Kapitel 3 – Kombinatorische Logik

Albert-Ludwigs-Universität Freiburg

Prof. Dr. Armin Biere

Institut für Informatik

Sommersemester 2024

Billigste Überdeckung der markierten Ecken

Wir suchen ein sogenanntes Minimalpolynom, das heißt ein Polynom mit minimalen Kosten.

Definition

Ein Minimalpolynom p einer Booleschen Funktion f ist ein Polynom von f mit minimalen Kosten, das heißt mit der Eigenschaft $cost(p) \leq cost(p')$ für alle Polynome p' von f.

Quine's Primimplikantensatz

Satz

Jedes Minimalpolynom p einer Booleschen Funktion f besteht ausschließlich aus Primimplikanten von f.

Beweis:

- Nehme an, dass p einen nicht primen Implikanten m von f enthält.
- m wird durch zumindest einen echten Primimplikanten m' von f überdeckt, ist also in m' enthalten.
- Es gilt demnach $cost(m') < cost(m)$.
- Ersetzt man in p den Implikanten m durch den Primimplikanten m', so erhält man ein Polynom p', das ein Polynom von f ist mit $cost(p') < cost(p)$ (also mit strikt besseren Kosten).
- **Widerspruch** dazu, dass p ein Minimalpolynom ist.

Berechnung von Implikanten

Lemma 1

Ist m ein Implikant von f, so auch $m \cdot x$ und $m \cdot x'$ für jede Variable x, die in m weder als positives, noch als negatives Literal vorkommt.

Beweis:

- $m \cdot x$ und $m \cdot x'$ sind Teilwürfel des Würfels m.
- Wenn Ecken von m markiert, dann auch Ecken von $m \cdot x$ und $m \cdot x'$.

Lemma 2

Sind $m \cdot x$ und $m \cdot x'$ Implikanten von f, so auch m.

Beweis:

- $f \geq m \cdot x + m \cdot x' = m \cdot (x + x') = m$

Charakterisierung von Implikanten

Satz

Ein Monom m ist genau dann ein Implikant von f, wenn entweder

- m ein Minterm von f ist, oder
- $m \cdot x$ und $m \cdot x'$ Implikanten von f sind für eine Variable x, die nicht in m vorkommt.

- Äquivalente Schreibweise:

  $m \in \mathit{Implikant}(f)$

  $\Leftrightarrow (m \in \mathit{Minterm}(f)) \vee (m \cdot x, m \cdot x' \in \mathit{Implikant}(f))$

- Beweis folgt unmittelbar aus Lemma 1 und Lemma 2.

Berechnung eines Minimalpolynoms

- Verfahren von Quine-McCluskey zur Berechnung aller Primimplikanten.
  - Idee: Berechne sogar *alle* Implikanten.
  - Dann ist auch klar, welche Primimplikanten sind.

- Verfahren zur Lösung des „Überdeckungsproblems".
  - Treffe unter den Primimplikanten eine geeignete Auswahl,
  - so dass die Disjunktion der ausgewählten Primimplikanten ein Polynom für f ist
  - und minimale Kosten hat.

UNI FREIBURG

Quine vs Karnaugh-Veitch

Die zwei Verfahren berechnen das selbe, aber anders: Quine ist besser für den Komputer und Karnaugh-Veitch funktioniert weil Menschen Motive erkenne.

4 Variablen

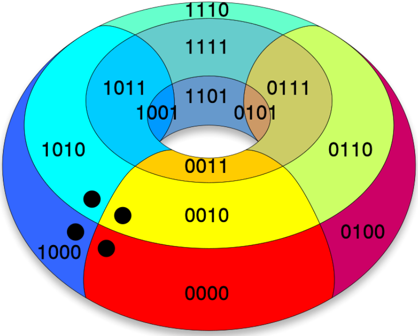
Funktion wird als Tabelle dargestellt.
Gray-Code für die Einträge: nur eine Unterscheidung von einer Zeiler zur nächsten.

	00	01	01	11	10
00					
01					
11					
10					

4 Variablen

Eigentlich Torus (wikipedia bild)

0000	0100	1100	1000
0001	0101	1101	1001
0011	0111	1111	1011
0010	0110	1110	1010



Bedeutung der Tabelle

Jede Bedeckung ist eine Konjunktion von mehreren Boxen.
Hier nur die positiven Literale:

X_3X_2 \ X_1X_0	00	01	11	10
00		X_0	X_1	
01	X_2			
11	X_3			X_2
10			X_0	X_3/X_1

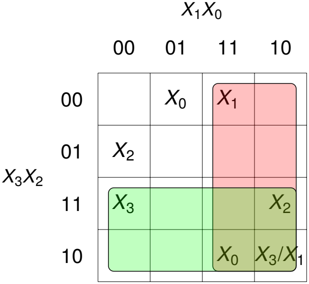
Bedeutung der Tabelle

Jede Bedeckung ist eine Konjunktion von mehreren Boxen.
Hier nur die positiven Literale:

X_3X_2 \ X_1X_0	00	01	11	10
00		X_0	X_1	
01	X_2			
11	X_3			X_2
10			X_0	X_3/X_1

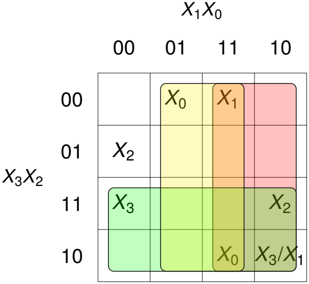
Bedeutung der Tabelle

Jede Bedeckung ist eine Konjunktion von mehreren Boxen.
Hier nur die positiven Literale:



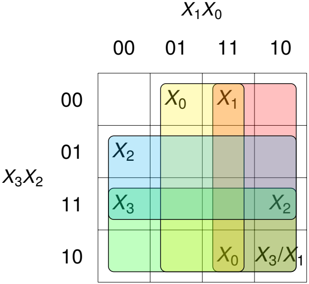
Bedeutung der Tabelle

Jede Bedeckung ist eine Konjunktion von mehreren Boxen.
Hier nur die positiven Literale:



Bedeutung der Tabelle

Jede Bedeckung ist eine Konjunktion von mehreren Boxen.
Hier nur die positiven Literale:



Ziel: Maximale Bedeckung finden

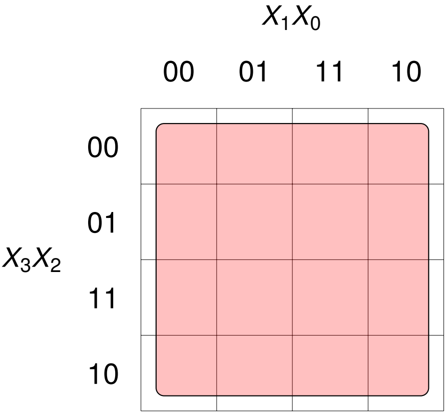
Gibt eine Bedeckung die nicht vollständig bedeckt ist:

- 4×4? monom: 0 literal
- 4×2 oder 2×4 ? monom: 1 literal
- 2×2? monom: 2 literal
- 2×1 oder 1×2? monom: 3 literal
- 1×1? monom: 4 literal

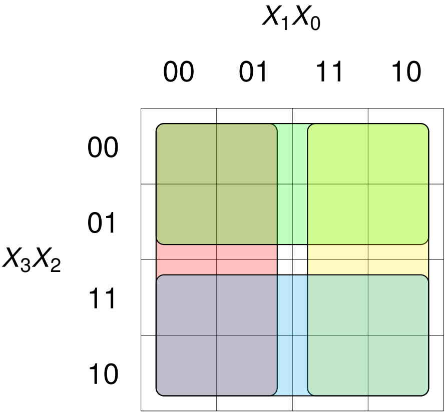
(nicht vergessen: es ist ein Torus!)

Für jedes: Monom schreiben!

Bedeckung 4×4

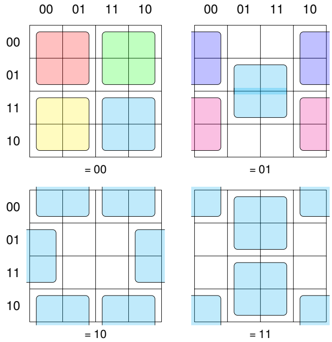


Bedeckung 4×2



Bedeckung 2×2

Es ist viel leichter zu erkennen als alle zu zeichnen:



Ziel: Maximale Bedeckung finden

x_3x_4

x_1x_2	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	1	0
10	1	1	1	0

Ziel: Maximale Bedeckung finden

x_1x_2 \ x_3x_4	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	1	0
10	1	1	1	0

Ziel: Maximale Bedeckung finden

x_1x_2 \ x_3x_4	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	1	0
10	1	1	1	0

Ziel: Maximale Bedeckung finden

x_1x_2 \ x_3x_4	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	1	0
10	1	1	1	0

Ziel: Maximale Bedeckung finden

x_1x_2 \ x_3x_4	00	01	11	10
00	1	1	0	1
01	1	1	0	1
11	1	1	1	0
10	1	1	1	0

Also: $f = \neg x_3 + \neg x_1 \neg x_4 + x_1 x_4$

Verfahren von Quine: Der Algorithmus

Prime implicants function **Quine** ($f : \mathbb{B}^n \to \mathbb{B}$)

begin
  $L_0 := Minterm(f)$;
  $i := 0$;
  $Prim(f) := \emptyset$
  while ($L_i \neq \emptyset$) **and** ($i < n$) **do**
    // L_i enthält alle Implikanten von f der Länge $n - i$.
    $L_{i+1} := \{m \mid m \cdot x$ und $m \cdot x'$ sind in L_i für ein $x\}$;
    $Prim(f) := Prim(f) \cup$
      $\{m' \mid m' \in L_i$ und m' wird von keinem $q \in L_{i+1}$ überdeckt$\}$;
    $i := i + 1$;
  end while;
  return $Prim(f) \cup L_i$;
end;

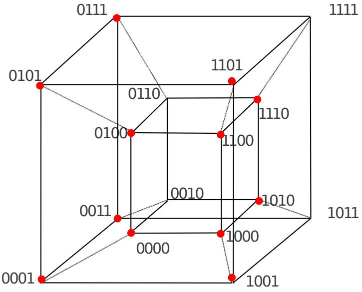
UNI FREIBURG

Verbesserung durch McCluskey

- Vergleiche nur Monome untereinander
  - welche die gleichen Variablen enthalten und
  - bei denen sich die Anzahl der positiven Literale nur um 1 unterscheidet.
- Dies wird erreicht durch:
  - Partitionierung von L_i in Klassen L_i^M, mit $M \subseteq \{x_1, \ldots, x_n\}$ und $|M| = n - i$.
  - L_i^M enthält die Implikanten aus L_i, deren Literale alle aus M sind.
  - Anordnung der Monome in L_i^M gemäß der Anzahl der positiven Literale.

UNI FREIBURG

Beispiel Quine-McCluskey



$L_0^{\{x_1,x_2,x_3,x_4\}}$:

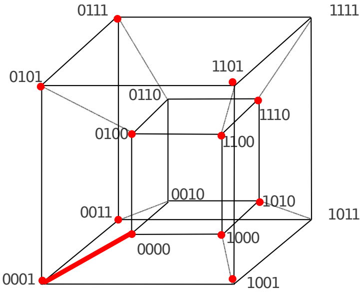
0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
0 1 1 1
1 1 0 1
1 1 1 0

steht für $x_1'x_2'x_3'x_4'$

steht für $x_1x_2'x_3'x_4$

Vergleiche im Folgenden nur Monome aus benachbarten Blöcken!

Beispiel Quine-McCluskey: Bestimmung von L_1 (1/4)



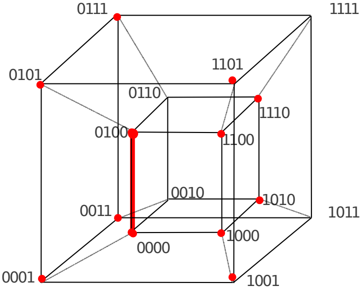
$L_0^{\{x_1,x_2,x_3,x_4\}}$:

0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
0 1 1 1
1 1 0 1
1 1 1 0

$L_1^{\{x_1,x_2,x_3\}}$:

0 0 0 -

Beispiel Quine-McCluskey: Bestimmung von L_1 (2/4)



$L_0^{\{x_1,x_2,x_3,x_4\}}$:

0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
0 1 1 1
1 1 0 1
1 1 1 0

$L_1^{\{x_1,x_2,x_3\}}$:

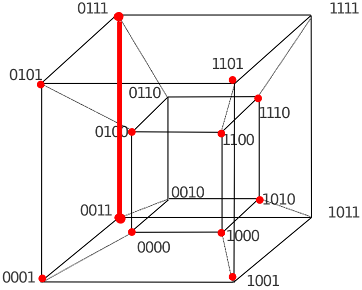
0 0 0 -

$L_1^{\{x_1,x_3,x_4\}}$:

0 - 0 0

UNI FREIBURG

Beispiel Quine-McCluskey: Bestimmung von L_1 (3/4)



$L_0^{\{x_1,x_2,x_3,x_4\}}$:

0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
➡ 0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
➡ 0 1 1 1
1 1 0 1
1 1 1 0

$L_1^{\{x_1,x_2,x_3\}}$:

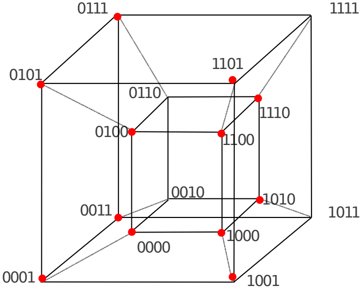
0 0 0 -

$L_1^{\{x_1,x_3,x_4\}}$:

0 - 0 0

0 - 1 1

Beispiel Quine-McCluskey: Bestimmung von L_1 (4/4)



$L_0^{\{x_1,x_2,x_3,x_4\}}$:

0 0 0 0
0 0 0 1
0 1 0 0
1 0 0 0
→ 0 0 1 1
0 1 0 1
1 0 0 1
1 0 1 0
1 1 0 0
0 1 1 1
→ 1 1 0 1
1 1 1 0

$L_1^{\{x_1,x_2,x_3\}}$:

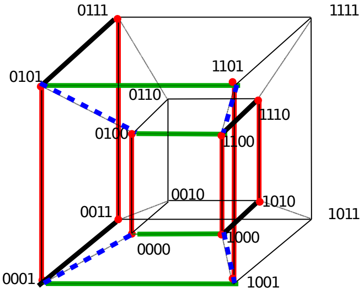
0 0 0 -

$L_1^{\{x_1,x_3,x_4\}}$:

0 - 0 0
0 - 1 1

Nicht kürzbar, da nicht Ecken der gleichen Kante.

Beispiel Quine-McCluskey: Alle bestimmten Mengen L_1



$L_1^{\{x_1,x_2,x_4\}}$:

0 0 - 1
1 0 - 0
0 1 - 1
1 1 - 0

$L_1^{\{x_1,x_2,x_3\}}$:

0 0 0 -
0 1 0 -
1 0 0 -
1 1 0 -

$L_1^{\{x_2,x_3,x_4\}}$:

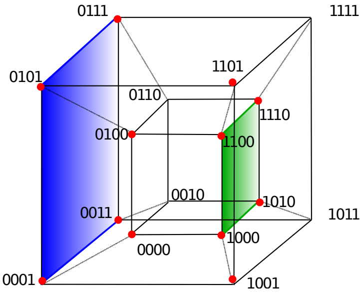
- 0 0 0
- 0 0 1
- 1 0 0
- 1 0 1

$L_1^{\{x_1,x_3,x_4\}}$:

0 - 0 0
0 - 0 1
1 - 0 0
0 - 1 1
1 - 0 1
1 - 1 0

Alle Minterme von f sind Eckpunkte von Kanten, die Implikanten sind: $Prim(f) = \emptyset$

Beispiel Quine-McCluskey: Bestimmung von L_2 (1/2)



$L_1^{\{x_1,x_2,x_4\}}$:

0 0 - 1
1 0 - 0

0 1 - 1
1 1 - 0

$L_1^{\{x_1,x_2,x_3\}}$:

0 0 0 -

0 1 0 -
1 0 0 -

1 1 0 -

$L_1^{\{x_2,x_3,x_4\}}$:

- 0 0 0

- 0 0 1
- 1 0 0

- 1 0 1

$L_1^{\{x_1,x_3,x_4\}}$:

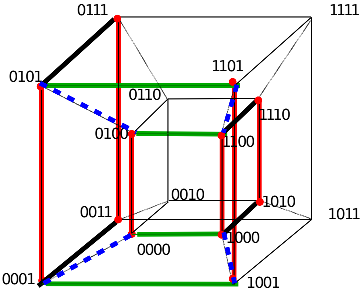
0 - 0 0

0 - 0 1
1 - 0 0

0 - 1 1
1 - 0 1
1 - 1 0

Alle Implikanten aus $L_1^{\{x_1,x_2,x_4\}}$ sind Kanten von Flächen, die Implikanten sind: $Prim(f) = \emptyset$

Beispiel Quine-McCluskey: Bestimmung von L_2 (2/2)



$L_1^{\{x_1,x_2,x_4\}}$:

0 0 - 1
1 0 - 0
0 1 - 1
1 1 - 0

$L_1^{\{x_1,x_2,x_3\}}$:

0 0 0 -
0 1 0 -
1 0 0 -
1 1 0 -

$L_1^{\{x_2,x_3,x_4\}}$:

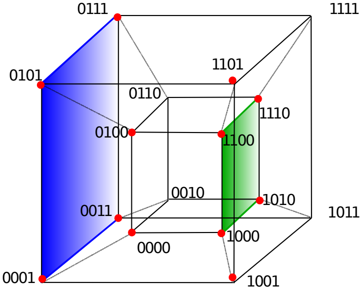
- 0 0 0
- 0 0 1
- 1 0 0
- 1 0 1

$L_1^{\{x_1,x_3,x_4\}}$:

0 - 0 0
0 - 0 1
1 - 0 0
0 - 1 1
1 - 0 1
1 - 1 0

Alle Implikanten aus $L_1{}^M$ sind Kanten von Flächen, die Implikanten sind: $Prim(f) = \emptyset$

Beispiel Quine-McCluskey: Bestimmung von L_3 (1/2)



$L_2^{\{x_1,x_2\}}$:

$L_2^{\{x_1,x_4\}}$:

0 - - 1
1 - - 0

$L_2^{\{x_2,x_4\}}$:

$L_2^{\{x_1,x_3\}}$:

0 - 0 -
1 - 0 -

$L_2^{\{x_2,x_3\}}$:

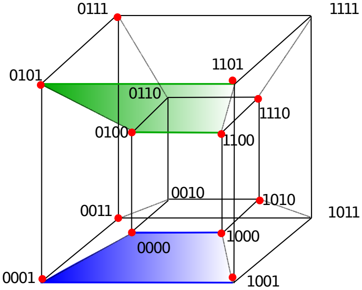
- 0 0 -
- 1 0 -

$L_2^{\{x_3,x_4\}}$:

- - 0 0
- - 0 1

Die markierten Implikanten-Flächen sind nicht Rand eines 3-dim. Implikanten. Sie sind also prim! $\Rightarrow$ $Prim(f) = \{x_1'x_4, x_1x_4'\}$

Beispiel Quine-McCluskey: Bestimmung von L_3 (2/2)



$L_2^{\{x_1,x_2\}}$:

$L_2^{\{x_1,x_4\}}$:

0 - - 1
1 - - 0

$L_2^{\{x_2,x_4\}}$:

$L_2^{\{x_1,x_3\}}$:

0 - 0 -
1 - 0 -

$L_2^{\{x_2,x_3\}}$:

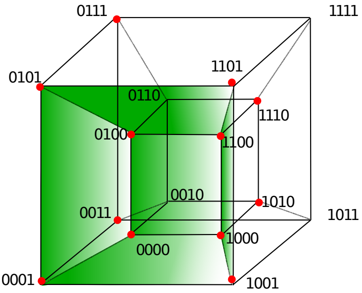
- 0 0 -
- 1 0 -

$L_2^{\{x_3,x_4\}}$:

- - 0 0
- - 0 1

Die markierten Implikanten-Flächen sind Rand eines 3-dimensionalen Implikanten. Sie sind also nicht prim! $\Rightarrow$ $Prim(f) = \{x_1' x_4, x_1 x_4'\}$

Beispiel Quine-McCluskey: Ende



$L_3^{\{x_1\}}$:

$L_3^{\{x_2\}}$:

$L_3^{\{x_3\}}$:

- - 0 -

$L_3^{\{x_4\}}$:

$Prim(f) = \{x_1' x_4, x_1 x_4'\}$

$\Rightarrow Prim(f) = \{x_1' x_4, x_1 x_4', x_3'\}$

$p_{complete}(f) = x_1' x_4 + x_1 x_4' + x_3'$

Korrektheit von Quine-McCluskey (1/2)

Prime implicants function **Quine** ($f : \mathbb{B}^n \to \mathbb{B}$)

begin
  $L_0 := Minterm(f)$;
  $i := 0$;
  $Prim(f) := \varnothing$
  while ($L_i \neq \varnothing$) **and** ($i < n$) **do**
    // L_i *enthält alle Implikanten von* f *der Länge* $n - i$.
    $L_{i+1} := \{m \mid m \cdot x \text{ und } m \cdot x' \text{ sind in } L_i \text{ für ein } x\}$;
    $Prim(f) := Prim(f) \cup$
      $\{m' \mid m' \in L_i \text{ und } m' \text{ wird von keinem } q \in L_{i+1} \text{ überdeckt}\}$;
    $i := i + 1$;
  end while;
  return $Prim(f) \cup L_i$;
end;

UNI FREIBURG

Korrektheit von Quine-McCluskey (2/2)

Satz

Für alle $i = 0, 1, \ldots, n$ gilt:

- L_i enthält nur Monome mit $n - i$ Literalen.
- L_i enthält genau die Implikanten von f mit $n - i$ Literalen.
- Nach Iteration i enthält $\mathit{Prim}(f)$ genau die Primimplikanten von f mit mindestens $n - i$ Literalen.

Beweis:
Induktion über i.

- Abbruchbedingung $(L_i = \emptyset)$ oder $(i = n)$:
- $L_i = \emptyset$ bedeutet, dass keine Implikanten bei der "Partnersuche" entstanden sind, d.h. L_{i-1} ist vollständig in $\mathit{Prim}(f)$ aufgegangen.
- $i = n$ bedeutet, dass L_n berechnet wurde, es gilt dann $L_n = \emptyset$ oder $L_n = \{1\}$, letzteres bedeutet f ist die Eins-Funktion und $\mathit{Prim}(f) = \{1\}$.

UNI FREIBURG

Kosten des Verfahrens

Lemma

Es gibt 3^n verschiedene Monome in n Variablen.

Beweis:
Für jedes Monom m und jede der n Variablen x liegt genau eine der drei folgenden Situationen vor:

- m enthält weder das positive noch das negative Literal von x.
- m enthält das positive Literal x.
- m enthält das negative Literal x'.

Jedes Monom ist durch diese Beschreibung auch eindeutig bestimmt.

UNI FREIBURG

Komplexität des Verfahrens von Quine-McCluskey

Satz

Die Laufzeit des Verfahrens liegt in $O(n^2 \cdot 3^n)$ beziehungsweise in $O(\log^2(N) \cdot N^{\log(3)})$, wobei $N = 2^n$ die Größe der Funktionstabelle ist.

Beweisidee:
Jedes der 3^n Monome wird im Verlauf des Verfahrens mit höchstens n anderen Monomen verglichen.

- Gegeben sei ein Monom mx. Die Erzeugung von mx' und die Suche nach mx' in L_i ist bei Verwendung geeigneter Datenstrukturen in $O(n)$ durchführbar.

$O(n^2 \cdot 3^n) = O(\log^2(N) \cdot N^{\log(3)})$ durch Nachrechnen:

$$3^n = (2^{\log_2(3)})^n = (2^n)^{\log_2(3)} = N^{\log_2(3)}$$

UNI FREIBURG