Minimierung von logischen Schaltungen

WAS SIND LOGISCHE SCHALTUNGEN

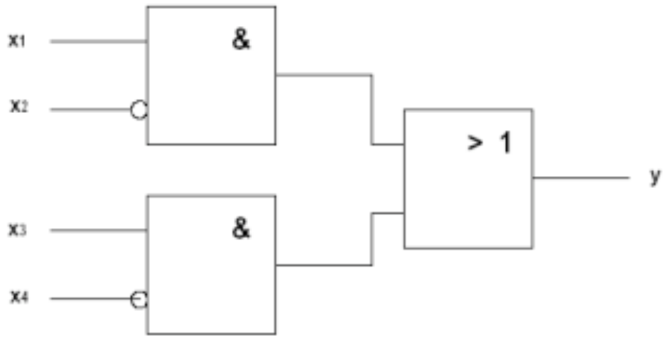
Logische Verknüpfungszeichen:

& = Logisches Und-Verknüpfung (Konjunktion).
V = Logische Oder-Verknüpfung (Disjunktion).
- = Nicht (Negation).

Logische Funktionen:

y = (-a & -b & -c) V (-a & -b & c) V (-a & b & -c)

Logische Schaltung:



Warum logische Funktionen/Schaltungen minimieren:

Die hardwaretechnische Realisierung wird einfacher und daher kostengünstiger.

ZWEI VERFAHREN ZUR MINIMIERUNG VON LOGISCHEN SCHALTUNGEN

1. Quine und McCluskey - Verfahren

2. Karnaugh und Veitch - Verfahren

VERFAHREN NACH KARNAUGH-VEITCH

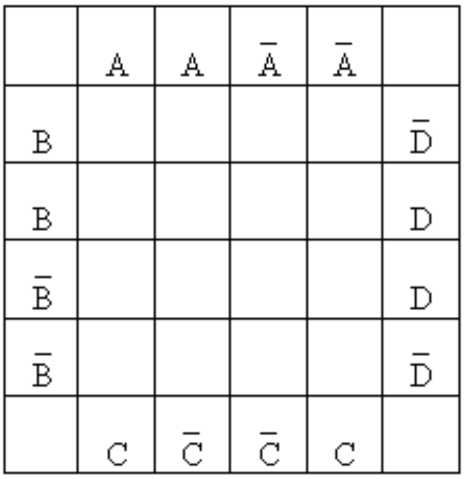
Erläuterung

Das Karnaugh-Veitch-Diagramm, kurz KV-Diagramm, dient der Vereinfachung Boolescher Funktionen. Es wurde 1952 von Edward W. Veitch entworfen und 1953 von Maurice Karnaugh zu seiner heutigen Form weiterentwickelt.

Mittels eines KV-Diagramms lässt sich jede beliebige boolesche Funktion in eine disjunktive oder konjunktive Minimalform umwandeln.

Grundprinzip

- Ein KV-Diagramm für n Eingangsvariablen hat 2n Felder.
- Das KV-Diagramm wird mit den Variablen an den Rändern beschriftet.
- Jede Variable in negierter und nicht-negierter Form vor.
- Die Zuordnung der Variablen zu den einzelnen Feldern kann beliebig erfolgen.
- Horizontal und vertikal benachbarte Felder müssen sich in genau einer Variablen unterscheiden.
- Es wird in die einzelnen Felder eine 1 eingetragen, wenn ein Minterm der Funktion vorliegt.

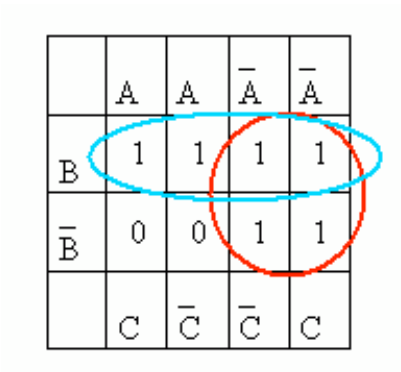


Vorgehen

Gegeben ist folgende logische Funktion:

$$l = \bar{A}\bar{B}\bar{C} \vee \bar{A}\bar{B}C \vee \bar{A}B\bar{C} \vee \bar{A}BC \vee AB\bar{C} \vee ABC$$

Die jeweiligen Konjunktionen tragen wir in dem Diagramm mit einer 1 ein,
und markieren Blöcke die man zusammenfassen kann. Es können immer nur Blöcke zusammengefasst werden, deren Anzahl einer 2er-Potenz entspricht:



Durch ablesen der jeweiligen Blöcke kommen wir nun zu der minimierten Funktion:

$$l = \bar{A} \vee B$$

Literatur

Maurice Karnaugh übernahm von 1952 bis 1966 als Professor für experimentelle Forschungen
eine leitende Position bei den Bell Laboratories in Murray Hills (New Jersey).
Seine wissenschaftliche Arbeit galt speziell dem Gebiet der Heuristik. Karnaugh entwickelte neue Techniken und Methoden für den schnelleren Entwurf informationstechnischer Systeme.
Das von ihm dafür entwickelte und nach ihm als Karnaugh-Diagramm (Karnaugh-Veitch-Diagramm, KV-Diagramm)
benannte Verfahren beschrieb er erstmals 1953 in der Fachzeitschrift „Communications and Electronics".

Primärliteratur:

Edward W. Veitch: A chart method for symplifying truth functions, Mai 1952,
Proc. Assoc. for Computing Machinery, Pittsburgh.

Maurice Karnaugh: The Map Method for Synthesis of Combinational Logic Circuits,
Transactions of the AIEE, Vol. 72, No. 9 (1953), 593—599.

Sekundärliteratur:

Schöning, U.: Logik für Informatiker, 1992
Heribert Koch: Theoretische Informatik, 2002
Erwin Holland-Moritz: Theoretische Informatik, 2002

IMPLEMENTIERUNG KARNAUGH-VEITCH

Ansatz:

Gegeben ist folgende logische Funktion:

$$l = \bar{A}\bar{B}\bar{C} \vee \bar{A}\bar{B}C \vee \bar{A}B\bar{C} \vee \bar{A}BC \vee AB\bar{C} \vee ABC$$

Das dazugehörige KV-Diagramm:

	A	**A**	**-A**	**-A**
B	1	1	1	1
-B	0	0	1	1
	C	**-C**	**-C**	**C**

(Zur Erinnerung: Es werden
alle Konjunktionen
mit einer 1 eingetragen)

Vorgehen:

<u>2 2-Dimensionale Arrays:</u>

1. Array:

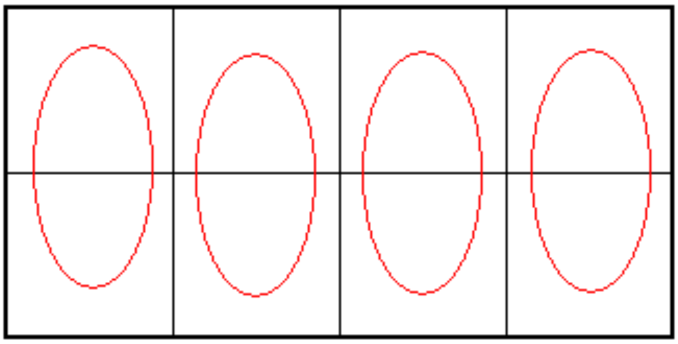
-	**0**	**1**	**2**	**3**
0	A B C	A B -C	-A B -C	-A B C
1	A -B C	A -B -C	-A -B -C	-A -B C

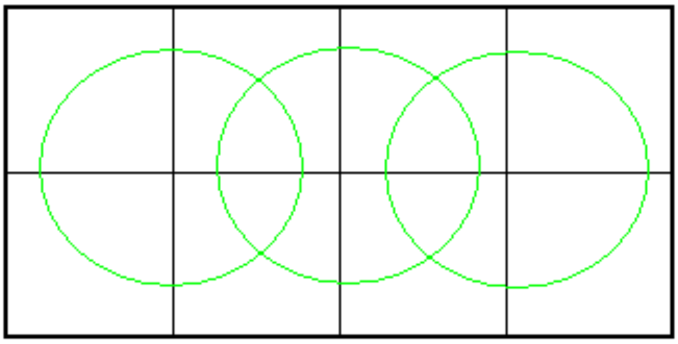
2. Array:

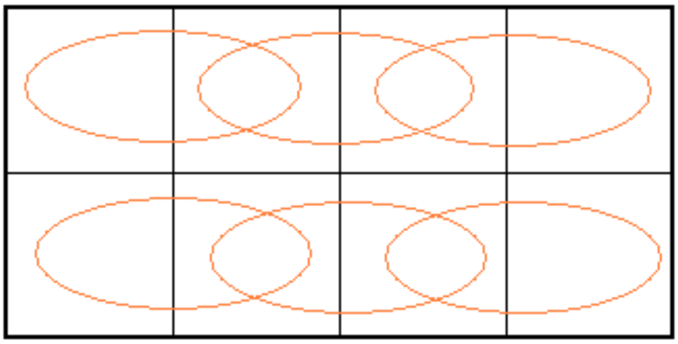
-	**0**	**1**	**2**	**3**
0	1	1	1	1
1	0	0	1	1

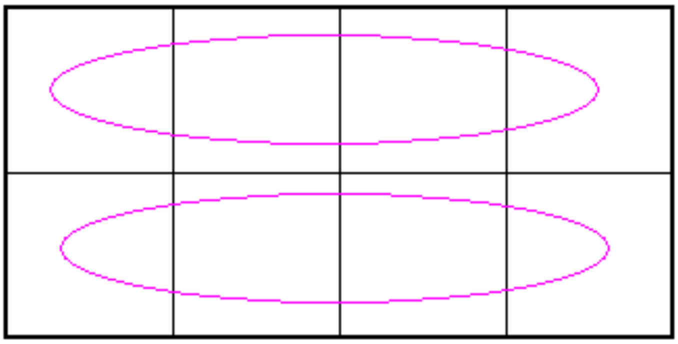
Alle Möglichkeiten abfragen:

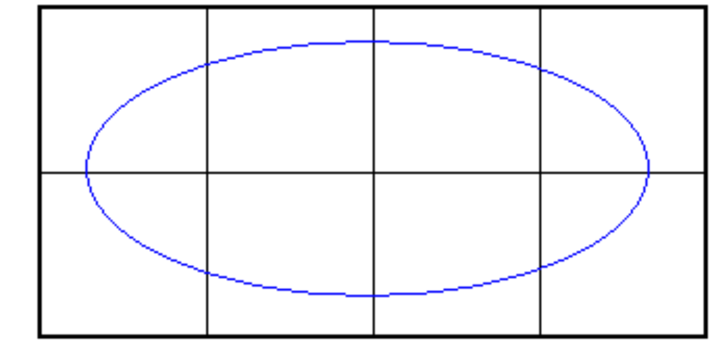
Es werden immer nur 2er-Potenzen zusammengefasst:

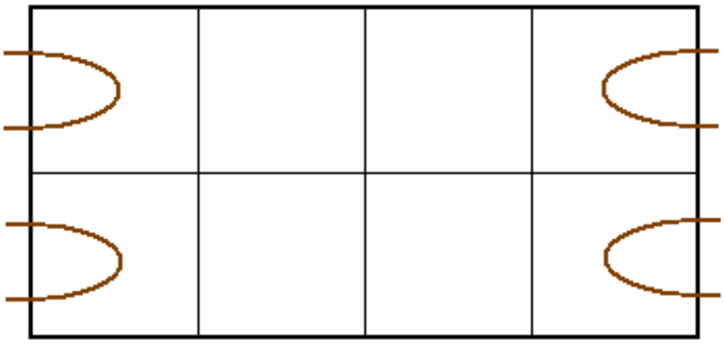


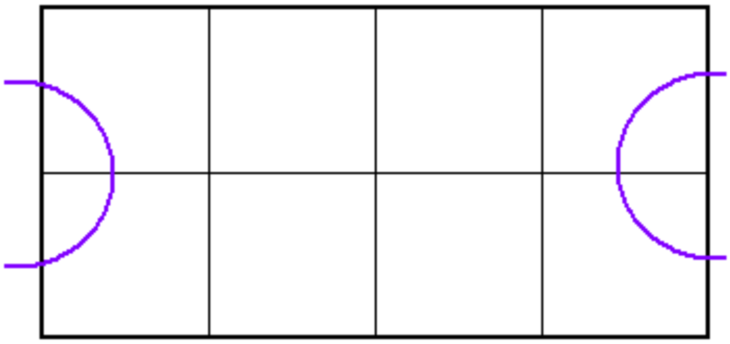












Ergebnis:

	A	A	$\bar{A}$	$\bar{A}$
B	1	1	1	1
$\bar{B}$	0	0	1	1
	C	$\bar{C}$	$\bar{C}$	C

$$l = \bar{A} \vee B$$

VERFAHREN NACH QUINE-MCCLUSKEY

Erläuterung

- Von Willard Van Orman Quine und Edward J. McCluskey.
- Methode, um Boolesche Funktionen zu minimieren.
- Der Kern des Verfahrens wurde bereits von Quine vollständig beschrieben.
- Verfeinerungen von McCluskey: Praktische algorithmische Durchführbarkeit.
- Das Verfahren findet immer eine minimale Lösung.

Grundprinzip

Unterscheiden sich zwei durch Disjunktion verknüpfte Konjunktionsterme nur durch die Negation einer einzigen Variablen, so kann man diese beiden Terme verschmelzen und dabei die betreffende Variable entfernen.

(a & b & c & -d) V (a & b & -c & -d)

ergibt

<u>a & b & -d</u>

Vorgehen

1. Die Funktion muss in disjunktiver Normalform dargestellt sein:

```
f(a,b,c,d) = ( a &  b &  c & ¬d)  V
             ( a &  b & ¬c &  d)  V
             ( a &  b & ¬c & ¬d)  V
             ( a & ¬b &  c & ¬d)  V
             ( a & ¬b & ¬c & ¬d)  V
             (¬a & ¬b &  c &  d)  V
             (¬a & ¬b &  c & ¬d)  .
```

d. h., jede einzelne Variable muss in jedem Term min. ein mal negiert oder nicht negiert auftreten.

2. Zusammenfassen von Termen:

Alle Terme werden jetzt nach aufsteigender Klasse sortiert, in einer Tabelle aufgelistet.
Die Klasse einer Konjunktion ist die Anzahl der darin vorkommenden nicht negierten Variablen.

Zeile	Gruppe	Vollkonjunktion
1	1	K_1 := a & b & c & ¬d
2		K_2 := a & b & ¬c & d
3	2	K_3 := a & b & ¬c & ¬d
4		K_4 := a & ¬b & c & ¬d
5		K_5 :=¬a & ¬b & c & d
6	3	K_6 := a & ¬b & ¬c & ¬d
7		K_7 :=¬a & ¬b & c & ¬d

Man vergleicht nun alle Terme benachbarter Klassen daraufhin, ob sie sich paarweise um eine einzige Negation unterscheiden. Ist dies der Fall, so verschmilzt man die beiden Terme zu einem neuen Term und trägt ihn in eine zweite Tabelle ein:

Zeile				
1 (1,3)	a	b		¬d
2 (4,6)	a	¬b		¬d
3 (1,4)	a		c	¬d
4 (3,6)	a		¬c	¬d
5 (2,3)	a	b	¬c	
6 (5,7)	¬a	¬b	c	
7 (4,7)		¬b	c	¬d

Auf die verschmolzenen Terme wendet man das Verfahren rekursiv in mehreren Stufen so lange an, bis keine weiteren Verschmelzungen mehr möglich sind.
Es ist darauf zu achten, dass beide Terme die gleichen Variablen enthalten

Zeile				
1 (1,2)	a			¬d
2 (3,4)	a			¬d
3 (5)	a	b	¬c	
4 (6)	¬a	¬b	c	
5 (7)		¬b	c	¬d

Ab der zweiten Minimierungsstufe tauchen die gleichen Konjunktionen mehrfach auf,
sie sind jedoch nur einmal zu berücksichtigen.

Während dieses Vorganges verbleiben einige Terme, die nicht verschmolzen werden können,
diese Terme bezeichnet man als Primterme

$R_1 = a \;\&\; \neg d$, $R_2 = a \;\&\; b \;\&\; \neg c$ $R_3 = \neg a \;\&\; \neg b \;\&\; c$, $R_4 = \neg b \;\&\; c \;\&\; \neg d$,

3. Erstellen der Primtermtabelle:

Es ist durchaus möglich, dass nicht alle Primterme benötigt werden. Um herauszufinden,
welche Primterme man unbedingt benötigt, erstellt man eine sogenannte Primtermtabelle:

Man trägt nun immer dann eine Markierung in der jeweiligen Zelle ein,
wenn der Minterm Element des Primterms ist:

	K_1	K_2	K_3	K_4	K_5	K_6	K_7
R_1	*		*	*		*	
R_2		*	*				
R_3					*		*
R_4				*			*

4. Spaltendominanzprüfung:

Die Spalten werden paarweise darauf verglichen, ob nicht eine Spalte existiert, in denen die markierten Primterme eine Teilmenge der markierten Primterme der anderen Spalte sind. Ist dies der Fall, so kann die Spalte mit der Obermenge gestrichen werden, denn es müssen alle Konjunktionen erfasst werden und daher ist die Konjunktion mit der Obermenge durch Auswahl der Konjunktion mit der Teilmenge ebenfalls erfasst.

	K_1	K_2	K_3	K_4	K_5	K_6	K_7
R_1	*		*	*		*	
R_2		*	*				
R_3					*		*
$\cancel{R_4}$				*			*

R1 muss in die Lösung aufgenommen werden, da nur hiermit die Vollkonjunktionen K1 und K6 ersetzt werden können; R2 überdeckt allein K2 und R3 ersetzt K5 und K7. R4 wird für die Lösung nicht benötigt, da K4 und K7 bereits abgedeckt sind.

Als Ergebnis erhalten wir die minimierte Funktion:

f(a,b,c,d) = R1 V R2 V R3

dies ergibt

(a & -d) V (a & b & -c) V (-a & -b & c)

Literatur

Als junger Wissenschaftler beschäftigte sich Quine hauptsächlich mit mathematischer Logik, wandte sich aber in den späten 1940er Jahren immer mehr Themen der Erkenntnistheorie zu. Seine hervorragenste Leistung auf dem Gebiet der formalen Logik ist die zusammen mit Edward J. McCluskey durchgeführte Entwicklung eines Algorithmus zur Minimierung boolscher Funktionen, der heute unter dem Namen Quine-McCluskey-Verfahren bekannt ist.

Primärliteratur:

(Quine, Willard van Orman.: Grundzüge der Logik - Frankfurt a.M. : Suhrkamp, 1969)

Sekundärliteratur:

Schöning, U.: Logik für Informatiker, 1992
Ehrig, H. et al.: Mathematisch-strukturelle Gnundlagen der Ingormatik, 1999
Heribert Koch: Theoretische Informatik, 2002

DIE LAUFZEIT DES QUINE-MCCLUSKEY-ALGORITHMUS:

Ratio-Test:

Mit dem Ratio-Test wird überprüft, ob eine geschätze Laufzeit der tatsächlichen Laufzeit entspricht.
Dazu setzt man **t(n)** mit **f(n)** in Relation.

n -> Die Anzahl der Terme, der boolschen Funktion
t(n) -> Anzahl der durchgeführten Operationen
f(n) -> Die geschätzte Laufzeit
<u>r(n) -> t(n) / f(n)</u>

Die folgende Tabelle zeigt die Ergebnisse der Schätzungen für den Quine-McCluskey-Algorithmus.
Schätzungen für den Algorithmus: n, n^2, n^3 und log(n).

Ratio-Test

n	Terme
t(n)	Operationen
f(n)	geschätzte Laufzeit
r(n)	t(n) / f(n)

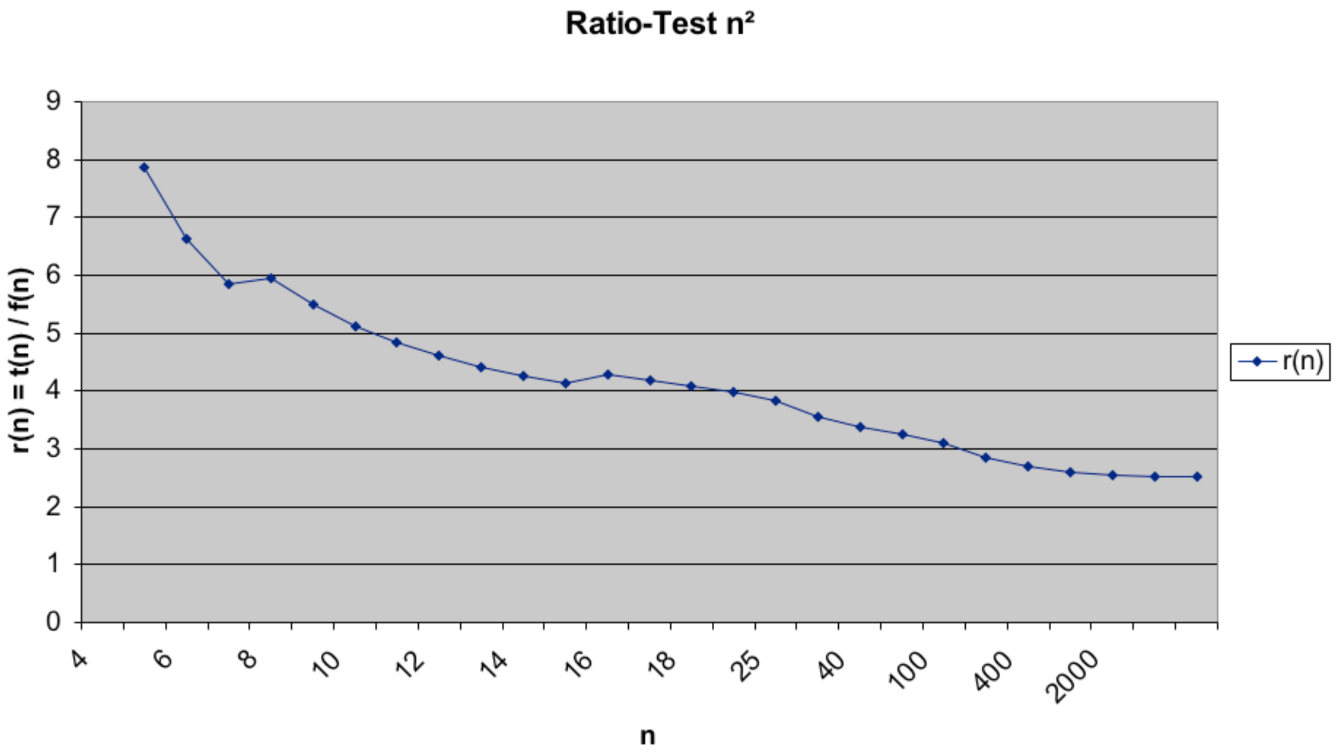
		n^2		n		n^3		log(n)	
n	t(n)	f(n)	r(n)	f(n)	r(n)	f(n)	r(n)	f(n)	r(n)
4	126	16	7,875	4	31,5	64	1,96875	0,602059991	209,28147
5	166	25	6,64	5	33,2	125	1,328	0,698970004	237,4923086
6	211	36	5,861111111	6	35,16666667	216	0,976851852	0,77815125	271,1555111
7	292	49	5,959183673	7	41,71428571	343	0,851311953	0,84509804	345,5220414
8	351	64	5,484375	8	43,875	512	0,685546875	0,903089987	388,6655871
9	415	81	5,12345679	9	46,11111111	729	0,569272977	0,954242509	434,8999294
10	484	100	4,84	10	48,4	1000	0,484	1	484
11	558	121	4,611570248	11	50,72727273	1331	0,419233659	1,041392685	535,8209328
12	637	144	4,423611111	12	53,08333333	1728	0,368634259	1,079181246	590,2622959
13	721	169	4,266272189	13	55,46153846	2197	0,328174784	1,113943352	647,2501483
14	810	196	4,132653061	14	57,85714286	2744	0,295189504	1,146128036	706,7273243
15	967	225	4,297777778	15	64,46666667	3375	0,286518519	1,176091259	822,2151067
16	1070	256	4,1796875	16	66,875	4096	0,261230469	1,204119983	888,6157654
17	1178	289	4,076124567	17	69,29411765	4913	0,239772033	1,230448921	957,3741579
18	1291	324	3,984567901	18	71,72222222	5832	0,221364883	1,255272505	1028,461943
20	1532	400	3,83	20	76,6	8000	0,1915	1,301029996	1177,528577
25	2222	625	3,5552	25	88,88	15625	0,142208	1,397940009	1589,481656
30	3037	900	3,374444444	30	101,2333333	27000	0,112481481	1,477121255	2056,0262
40	5205	1600	3,253125	40	130,125	64000	0,081328125	1,602059991	3248,942005
50	7750	2500	3,1	50	155	125000	0,062	1,698970004	4561,587303
100	28378	10000	2,8378	100	283,78	1000000	0,028378	2	14189
200	107531	40000	2,688275	200	537,655	8000000	0,013441375	2,301029996	46731,68112
400	416634	160000	2,6039625	400	1041,585	64000000	0,006509906	2,602059991	160116,9848
800	1636437	640000	2,556932813	800	2045,54625	512000000	0,003196166	2,903089987	563688,0039
2000	10099040	4000000	2,52476	2000	5049,52	8000000000	0,00126238	3,301029996	3059360,264
3000	22660543	9000000	2,517838111	3000	7553,514333	27000000000	0,000839279	3,477121255	6517041,351

Wie man an der Tabelle erkennen kann, sind die Schätzungen für n, n^3 und log(n) schlecht, denn

n tendiert gegen unendlich -> Zu klein geschätzt.
n^3 tendiert gegen null -> Zu hoch geschätzt.
log(n) tendiert gegen unendlich -> Zu klein geschätzt.

<u>Da sich n^2 auf eine Konstante > 0 einpendelt, ist anzunehmen, dass diese Schätzung richtig ist.</u>

Diagramm für f(n) = n^2:


Ratio-Test n²
r(n) = t(n) / f(n)
9
8
7
6
5
4
3
2
1
0
4
6
8
10
12
14
16
18
25
40
100
400
2000
n
r(n)