprozesse, Wurzelbäume, Codes)

(a) Sei eine Menge S von 15 Münzen gegeben, die von 1 bis 15 numeriert sind. Genau eine der Münzen ist defekt, d.h. schwerer als die anderen Münzen. Eine Wiegung entspricht der Wahl von 2 disjunkten Teilmengen A und B von $\{1, \ldots, 15\}$ mit $|A| = |B|$, wobei A und B die Menge der Münzen aus der linken und rechten Waagschale bezeichnen.

Behauptung: Man benötigt im schlechtesten Fall genau drei Wägungen, um die defekte (d.h. schwerere) Münze zu finden.

Beweis:

Pro Wägung (A, B) gibt es drei mögliche Ergebnisse:

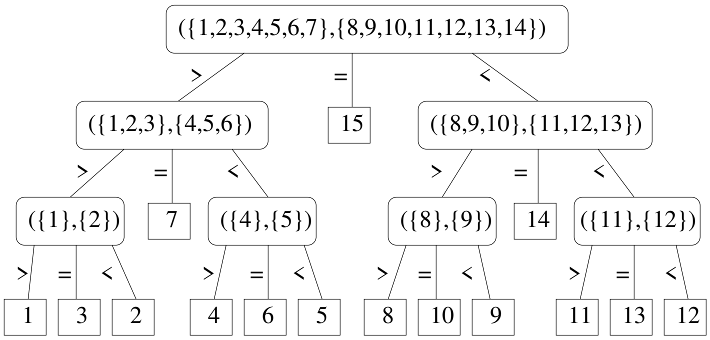
(i) Die Münzen in der Menge A sind zusammen schwerer als die in der Menge B.

(ii) Die Münzen in der Menge A sind zusammen genauso schwer wir die Münzen aus der Menge B.

(iii) Die Münzen in der Menge B sind zusammen schwerer als die in der Menge A.

Dem Problem liegt also ein ternärer Entscheidungsbaum zugrunde. Für jedes der 15 möglichen Ergebnisse muß ein Blatt existieren, da jede der 15 Münzen die gesuchte sein kann. Bei einem ternären Baum der Höhe 2 kann es jedoch maximal 9 Blätter geben. Folglich sind zwei Wägungen im allgemeinen nicht ausreichend.

Daß drei Wägungen ausreichen, zeigt folgender Entscheidungsbaum:



Hierbei symbolisiert ein innerer Knoten eine Wägung (A, B) und jeder seiner drei Teilbäume eines der möglichen Ergebnisse: Der linke Teilbaum steht für Ergebnis (i), der mittlere für Ergebnis (ii) und der rechte für Ergebnis (iii) (siehe oben). In den Blätter befindet sich dann jeweils die gesuchte Münze.

(b) Sei aus der Menge S von 15 Münzen wieder genau eine defekt, wobei defekt nun bedeutet, daß diese Münze schwerer oder leichter ist als die restlichen.

Behauptung: Man benötigt vier Wägungen zur Bestimmung der defekten Münze.

Beweis:

Angabe eines Algorithmus zur Bestimmung der defekten Münze mit maximal vier Wägungen (siehe Algorithmus 1).

Algorithmus 1 FINDEDEFEKTEMÜNZE

```
erg1 ← WIEGE({1, ... ,5},{6, ... ,10})
erg2 ← WIEGE({1, ... ,5},{11, ... ,15})
if erg1 = '<' and erg2 = '<' then
  return SUCHEKLEINER(1, ... ,5)
else if erg1 = '>' and erg2 = '=' then
  return SUCHEKLEINER(6, ... ,10)
else if erg1 = '=' and erg2 = '>' then
  return SUCHEKLEINER(11, ... ,15)
else if erg1 = '>' and erg2 = '>' then
  return SUCHEGRÖSSER(1, ... ,5)
else if erg1 = '<' and erg2 = '=' then
  return SUCHEGRÖSSER(6, ... ,10)
else if erg1 = '=' and erg2 = '<' then
  return SUCHEGRÖSSER(11, ... ,15)
```

Algorithmus 2 SUCHEKLEINER($i_1, \ldots, i_5$)

```
erg1 ← WIEGE({i1, i2},{i4, i5})
if erg1 = '<' then
  erg2 ← WIEGE({i1},{i2})
  if erg2 = '<' then
    return i1
  else
    return i2
else if erg1 = '=' then
  return i3
else
  erg2 ← WIEGE({i4},{i5})
  if erg2 = '<' then
    return i4
  else
    return i5
```

Der Algorithmus benötigt offenbar in keinem Fall mehr als vier Wägungen.

Es bleibt zu zeigen, daß die defekte Münze im allgemeinen nicht mit weniger als vier Wägungen bestimmt werden kann.

Da bei einer Wägung drei verschiedene Ergebnisse möglich sind, ergibt sich wie in Teilaufgabe (a) ein ternärer Entscheidungsbaum. Da wir 30 Möglichkeiten für die defekte Münze haben (jede Münze könnte leichter

Algorithmus 3 SUCHEGRÖSSER($i_1, \ldots, i_5$)

```
erg1 ← WIEGE({i1, i2},{i4, i5})
if erg1 = '<' then
   erg2 ← WIEGE({i4},{i5})
   if erg2 = '<' then
      return i5
   else
      return i4
else if erg1 = '=' then
   return i3
else
   erg2 ← WIEGE({i1},{i2})
   if erg2 = '<' then
      return i2
   else
      return i1
```

oder schwerer als normal sein), ein ternärer Baum der Höhe 3 jedoch höchstens 27 Blätter besitzt, kann es keinen Algorithmus geben, der das Problem im allgemeinen mit weniger als 4 Wägungen löst.

(c) Aus der Menge S von 15 Münzen sind nun 2 Münzen defekt (beide schwerer als die restlichen Münzen).

Behauptung: Man kann mit 5 Wägungen die 2 defekten Münzen ermitteln.

Beweis: Wir formulieren die Lösung als Suchprozess (siehe Algorithmus 4) und geben einen Entscheidungsbaum für den Algorithmus an.

Die in Algorithmus 4 auftretende Unterprozedur SUCHEZWEISCHWEREMÜNZEN läßt sich leicht realisierien und wird daher hier nicht gesondert angegeben.

Sei $S := \{m_1, \ldots, m_15\}$. $A \left\{ \begin{array}{c} < \\ = \\ > \end{array} \right\} B$ mit $A, B \subsetneq S$, $|A| = |B|$ bedeute $A \left\{ \begin{array}{c} \text{ist leichter als} \\ \text{ist gleichschwer wie} \\ \text{schwerer als} \end{array} \right\} B$.

Daß dieser Suchprozeß die gewünschte Menge defekter Münzen nach maximal 5 Wägungen liefert, läßt sich am zugehörigen Entscheidungsbaum (siehe Abbildung 1) ablesen. Hierbei bedeutet eine Verzweigung nach links „$A > B$“, in die Mitte „$A = B$“ und nach rechts „$A < B$“.

Aus Platzgründen nicht dargestellte Teilbäume ergeben sich analog zu den gezeigten und zu Teilaufgabe 22(a).

(d) Gibt es einen binären Suchbaum, dessen Nicht-Blätter (innere Ecken) jeweils 2 Söhne haben und dessen 13 Blätter die Höhen 2, 2, 3, 3, 4, 4, 5, 5, 6, 6, 6, 6, 6 haben?

Behauptung: Einen derartigen binären Suchbaum gibt es nicht.

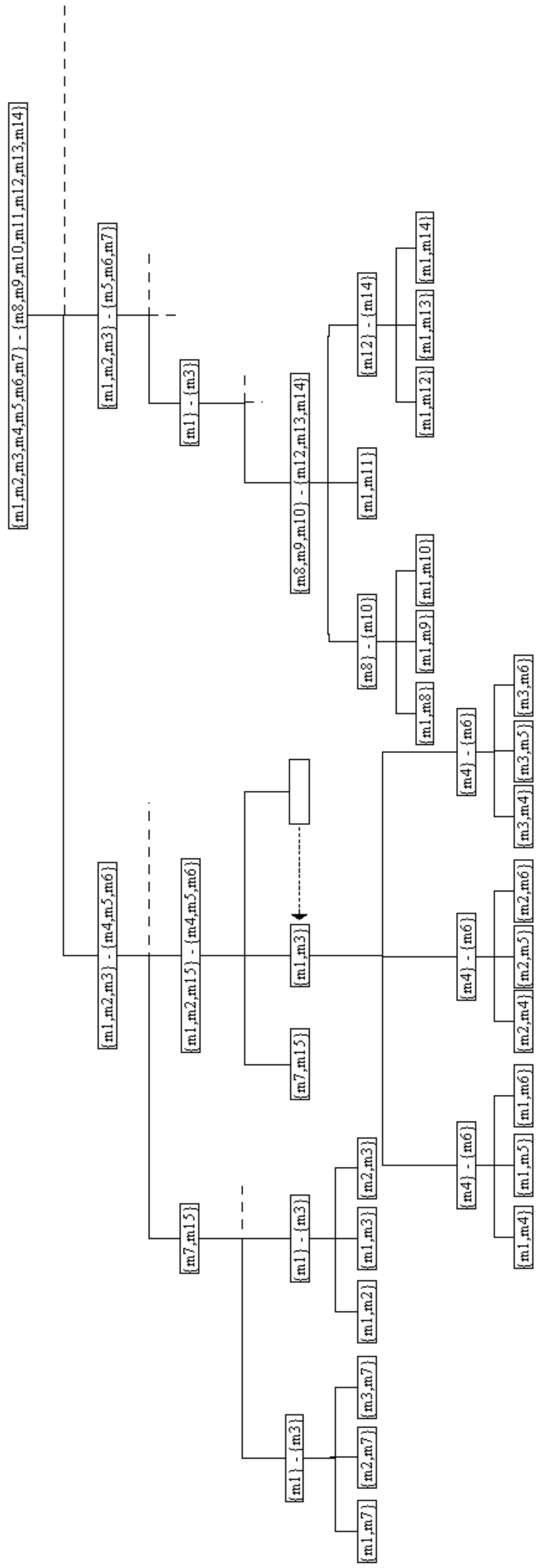


Abbildung 1: Entscheidungsbaum zu Teilaufgabe 22(c)

Algorithmus 4 SucheSchwereMünze($\{m_1, \ldots, m_{15}\}$)

$A \leftarrow \{m_1, \ldots, m_7\}$
$B \leftarrow \{m_8, \ldots, m_{14}\}$
if $A > B$ **then** $\{A$ enthält eine oder beide defekte Münzen; B enthält keine. Im ersten Fall ist m_{15} defekt.$\}$
  $A' \leftarrow \{m_1, m_2, m_3\}$
  $B' \leftarrow \{m_4, m_5, m_6\}$
  if $A' > B'$ **then** $\{A$ enthält eine oder beide defekte Münzen. Möglicherweise ist m_7 oder m_{15} defekt.$\}$
    if $\{m_7\} > \{m_{15}\}$ **then**
      return $\{m_7\} \cup$ SucheSchwereMünze(A') {vgl. Teilaufgabe 22(a)}
    else if $\{m_7\} = \{m_{15}\}$ **then**
      return SucheZweiSchwereMünzen(A')
    else {Es ist $\{m_7\} < \{m_{15}\}$}
      return $\{m_{15}\} \cup$ SucheSchwereMünze(A')
  else if $A' < B'$ **then**
    ... {analog zum Fall $A' > B'$}
  else {Es ist $A' = B'$. Entweder enthalten A', B' beide eine oder beide keine defekte Münze. Im zweiten Fall ist $\{m_7, m_{15}\}$ die gesuchte Menge.}
    $\tilde{A}' \leftarrow \{m_1, m_2, m_{15}\}$
    if $\tilde{A}' > B'$ **then**
      return $\{m_7, m_{15}\}$ $\{m_{15}$ ist defekt, also auch $m_7\}$
    else
      return SucheSchwereMünze(A') $\cup$ SucheSchwereMünze(B')
else if $A < B$ **then**
  ... {analog zum Fall $A > B$}
else {Es ist $A = B$. A und B enthalten jeweils eine defekte Münze.}
  return SucheSchwereMünze(A) $\cup$ SucheSchwereMünze(B)

Beweis:

Da jede innere Ecke zwei Söhne hat, hat jedes Blatt der Höhe $h = 6$ einen Bruder. Da der Baum Höhe 6 haben muß, ist dieser Bruder auch ein Blatt. Folglich kann es nur Bäume der geforderten Art mit einer geraden Anzahl von Blättern auf der untersten Ebene (dem Niveau der Höhe 6) geben. Insbesondere gibt es daher keinen binären Suchbaum mit (wie gefordert) fünf Blättern der Höhe $h = 6$. $\square$

Lösung zu Aufgabe 23: (Public Key Cryptosystem)

In der Vorlesung wurde die RSA-Methode als Beispiel für Falltürfunktionen angegeben. Jeder Benutzer T_l eines Kommunikationssystems wählt 2 verschiedene Primzahlen p_l und q_l mit $n_l = p_l \cdot q_l$ und $p_l, q_l > m$ und dazu eine Zahl e_l mit $3 < e_l < \lambda(n_l) := kgV\{(p_l - 1), (q_l - 1)\}$ und $ggT\{e_l, \lambda(n_l)\} = 1$. ($m$ ist dabei die größte Dezimalzahl, die in dem System zu übertragen ist.) Sein privater Schlüssel ist das Tupel (n_l, d_l). Dabei sei d_l eindeutig gegeben durch $e_l \cdot d_l \equiv 1 \bmod \lambda(n_l)$.

Teilnehmer T_l teilt dem Systemverwalter den öffentlichen Schlüssel (n_l, e_l) mit. Eine Nachricht M von T_i nach T_j (mit $0 \leq M \leq m$) wird von Teilnehmer T_i gemäß $C := E(M) := M^{e_j} \bmod n_j$ chiffriert. T_j decodiert $C = E(M)$ mittels $D(C) = D(M^{e_j} \bmod n_j) := (M^{e_j})^{d_j} \bmod n_j$.

(a) Wir zeigen nun, daß nachfolgender Algorithmus $C = M^{e_j} \bmod n_j$ korrekt berechnet und höchstens $2 \cdot \lceil \log_2 e_j \rceil$ Multiplikationen und $2 \cdot \lceil \log_2 e_j \rceil$ Divisionen benötigt.

Algorithmus 5 RSA

Es sei $e_k^j e_{k-1}^j \dots e_0^j$ die Binärrepräsentation von e_j.
$C := 1$
for $i := k, k-1, \dots, 0$ **do**
  $C := C^2 \bmod n_j$
  if $e_i^j = 1$ **then**
    $C := C \cdot M \bmod n_j$

Sei $e_k^j e_{k-1}^j \dots e_0^j$ wie im Algorithmus die Binärdarstellung von e_j. Dann gilt:

$$\begin{aligned} M^{e_j} &= M^{\sum_{i=0}^{k} e_i^j \cdot 2^i} \\ &= M^{e_0^j} \cdot M^{2e_1^j} \cdot M^{4e_2^j} \cdot \dots \cdot M^{2^k e_k^j} \\ &= M^{e_0^j} \cdot (M^{e_1^j})^2 \cdot (M^{e_2^j})^4 \cdot \dots \cdot (M^{e_k^j})^{2^k} \\ &= M^{e_0^j} \cdot (M^{e_1^j} \cdot (M^{e_2^j} \cdot \dots \cdot (M^{e_k^j})^2 \dots)^2)^2 \end{aligned}$$

Betrachtet man die Belegungen, die die Variable C während eines Laufes des Algorithmus erhält, so läßt sich folgendes erkennen:

Nach dem ersten Schleifendurchlauf ist

$$\begin{aligned} C &= M^{e_k^j} \bmod n_j \\ &=: C_1 \end{aligned}$$

Nach dem zweiten Schleifendurchlauf ist

$$\begin{aligned} C &= (M^{e_{k-1}^j}((C_1)^2 \bmod n_j)) \bmod n_j \\ &= (M^{e_{k-1}^j}((M^{e_k^j} \bmod n_j)^2 \bmod n_j)) \bmod n_j \\ &= (M^{e_{k-1}^j}(M^{e_k^j})^2) \bmod n_j \\ &=: C_2 \end{aligned}$$

Nach dem dritten Schleifendurchlauf ist

$$\begin{aligned} C &= (M^{e^j_{k-2}}((C_2)^2 \bmod n_j)) \bmod n_j \\ &= (M^{e^j_{k-2}}(((M^{e^j_{k-1}}(M^{e^j_k})^2) \bmod n_j)^2 \bmod n_j)) \bmod n_j \\ &= (M^{e^j_{k-2}}(M^{e^j_{k-1}}(M^{e^j_k})^2)^2) \bmod n_j \\ &=: C_3 \end{aligned}$$

Induktiv folgt also, daß nach dem $(k+1)$-ten Schleifendurchlauf

$$\begin{aligned} C &= (M^{e^j_0} \cdot (M^{e^j_1} \cdot (M^{e^j_2} \cdot \ldots \cdot (M^{e^j_k})^2 \ldots)^2)^2) \bmod n_j \\ &= M^{e_j} \bmod n_j \end{aligned}$$

gilt, womit die Korrektheit des Algorithmus gezeigt ist.

Sei $k+1$ die Länge der Binärdarstellung von e_j, also $k+1 = \lceil \log_2 e_j \rceil$. Dann wird die Schleife $k+1$ Mal durchlaufen, wobei in Zeile 4 und evtl. auch in Zeile 6 jeweils eine Multiplikation und eine Division (zur Bestimmung des Restes) durchgeführt werden. Also erhalten wir insgesamt höchstens $2 \cdot \lceil \log_2 e_j \rceil$ Multiplikationen und ebensoviele Divisionen.

(b) Seien für den Adressaten T_2 die Parameter $p_2 = 47$, $q_2 = 59$, $n_2 = p_2 \cdot q_2 = 47 \cdot 59 = 2773$ und $e_2 = 17$ gegeben. Wir codieren die folgende Message

James␣Bond␣ist␣da

von T_1 an T_2 zeichenweise per RSA-Chiffre zu den gegebenen Parametern. Dazu codieren wir diese zunächst über dem Alphabet $\{00, 01, \ldots, 26\}$ mit $\sqcup \to 00, A \to 01, \ldots, Z \to 26$, und anschließend die Symbole dieses Alphabetes (als Zahlen $0, 1, \ldots, 26 \in \mathbb{N}$ interpretiert) zeichenweise.

- Codiere $M =$'J' (es ist $J \to 10$ und zu Beginn $C = 1$):

  $\underline{i=4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e^j_4 = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 10) \bmod 2773 = 10$.
  $\underline{i=3:}$ $C := C^2 \bmod n_j = 10^2 \bmod 2773 = 100$.
  $\underline{i=2:}$ $C := C^2 \bmod n_j = 100^2 \bmod 2773 = 1681$.
  $\underline{i=1:}$ $C := C^2 \bmod n_j = 1681^2 \bmod 2773 = 74$.
  $\underline{i=0:}$ $C := C^2 \bmod n_j = 74^2 \bmod 2773 = 2703$, und wegen $e^j_0 = 1$ weiter $C := (C \cdot M) \bmod n_j = (2703 \cdot 10) \bmod 2773 = 2073$.

  Also ist E('J') $= 2073$.

- Codiere $M =$'A' (es ist $A \to 1$ und zu Beginn $C = 1$):

  $\underline{i=4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e^j_4 = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 1) \bmod 2773 = 1$.
  $\underline{i=3:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$.
  $\underline{i=2:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$.
  $\underline{i=1:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$.
  $\underline{i=0:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e^j_0 = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 1) \bmod 2773 = 1$.

Also ist $E(\text{'A'}) = 1$.

- Codiere $M =$'M' (es ist $M \to 13$ und zu Beginn $C = 1$):

  $\underline{i = 4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 13) \bmod 2773 = 13$.

  $\underline{i = 3:}$ $C := C^2 \bmod n_j = 13^2 \bmod 2773 = 169$.

  $\underline{i = 2:}$ $C := C^2 \bmod n_j = 169^2 \bmod 2773 = 831$.

  $\underline{i = 1:}$ $C := C^2 \bmod n_j = 831^2 \bmod 2773 = 84$.

  $\underline{i = 0:}$ $C := C^2 \bmod n_j = 84^2 \bmod 2773 = 1510$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1510 \cdot 13) \bmod 2773 = 219$.

  Also ist $E(\text{'M'}) = 219$.

- Codiere $M =$'E' (es ist $E \to 5$ und zu Beginn $C = 1$):

  $\underline{i = 4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 5) \bmod 2773 = 5$.

  $\underline{i = 3:}$ $C := C^2 \bmod n_j = 5^2 \bmod 2773 = 25$.

  $\underline{i = 2:}$ $C := C^2 \bmod n_j = 25^2 \bmod 2773 = 625$.

  $\underline{i = 1:}$ $C := C^2 \bmod n_j = 625^2 \bmod 2773 = 2405$.

  $\underline{i = 0:}$ $C := C^2 \bmod n_j = 2405^2 \bmod 2773 = 2320$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (2320 \cdot 5) \bmod 2773 = 508$.

  Also ist $E(\text{'E'}) = 508$.

- Codiere $M =$'S' (es ist $S \to 19$ und zu Beginn $C = 1$):

  $\underline{i = 4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 19) \bmod 2773 = 19$.

  $\underline{i = 3:}$ $C := C^2 \bmod n_j = 19^2 \bmod 2773 = 361$.

  $\underline{i = 2:}$ $C := C^2 \bmod n_j = 361^2 \bmod 2773 = 2763$.

  $\underline{i = 1:}$ $C := C^2 \bmod n_j = 2763^2 \bmod 2773 = 100$.

  $\underline{i = 0:}$ $C := C^2 \bmod n_j = 100^2 \bmod 2773 = 1681$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1681 \cdot 19) \bmod 2773 = 1436$.

  Also ist $E(\text{'S'}) = 1436$.

- Codiere $M =$'$\sqcup$' (es ist $\sqcup \to 00$ und zu Beginn $C = 1$):

  $\underline{i = 4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 0) \bmod 2773 = 0$.

  $\underline{i = 3:}$ $C := C^2 \bmod n_j = 0^2 \bmod 2773 = 0$.

  $\underline{i = 2:}$ $C := C^2 \bmod n_j = 0^2 \bmod 2773 = 0$.

  $\underline{i = 1:}$ $C := C^2 \bmod n_j = 0^2 \bmod 2773 = 0$.

  $\underline{i = 0:}$ $C := C^2 \bmod n_j = 0^2 \bmod 2773 = 0$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (0 \cdot 0) \bmod 2773 = 0$.

  Also ist $E(\text{'}\sqcup\text{'}) = 0$.

- Codiere $M =$'B' (es ist $B \to 2$ und zu Beginn $C = 1$):

  $\underline{i = 4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 2) \bmod 2773 = 2$.

  $\underline{i = 3:}$ $C := C^2 \bmod n_j = 2^2 \bmod 2773 = 4$.

$\underline{i=2:}$ $C := C^2 \bmod n_j = 4^2 \bmod 2773 = 16$.

$\underline{i=1:}$ $C := C^2 \bmod n_j = 16^2 \bmod 2773 = 256$.

$\underline{i=0:}$ $C := C^2 \bmod n_j = 256^2 \bmod 2773 = 1757$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1757 \cdot 2) \bmod 2773 = 741$.

Also ist $E(\text{'B'}) = 741$.

- Codiere $M =$'O' (es ist $O \to 15$ und zu Beginn $C = 1$):

  $\underline{i=4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 15) \bmod 2773 = 15$.

  $\underline{i=3:}$ $C := C^2 \bmod n_j = 15^2 \bmod 2773 = 225$.

  $\underline{i=2:}$ $C := C^2 \bmod n_j = 225^2 \bmod 2773 = 711$.

  $\underline{i=1:}$ $C := C^2 \bmod n_j = 711^2 \bmod 2773 = 835$.

  $\underline{i=0:}$ $C := C^2 \bmod n_j = 835^2 \bmod 2773 = 1202$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1202 \cdot 15) \bmod 2773 = 1392$.

  Also ist $E(\text{'O'}) = 1392$.

- Codiere $M =$'N' (es ist $N \to 14$ und zu Beginn $C = 1$):

  $\underline{i=4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 14) \bmod 2773 = 14$.

  $\underline{i=3:}$ $C := C^2 \bmod n_j = 14^2 \bmod 2773 = 196$.

  $\underline{i=2:}$ $C := C^2 \bmod n_j = 196^2 \bmod 2773 = 2367$.

  $\underline{i=1:}$ $C := C^2 \bmod n_j = 2367^2 \bmod 2773 = 1229$.

  $\underline{i=0:}$ $C := C^2 \bmod n_j = 1229^2 \bmod 2773 = 1929$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1929 \cdot 14) \bmod 2773 = 2049$.

  Also ist $E(\text{'N'}) = 2049$.

- Codiere $M =$'D' (es ist $D \to 4$ und zu Beginn $C = 1$):

  $\underline{i=4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 4) \bmod 2773 = 4$.

  $\underline{i=3:}$ $C := C^2 \bmod n_j = 4^2 \bmod 2773 = 16$.

  $\underline{i=2:}$ $C := C^2 \bmod n_j = 16^2 \bmod 2773 = 256$.

  $\underline{i=1:}$ $C := C^2 \bmod n_j = 256^2 \bmod 2773 = 1757$.

  $\underline{i=0:}$ $C := C^2 \bmod n_j = 1757^2 \bmod 2773 = 700$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (700 \cdot 4) \bmod 2773 = 27$.

  Also ist $E(\text{'D'}) = 27$.

- Codiere $M =$'⊔': Wie bereits gesehen ist $E(\text{'}\sqcup\text{'}) = 0$.

- Codiere $M =$'I' (es ist $I \to 9$ und zu Beginn $C = 1$):

  $\underline{i=4:}$ $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 9) \bmod 2773 = 9$.

  $\underline{i=3:}$ $C := C^2 \bmod n_j = 9^2 \bmod 2773 = 81$.

  $\underline{i=2:}$ $C := C^2 \bmod n_j = 81^2 \bmod 2773 = 1015$.

  $\underline{i=1:}$ $C := C^2 \bmod n_j = 1015^2 \bmod 2773 = 1442$.

  $\underline{i=0:}$ $C := C^2 \bmod n_j = 1442^2 \bmod 2773 = 2387$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (2387 \cdot 9) \bmod 2773 = 2072$.

Also ist $E(\text{'I'}) = 2072$.

- Codiere $M =$'S': Wie bereits gesehen ist $E(\text{'S'}) = 1436$.
- Codiere $M =$'T' (es ist $T \to 20$ und zu Beginn $C = 1$):

  $\underline{i = 4}$: $C := C^2 \bmod n_j = 1^2 \bmod 2773 = 1$, und wegen $e_4^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1 \cdot 20) \bmod 2773 = 20$.
  $\underline{i = 3}$: $C := C^2 \bmod n_j = 20^2 \bmod 2773 = 400$.
  $\underline{i = 2}$: $C := C^2 \bmod n_j = 400^2 \bmod 2773 = 1939$.
  $\underline{i = 1}$: $C := C^2 \bmod n_j = 1939^2 \bmod 2773 = 2306$.
  $\underline{i = 0}$: $C := C^2 \bmod n_j = 2306^2 \bmod 2773 = 1795$, und wegen $e_0^j = 1$ weiter $C := (C \cdot M) \bmod n_j = (1795 \cdot 20) \bmod 2773 = 2624$.

  Also ist $E(\text{'T'}) = 2624$.
- Codiere $M =$'⊔': Wie bereits gesehen ist $E(\text{'⊔'}) = 0$.
- Codiere $M =$'D': Wie bereits gesehen ist $E(\text{'D'}) = 27$.
- Codiere $M =$'A': Wie bereits gesehen ist $E(\text{'A'}) = 1$.

Also lautet die zeichenweise chiffrierte Version von 'JAMES BOND IST DA':

> 2073, 1, 219, 508, 1436, 0, 741, 1392, 2049, 27, 0, 2072, 1436, 2624, 0, 27, 1