Chomsky-Grammatik zur Erzeugung aller 126 Kombinationen von 4 aus 9 Buchstaben a, b, c, d, e, f, g, h, i

Friedrich-Schiller-Universität Jena
Modul Molekulare Algorithmen, Sommersemester 2021

Denis Grabiger und Pauline Rauh

1 Chomsky Grammatik

Eine Chomsky Grammatik ist ein 4-Tupel G mit G = (V, Σ, P, S) dabei ist:
V – das Alphabet der Variablensymbole
Σ – das Alphabet der Terminalsymbole
P – die endliche Regelmenge
S – das Startsymbol
Eine Regel in P, auch Produktionsregel oder Ableitung genannt, mit (u, w) ∈ P wird geschrieben als u → w.

2 Problemstellung

Aufgabenstellung ist es eine Chomsky Grammatik G = (V, Σ, P, S) zu entwickeln, welche alle Kombinationen von 4 aus 9 der Buchstaben (a, b, c, d, e, f, g, h, i) erzeugt. Dabei soll die Grammatik mit weitestgehend wenig Regeln auskommen, nicht mehr als 50. Des Weiteren soll erklärt werden, wie viele Regelanwendungen maximal nötig sind, um ein Wort mit Hilfe der Grammatik zu erzeugen.

Es soll also eine Grammatik erstellt werden welche die Wörter "abcd", "abce", ..., "fghi" erzeugt. Daher sind verschiedene Probleme zu beachten wie:

1. Keiner der 9 Buchstaben darf doppelt vorkommen
2. Die Wörter müssen genau 4 Buchstaben lang sein

3 Lösungsansätze

3.1 Lösungsansatz 1 - Triviale Lösung

Der initiale Lösungsansatz für das Problem bestand darin, alle möglichen Kombinationen als Regeln aufzuschreiben. Die Grammatik dafür ist wie folgt:
G = (V, Σ, P, S)
V = {S}
Σ = {a, b, c, d, e, f, g, h, i}
P = {S → abcd, S → abce, ... , S → fghi}

Da diese Grammatik jedoch 126 Regeln benötigt, da es insgesammt 126 verschiedene Kombinationen für die 9 Buchstaben gibt, ist die Lösung nicht ausreichend.

3.2 Lösungsansatz 2

Das Prinzip des nächsten Lösungsansatzes war vom Startsymbol aus 4 Variablensymbole {1,2,3,4} abzuleiten (S → 1234). Diese können unterschiedlich zu Variablensymbolen {A, B, C, D, E, F, G, H, I} abgeleitet werden.

Da nicht jeder Buchstabe an jeder Stelle im Wort vorkommen muss sondern bei einer nach Alphabet sortierten Lösung für alle Kombinationen (abcd, abce, .. , fghi) der Anfangsbuchstabe aller Wörter nur im Bereich {a, b, c, d, e, f} und der Endbuchstabe aller Wörter im Bereich {d, e, f, g, h, i} ist, werden die 1 und 4 entsprechend nur in diese Buchstaben als Variablen abgeleitet. An der 2. befinden sich bei der sortierten Lösung die Buchstaben {b, c, d, e, f, g} und an der 3. Stelle die Buchstaben {c, d, e, f, g, h}.

Daher folgen die Regeln:
1 → A, 1 → B, 1 → C, 1 → D, 1 → E, 1 → F
2 → B, 2 → C, 2 → D, 2 → E, 2 → F, 2 → G
3 → C, 3 → D, 3 → E, 3 → F , 3 → F, 3 → H
4 → D, 4 → E, 4 → F, 4 → G, 4 → H, 4 → I

Außerdem werden noch Regeln benötigt, durch die die Nichtterminale in Terminale umgewandelt werden, also:
A → a, B → B, C → c, D → d, E → e, F → f, G → g, G → g, H → H, I → i

Die Regeln die hierfür benötigt werden sind 1 für das Startsymbol und jeweils 6 für 1 bis 4 und außerdem 9 zum umwandeln der Nichtterminale in Terminale. Damit benötigt dieser Ansatz nur 34 Regeln.

Alle Kombinationen sind enthalten, da die Grammatik die nach alphabet sortierte Ordnung aller Kombinationen wiederspiegelt, jedoch sind auch Wörter enthalten, in denen

sich Buchstaben doppeln. Zum Beispiel wäre ffff ein Wort der Sprache, was jedoch nicht eine Kombination aus 4 der 9 Buchstaben ist. Daher ist diese Grammatik nicht korrekt.

3.3 Lösungsansatz 3

Der nächste Ansatz ist eine Verbesserung des Lösungsansatzes 2. Die Regeln zur Umwandlung der Terminalen in Nichtterminale werden hierfür verändert. Um die Dopplungen der Buchstaben zu verhindern werden nur Buchstaben in Terminale umgewandelt, welche nicht mehrmals vorkommen. Dies geschiet durch folgenden Ansatz:

A kann nur in 1 vorkommen, daher braucht man hierfür keine extra Regel außer A $\rightarrow$ a.

B kann in 1 und 2 vorkommen, daher wird es nur in ein Nichtterminal umgewandelt, wenn es nur 1x vorkommt. Dies wird sichergestellt durch folgende Regeln:
BC $\rightarrow$ bC, BD $\rightarrow$ bD, ... AB $\rightarrow$ Ab, CB $\rightarrow$ Cb.

Das Problem dieses Lösungsansatzes ist vorallem, dass viel zu viele Regeln benötigt werden. Außderdem entsteht ein Problem dadurch, dass die Nichtterminale stückweise umgewandelt werden und so die Überprüfung auf beispielsweise DF nicht gemacht werden kann, da der Nichtterminal D schon in d umgewandelt wurde.

Daher ist auch dieser Lösungsansatz nicht zielführend für das gestellte Problem.

3.4 Lösungsansatz 4

Die Idee dieses Lösungsansatzes ist es, die möglichen Positionen von Buchstaben zu begrenzen. Zum Beispiel reicht es, wenn der Buchstabe a nur an erster Stelle vorkommen kann, und der Buchstabe i nur an letzter. Da weiterhin Dopplungen auftreten können, werden zwischen den Terminalen zwei nicht Terminale platziert. Um diese nicht Terminale zu entfernen, gibt es Regeln, allerdings nur für Fälle, in denen keine Dopplung auftritt.

Aufteilung der Buchstaben auf die vier möglichen Positionen gestaltet sich wie folgt.

Position 1 muss nur die Buchstaben a, b, c, d, e, f annehmen können.
Position 2: b, c, d, e, f, g
Position 3: c, d, e, f, g, h
Position 4: d, e, f, g, h, i

S $\rightarrow$ 1234
1 $\rightarrow$ aA | bB | cC | ... | fF
2 $\rightarrow$ BbB | X
3 $\rightarrow$ X | HhH
4 $\rightarrow$ Dd | Ee | Ff | ... | Ii
X $\rightarrow$ CcC | DdD | EeE | FfF | GgG

1+6+2+2+6+5 = 22 Regeln

Beispiel der Erzeugung eines Wortes mit den obigen Regeln:
aABbBDdDEe

Nun definieren wir Regeln welche nur Wörter zulassen, bei denen die restlichen nicht Terminalen keine Dopplung enthalten. Regeln, welche nie gebraucht werden, würden wie z.B. AI → können gleich weggelassen werden.
AB → ∼, AC → ∼, AD → ∼, AE → ∼, AF → ∼, AG → ∼
BC → ∼, BD → ∼, ... , BH → ∼
CB → ∼, CD → ∼, ... , CI → ∼
DB → ∼, DC → ∼, DE → ∼, ... , DI → ∼
EB → ∼, EC → ∼, ED → ∼, EF → ∼, ... , EI → ∼
FB → ∼, ... , FE → ∼, FG → ∼, FH → ∼, FI → ∼
GC → ∼, ... , GF → ∼, GH → ∼, GI → ∼
HD → ∼, ... , HG → ∼, HI → ∼
6 + 6 + 7 + 7 + 7 + 7 + 6 + 5 = 51 Regeln

Also insgesamt 73 Regeln, daher haben wir 23 Regeln zu viel.

3.5 Finale Lösung

Dieser Lösungsansatz basiert darauf, die Buchstaben in zwei Teile zu untergliedern. Dafür wurden die Buchstaben von a bis i in der Mitte getrennt. Die "linke Hälfte"(L) der Buchstaben besteht aus a, b, c und d, während die "rechte Hälfte"(R) aus e, f, g, h und i besteht. Da sich die neun Buchstaben nicht restlos durch 2 Teilen lassen, hat die "rechte Hälfte" einen Buchstaben mehr.

Alle Kombinationen für die Buchstaben als Regeln festzulegen benötigt zu viele Regeln (Ansatz 1), während keine Kombinationen für die Buchstaben als Regeln festzulegen nicht korrekt ist (Ansatz 2). Daher ist dieser Lösungsansatz eine Kombination der vorherigen Ansätze.

Durch das Aufteilen der Buchstaben in zwei Gruppen ist es möglich, die Kombinationen beider Teile zu bilden und nur eine geringe Anzahl der Kombinationen als Regeln festzulegen. Außerdem werden die beiden Gruppen miteinander verbunden, so dass immer 4 Buchstaben vorhanden sind. Dafür benötigen wir mehrere Regeln. Der erste Regelsatz wird dafür benötigt, dass alle Kombinationen aus den beiden Gruppen L und R vorhanden sind. Daher ergeben sich die 6 Regeln:
S → abcd
S → efgh, S → efgi, S → efhi, S → eghi, S → fghi

Der nächste Regelsatz stellt sicher, dass alle Kombinationen aus 3 Buchstaben einer Seite mit einem Buchstaben der anderen Seite verbunden werden:

$S \rightarrow XY$, $S \rightarrow MN$
$X \rightarrow abc$, $X \rightarrow abd$, $X \rightarrow acd$, $X \rightarrow bcd$
$Y \rightarrow e$, $Y \rightarrow f$, $Y \rightarrow g$, $Y \rightarrow h$, $Y \rightarrow i$
$M \rightarrow a$, $M \rightarrow b$, $M \rightarrow c$, $M \rightarrow d$
$N \rightarrow efg$, $N \rightarrow efh$, $N \rightarrow efi$, $N \rightarrow egh$, $N \rightarrow egi$, $N \rightarrow ehi$, $N \rightarrow fgh$, $N \rightarrow fgi$, $N \rightarrow fhi$, $N \rightarrow ghi$

XY steht für eine Kombination aus drei Buchstaben von L und einem Buchstaben von R, während MN für eine Kombination aus drei Buchstaben von R und einem Buchstaben von L steht. Dafür werden 25 Regeln benötigt.

Abschließend wird noch ein Regelsatz benötigt, durch welchen jeweils zwei Buchstaben von L und R miteinander kombiniert werden.
$S \rightarrow LR$

$L \rightarrow ab$, $L \rightarrow ac$, $L \rightarrow ad$, $L \rightarrow bc$, $L \rightarrow bd$, $L \rightarrow cd$

$R \rightarrow ef$, $R \rightarrow eg$, $R \rightarrow eh$, $R \rightarrow ei$, $R \rightarrow fg$, $R \rightarrow fh$, $R \rightarrow fi$, $R \rightarrow gh$, $R \rightarrow gi$, $R \rightarrow hi$

Alle 2er Kombinationen aus L und R wurden dafür gebildet und durch die $S \rightarrow LR$ Regel ist es möglich, alle Kombinationen aus den Kombinierten Buchstaben für die Listen zu bilden. Dafür werden 17 Regeln benötigt.

Schlussendlich ergibt sich durch den Lösungsansatz folgende Grammatik:
$G = (V , \Sigma, P, S)$
$V = \{S, X, Y, L, R, M, N\}$
$\Sigma = \{a, b, c, d, e, f, g, h, i\}$
$P = \{S \rightarrow XY, S \rightarrow MN, S \rightarrow LR, S \rightarrow abcd, S \rightarrow efgh, S \rightarrow efgi, S \rightarrow efhi, S \rightarrow eghi, S \rightarrow fghi, X \rightarrow abc, X \rightarrow abd, X \rightarrow acd, X \rightarrow bcd, Y \rightarrow e, Y \rightarrow f, Y \rightarrow g, Y \rightarrow h, Y \rightarrow i, M \rightarrow a, M \rightarrow b, M \rightarrow c, M \rightarrow d, N \rightarrow efg, N \rightarrow efh, N \rightarrow efi, N \rightarrow egh, N \rightarrow egi, N \rightarrow ehi, N \rightarrow fgh, N \rightarrow fgi, N \rightarrow fhi, N \rightarrow ghi, L \rightarrow ab, L \rightarrow ac, L \rightarrow ad, L \rightarrow bc, L \rightarrow bd, L \rightarrow cd, R \rightarrow ef, R \rightarrow eg, R \rightarrow eh, R \rightarrow ei, R \rightarrow fg , R \rightarrow fh, R \rightarrow fi, R \rightarrow gh, R \rightarrow gi, R \rightarrow hi\}$

Dopplungen von Buchstaben sind in dieser Grammatik nicht möglich, da die beiden Listen disjunkt sind, immer nur die beiden Listen miteinander kombiniert werden und die festgelegten Regeln zum Umwandeln in Terminale keine Dopplungen enthalten.

Diese Grammatik bildet alle 4er Kombinationen aus den neun Buchstaben und benötigt insgesamt 48 Regeln. Daher erfüllt sie die Aufgabenstellung.

4 Korrektheit

Um die Korrektheit besser zeigen zu können, wurde ein Python Programm entwickelt, welches zuerst die korrekten 126 Kombinationen erzeugt. Danach erstellt das Programm eine Liste. Diese enthält die Kombinationen, welche von der in Abschnitt 3.5 beschriebenen Chomsky Grammatik erzeugt werden. Das Programm befindet sich im Anhang 1.

Zuerst werden die 4er Kombinationen aus den beiden Listen, {a, b, c, d} und {e, f, g, h, i} hinzugefügt. Danach bildet das Programm alle 3er Kombinationen aus beiden Listen und kombiniert diese jeweils mit jedem einzelnen Element der anderen Liste. Nun werden alle 2er Kombinationen beider Listen erzeugt und jede Kombination der einen Liste mit jeder Kombination der anderen Liste aneinandergefügt.

Alle diese Kombinationen werden sortiert und an die "result" Liste angefügt, welche zum Schluss ebenfalls sortiert wird. Nun werden die Liste mit den korrekten Kombinationen und die Liste mit allen unseren Kombinationen ("result"), subtrahiert. Das Ergebnis ist dann die Differenz beider Listen, also die Menge an Elementen, welche nur in einer der beiden Listen enthalten ist.

Wie in der Programmausgabe auf Anhang 2 zu sehen, ist die Differenz beider Listen leer. Das heißt die Menge aller Wörter, welche durch unsere Chomsky Grammatik erzeugt wird, ist gleich der Menge aller Kombinationen von 4 aus 9 Buchstaben.

5 Analyse der Grammatik

Die minimale Anzahl an Regelanwendungen tritt auf, wenn alle vier Buchstaben des Wortes aus einer der beiden Listen stammen. In diesem Fall gibt es nur genau eine Regel anzuwenden. Zum Beispiel für das Wort "abcd" die Regel S $\rightarrow$ abcd.

Die maximale Anzahl an Regelanwendungen tritt bei allen anderen Fällen auf. Nämlich entweder, wenn drei Buchstaben aus einer und ein Buchstabe aus der anderen Liste kommen oder wenn zwei Buchstaben aus einer und zwei aus der anderen Liste benötigt werden. Zum Beispiel werden die Wörter "abef" und "abdf" erzeugt mit:
S $\rightarrow$ LR
L $\rightarrow$ ab
R $\rightarrow$ ef
und
S $\rightarrow$ MN
M $\rightarrow$ acd
N $\rightarrow$ f.

Somit brauchen alle anderen Wörter genau drei Regelanwendungen. Es gibt sechs Wörter, bei denen alle vier Buchstaben aus nur einer Liste stammen, daher gibt es noch 120 Wörter welche drei Regelanwendungen nötig sind.

Wir können also sagen es gibt minimal eine Regelanwendung, maximal 3 Regelanwendungen und durchschnittlich etwa 2.9 Regelanwendungen.

6 Anhang

6.1 Anhang 1

```
1   import itertools
2
3   a = ['a', 'b', 'c', 'd', 'e', 'f', 'g', 'h', 'i']
4
5   #Bilde zum Vergleich die korrekte Anwort: alle 126 Kombinationen für 4 Elemente aus
    der Liste a
6   correctAnswer = list(itertools.combinations(a,4))
7   print(correctAnswer)
8   print("Die richtige Lösung hat", len(correctAnswer), "Elemente.")
9
10
11  left = ['a', 'b', 'c', 'd']
12  right = ['e', 'f', 'g', 'h', 'i']
13
14  #Zuerst fügen für jede 4er Kombination innerhalb einer Liste hinzu
15  result = list(itertools.combinations(left,4)) + list(itertools.combinations(right,4))
16
17  #Nun bilden wir jede 3er Kombination innerhalb einer Liste und fügen jeder 3er
    Kombination jedes einzelne Element aus der anderen Liste hinzu
18  leftList = list(itertools.combinations(left,3))
19  rightList = list(itertools.combinations(right,3))
20
21  #Zum Beispiel ist das die leftList: [('a', 'b', 'c'), ('a', 'b', 'd'), ('a', 'c',
    'd'), ('b', 'c', 'd')] und jetzt fügen wir jedes einzelne Element von ['e', 'f',
    'g', 'h', 'i'] zu jedem Element der leftList hinzu
22  for x in right:
23      for y in leftList:
24          k = list(y)
25          k.append(x)
26          k.sort()
27          result.append(tuple(k))
28
29  #neue Liste: ['a', 'b', 'c', 'e'], ['a', 'b', 'c', 'f'], .. , [('b', 'c', 'd', 'i')]
30
31  #Hier das gleiche für alle 3er Kombinationen aus der rechten Liste & die einzelnen
    der linken Liste
32  for x in left:
33      for y in rightList:
34          k = list(y)
35          k.append(x)
36          k.sort()
37          result.append(tuple(k))
38
39
40  #Hier bilden wir erstmal jede 2er Kombination aus der rechten & der linken Liste
41  leftList = list(itertools.combinations(left,2))
42  rightList = list(itertools.combinations(right,2))
43
44  #und fügen dann jedes Element aus der leftList an jedes Element aus der rightList
    z.B. ('c', 'd') + ('g', 'i') für unser neues Element ('c', 'd', 'h', 'i')
45  for x in leftList:
46      for y in rightList:
47          k = list(x)
48          for z in y:
49              k.append(z)
50          k.sort()
51          result.append(tuple(k))
52
53  result.sort()
54  print(result)
55  print("Unsere Lösung hat:",len(result),"Elemente.")
56
57  print("Die Elemente von beiden unterscheiden sich um: ", list(set(correctAnswer) -
    set(result)))
```

6.2 Anhang 2

[('a', 'b', 'c', 'd'), ('a', 'b', 'c', 'e'), ('a', 'b', 'c', 'f'), ('a', 'b', 'c', 'g'), ('a', 'b', 'c', 'h'), ('a', 'b', 'c', 'i'), ('a', 'b', 'd', 'e'), ('a', 'b', 'd', 'f'), ('a', 'b', 'd', 'g'), ('a', 'b', 'd', 'h'), ('a', 'b', 'd', 'i'), ('a', 'b', 'e', 'f'), ('a', 'b', 'e', 'g'), ('a', 'b', 'e', 'h'), ('a', 'b', 'e', 'i'), ('a', 'b', 'f', 'g'), ('a', 'b', 'f', 'h'), ('a', 'b', 'f', 'i'), ('a', 'b', 'g', 'h'), ('a', 'b', 'g', 'i'), ('a', 'b', 'h', 'i'), ('a', 'c', 'd', 'e'), ('a', 'c', 'd', 'f'), ('a', 'c', 'd', 'g'), ('a', 'c', 'd', 'h'), ('a', 'c', 'd', 'i'), ('a', 'c', 'e', 'f'), ('a', 'c', 'e', 'g'), ('a', 'c', 'e', 'h'), ('a', 'c', 'e', 'i'), ('a', 'c', 'f', 'g'), ('a', 'c', 'f', 'h'), ('a', 'c', 'f', 'i'), ('a', 'c', 'g', 'h'), ('a', 'c', 'g', 'i'), ('a', 'c', 'h', 'i'), ('a', 'd', 'e', 'f'), ('a', 'd', 'e', 'g'), ('a', 'd', 'e', 'h'), ('a', 'd', 'e', 'i'), ('a', 'd', 'f', 'g'), ('a', 'd', 'f', 'h'), ('a', 'd', 'f', 'i'), ('a', 'd', 'g', 'h'), ('a', 'd', 'g', 'i'), ('a', 'd', 'h', 'i'), ('a', 'e', 'f', 'g'), ('a', 'e', 'f', 'h'), ('a', 'e', 'f', 'i'), ('a', 'e', 'g', 'h'), ('a', 'e', 'g', 'i'), ('a', 'e', 'h', 'i'), ('a', 'f', 'g', 'h'), ('a', 'f', 'g', 'i'), ('a', 'f', 'h', 'i'), ('a', 'g', 'h', 'i'), ('b', 'c', 'd', 'e'), ('b', 'c', 'd', 'f'), ('b', 'c', 'd', 'g'), ('b', 'c', 'd', 'h'), ('b', 'c', 'd', 'i'), ('b', 'c', 'e', 'f'), ('b', 'c', 'e', 'g'), ('b', 'c', 'e', 'h'), ('b', 'c', 'e', 'i'), ('b', 'c', 'f', 'g'), ('b', 'c', 'f', 'h'), ('b', 'c', 'f', 'i'), ('b', 'c', 'g', 'h'), ('b', 'c', 'g', 'i'), ('b', 'c', 'h', 'i'), ('b', 'd', 'e', 'f'), ('b', 'd', 'e', 'g'), ('b', 'd', 'e', 'h'), ('b', 'd', 'e', 'i'), ('b', 'd', 'f', 'g'), ('b', 'd', 'f', 'h'), ('b', 'd', 'f', 'i'), ('b', 'd', 'g', 'h'), ('b', 'd', 'g', 'i'), ('b', 'd', 'h', 'i'), ('b', 'e', 'f', 'g'), ('b', 'e', 'f', 'h'), ('b', 'e', 'f', 'i'), ('b', 'e', 'g', 'h'), ('b', 'e', 'g', 'i'), ('b', 'e', 'h', 'i'), ('b', 'f', 'g', 'h'), ('b', 'f', 'g', 'i'), ('b', 'f', 'h', 'i'), ('b', 'g', 'h', 'i'), ('c', 'd', 'e', 'f'), ('c', 'd', 'e', 'g'), ('c', 'd', 'e', 'h'), ('c', 'd', 'e', 'i'), ('c', 'd', 'f', 'g'), ('c', 'd', 'f', 'h'), ('c', 'd', 'f', 'i'), ('c', 'd', 'g', 'h'), ('c', 'd', 'g', 'i'), ('c', 'd', 'h', 'i'), ('c', 'e', 'f', 'g'), ('c', 'e', 'f', 'h'), ('c', 'e', 'f', 'i'), ('c', 'e', 'g', 'h'), ('c', 'e', 'g', 'i'), ('c', 'e', 'h', 'i'), ('c', 'f', 'g', 'h'), ('c', 'f', 'g', 'i'), ('c', 'f', 'h', 'i'), ('c', 'g', 'h', 'i'), ('d', 'e', 'f', 'g'), ('d', 'e', 'f', 'h'), ('d', 'e', 'f', 'i'), ('d', 'e', 'g', 'h'), ('d', 'e', 'g', 'i'), ('d', 'e', 'h', 'i'), ('d', 'f', 'g', 'h'), ('d', 'f', 'g', 'i'), ('d', 'f', 'h', 'i'), ('d', 'g', 'h', 'i'), ('e', 'f', 'g', 'h'), ('e', 'f', 'g', 'i'), ('e', 'f', 'h', 'i'), ('e', 'g', 'h', 'i'), ('f', 'g', 'h', 'i')]

Die richtige Lösung hat 126 Elemente.

[('a', 'b', 'c', 'd'), ('a', 'b', 'c', 'e'), ('a', 'b', 'c', 'f'), ('a', 'b', 'c', 'g'), ('a', 'b', 'c', 'h'), ('a', 'b', 'c', 'i'), ('a', 'b', 'd', 'e'), ('a', 'b', 'd', 'f'), ('a', 'b', 'd', 'g'), ('a', 'b', 'd', 'h'), ('a', 'b', 'd', 'i'), ('a', 'b', 'e', 'f'), ('a', 'b', 'e', 'g'), ('a', 'b', 'e', 'h'), ('a', 'b', 'e', 'i'), ('a', 'b', 'f', 'g'), ('a', 'b', 'f', 'h'), ('a', 'b', 'f', 'i'), ('a', 'b', 'g', 'h'), ('a', 'b', 'g', 'i'), ('a', 'b', 'h', 'i'), ('a', 'c', 'd', 'e'), ('a', 'c', 'd', 'f'), ('a', 'c', 'd', 'g'), ('a', 'c', 'd', 'h'), ('a', 'c', 'd', 'i'), ('a', 'c', 'e', 'f'), ('a', 'c', 'e', 'g'), ('a', 'c', 'e', 'h'), ('a', 'c', 'e', 'i'), ('a', 'c', 'f', 'g'), ('a', 'c', 'f', 'h'), ('a', 'c', 'f', 'i'), ('a', 'c', 'g', 'h'), ('a', 'c', 'g', 'i'), ('a', 'c', 'h', 'i'), ('a', 'd', 'e', 'f'), ('a', 'd', 'e', 'g'), ('a', 'd', 'e', 'h'), ('a', 'd', 'e', 'i'), ('a', 'd', 'f', 'g'), ('a', 'd', 'f', 'h'), ('a', 'd', 'f', 'i'), ('a', 'd', 'g', 'h'), ('a', 'd', 'g', 'i'), ('a', 'd', 'h', 'i'), ('a', 'e', 'f', 'g'), ('a', 'e', 'f', 'h'), ('a', 'e', 'f', 'i'), ('a', 'e', 'g', 'h'), ('a', 'e', 'g', 'i'), ('a', 'e', 'h', 'i'), ('a', 'f', 'g', 'h'), ('a', 'f', 'g', 'i'), ('a', 'f', 'h', 'i'), ('a', 'g', 'h', 'i'), ('b', 'c', 'd', 'e'), ('b', 'c', 'd', 'f'), ('b', 'c', 'd', 'g'), ('b', 'c', 'd', 'h'), ('b', 'c', 'd', 'i'), ('b', 'c', 'e', 'f'), ('b', 'c', 'e', 'g'), ('b', 'c', 'e', 'h'), ('b', 'c', 'e', 'i'), ('b', 'c', 'f', 'g'), ('b', 'c', 'f', 'h'), ('b', 'c', 'f', 'i'), ('b', 'c', 'g', 'h'), ('b', 'c', 'g', 'i'), ('b', 'c', 'h', 'i'), ('b', 'd', 'e', 'f'), ('b', 'd', 'e', 'g'),

('b', 'd', 'e', 'h'), ('b', 'd', 'e', 'i'), ('b', 'd', 'f', 'g'), ('b', 'd', 'f', 'h'), ('b', 'd', 'f', 'i'), ('b', 'd', 'g', 'h'), ('b', 'd', 'g', 'i'), ('b', 'd', 'h', 'i'), ('b', 'e', 'f', 'g'), ('b', 'e', 'f', 'h'), ('b', 'e', 'f', 'i'), ('b', 'e', 'g', 'h'), ('b', 'e', 'g', 'i'), ('b', 'e', 'h', 'i'), ('b', 'f', 'g', 'h'), ('b', 'f', 'g', 'i'), ('b', 'f', 'h', 'i'), ('b', 'g', 'h', 'i'), ('c', 'd', 'e', 'f'), ('c', 'd', 'e', 'g'), ('c', 'd', 'e', 'h'), ('c', 'd', 'e', 'i'), ('c', 'd', 'f', 'g'), ('c', 'd', 'f', 'h'), ('c', 'd', 'f', 'i'), ('c', 'd', 'g', 'h'), ('c', 'd', 'g', 'i'), ('c', 'd', 'h', 'i'), ('c', 'e', 'f', 'g'), ('c', 'e', 'f', 'h'), ('c', 'e', 'f', 'i'), ('c', 'e', 'g', 'h'), ('c', 'e', 'g', 'i'), ('c', 'e', 'h', 'i'), ('c', 'f', 'g', 'h'), ('c', 'f', 'g', 'i'), ('c', 'f', 'h', 'i'), ('c', 'g', 'h', 'i'), ('d', 'e', 'f', 'g'), ('d', 'e', 'f', 'h'), ('d', 'e', 'f', 'i'), ('d', 'e', 'g', 'h'), ('d', 'e', 'g', 'i'), ('d', 'e', 'h', 'i'), ('d', 'f', 'g', 'h'), ('d', 'f', 'g', 'i'), ('d', 'f', 'h', 'i'), ('d', 'g', 'h', 'i'), ('e', 'f', 'g', 'h'), ('e', 'f', 'g', 'i'), ('e', 'f', 'h', 'i'), ('e', 'g', 'h', 'i'), ('f', 'g', 'h', 'i')]

Unsere Lösung hat: 126 Elemente. Die Elemente von beiden unterscheiden sich um: []