(b) zu zeigen: $SAT \notin TISP(n^{1,1}, n^{0,1})$

Beweisskizze:

Angenommen, $SAT \in TISP(n^{1,1}, n^{0,1})$.
Wir wollen zeigen, dass dann $NTIME(n) \subseteq TISP(n^{1,2}, n^{0,2})$ ist
— was im Widerspruch zu (a) steht.

Sei dazu $L \in NTIME(n)$. zu zeigen: $L \in TISP(n^{1,2}, n^{0,2})$.

Idee:
- Sei $M$ eine det. $n^{1,1}$-Zeit- und $n^{0,1}$-Platz beschränkte TM, die SAT entscheidet.
- Sei $f$ eine geeignete Reduktion von $L$ auf SAT.
- Sei $M'$ eine det. TM, die bei Eingabe von $x \in \{0,1\}^*$ die Berechnung von $M$ bei Eingabe $\varphi_x := f(x)$ simuliert
- Idee: $M'$ entscheidet $L$
  Ziel: Führe die Details in der Konstruktion von $M'$ so aus, dass $M'$ $n^{1,2}$-Zeit- und $n^{0,2}$-platzbeschränkt ist

Einige Details:

- zur "geeigneten Reduktion $f$":
  1) zu $L \in NTIME(n)$ gibt es eine stereotype 2-Band-NDTM $N$ die $L$ in Zeit $O(n \cdot \log n)$ entscheidet (Hennie und Stearns, 1966)
  2) Unter Betrachtung von $N$ erhält man für jedes $n \in \mathbb{N}$ eine CNF-Formel $\varphi_n$ s.d. gilt:
     - $\varphi_n$ hat Länge $\leq O(n \cdot (\log n)^2)$
     - $\varphi_n$ benutzt Variablen $x_1, \dots, x_n$ und weitere Variablen $y_1, y_2, \dots$
     - Ist für $x \in \{0,1\}^n$ $\varphi_x$ die Formel, die aus $\varphi_n$ entsteht, indem die Variablen $x_1, \dots, x_n$ mit den Werten $x_1, \dots, x_n$ belegt werden, so gilt: $x \in L \Leftrightarrow \varphi_x$ ist erfüllbar.

     Und es gibt eine Zahl $r \in \mathbb{N}$ und einen Algorithmus $A$, der bei Eingabe von $n$ und $j$ das $j$-te Bit der Repräsentation von $\varphi_n$ in Zeit und auf Platz $(\log n)^r$ berechnet

$\Rightarrow$ Es gibt einen Algorithmus $A'$, der bei Eingabe von $x \in \{0,1\}^n$ und $j$ das $j$-te Bit der Repräsentation von $\varphi_x$ in Zeit und auf Platz $(\log n)^r$ berech.

Beachte: $A'$ benötigt dazu "random access" auf die Bits von $x$ – d.h. bei Angabe von $i$ kann $A'$ in einem Schritt auf $x_i$ zugreifen.

Dieses Berechnungsmodell wird durch so genannte Random-Access-Turingmaschinen modelliert.

Die Aussage von Teil (a) gilt auch, wenn die Klasse $TISP(n^{1,2}, n^{0,2})$ bezüglich dieser Random-Access-Turingmaschinen definiert wird

- Zur Konstruktion von $M'$:

Ähnlich wie beim Nachweis der Transitivität von $\leq_e$ (logspace-Reduktionen, vgl. Kapitel 4):

$M'$ schreibt $\varphi_x$ auf ein "virtuelles Eingabeband", merkt sich die aktuelle Kopfposition und nutzt in jedem Schritt, mit dem sie $M$ bei Eingabe $\varphi_x$ simuliert, den Algorithmus $A'$ um das aktuell von $\varphi_x$ zu lesende Bit zu ermitteln.

$M'$ ist somit eine det. Random-Access-Turingmaschine, die $L$ entscheidet und bei Eingabe von $x \in \{0,1\}^n$ höchstens

- $|\varphi_x|^{1,1} \cdot (\log n)^r$ Berechnungsschritte durchführt

$$\leq \left(c \cdot n \cdot (\log n)^2\right)^{1,1} \cdot (\log n)^r \leq \tilde{c} \cdot n^{1,1} \cdot (\log n)^{2,2+r} \leq \tilde{c} \cdot n^{1,}$$

und

- $|\varphi_x|^{0,1} + (\log n)^r$ Platz benutzt

$$\leq \left(c \cdot n \cdot (\log n)^2\right)^{0,1} + (\log n)^r \leq \tilde{c} \cdot n^{0,1} \cdot (\log n)^{0,2} + (\log n)^r \leq \tilde{c} \cdot n^{0,}$$

(für geeignete Konstanten $c, \tilde{c} \in \mathbb{N}$).

Somit gehört $L$ zur Variante der Klasse $TISP(n^{1,2}, n^{0,2})$, die mittels Random-Access-TMs definiert ist. Der Widerspruch folgt mit der entsprechenden Variante von (9). □

Details zum Beweis von Theorem 5.22 finden sich in dem Artikel

"Time-Space Lower Bounds for Satisfiability"
von L. Fortnow, R. Lipton, D. van Melkebeek, A. Viglas.
Journal of the ACM, 52: 835-865, 2005.

# Kapitel 6: Boolesche Schaltkreise

**Boolesche Schaltkreise:** Modell für
- einfache Computerchips
- paralleles Rechnen
- nicht-uniform: Für jede Eingabelänge $n$ ein Algorithmus $A_n$ (der ganz anders vorgehen kann als $A_{n-1}$)

## 6.1 Boolesche Schaltkreise: Definition

**Definition 6.1**

Sei $n \in \mathbb{N}$.

(a) Ein Boolescher Schaltkreis (kurz: SK) mit $n$ Eingabegattern und einem Ausgabegatter ist ein endlicher gerichteter azyklischer Graph, für den gilt:
- Es gibt $n$ Knoten vom Ein-Grad 0, die mit den Symbolen $x_1, \dots, x_n$ markiert sind — die so genannten Eingabegatter.
- Es gibt einen Knoten vom Aus-Grad 0, das so genannte Ausgabegatter.
- Jeder Knoten hat Ein-Grad 0, 1 oder 2.
  - Jeder Knoten vom Ein-Grad 0, der kein Eingabegatter ist, ist mit einer der beiden Booleschen Konstanten **0** oder **1** markiert (und wird ⓪-Gatter bzw. ①-Gatter genannt).

- Jeder Knoten vom Ein-Grad 1 ist mit dem Symbol $\neg$ markiert (und wird $\neg$-Gatter genannt)
- Jeder Knoten vom Ein-Grad 2 ist mit einem der Symbole $\wedge$ bzw. $\vee$ markiert (und wird $\wedge$-Gatter bzw. $\vee$-Gatter genannt).

(b) Die Größe $|C|$ eines Sks $C$ ist die Anzahl der Knoten von $C$.

(c) Ist $C$ ein Sk mit $n$ Eingabegattern und ist $x \in \{0,1\}^n$, so ist die Ausgabe $C(x) \in \{0,1\}$ wie folgt definiert:
$C(x) = val_{v_{out}}(x)$, für das Ausgabegatter $v_{out}$, wobei $val_v(x)$ für jeden Knoten $v$ von $C$ wie folgt definiert ist:

- Ist $v$ ein mit $x_i$ markiertes Eingabegatter, so ist $val_v(x) = x_i$
- Ist $v$ ein $0$-Gatter (bzw. $1$-Gatter), so ist $val_v(x) = 0$ (bzw. 1)
- Ist $v$ ein $\neg$-Gatter, so ist
$$val_v(x) = \begin{cases} 1, & \text{falls } val_u(x) = 0 \\ 0, & \text{falls } val_u(x) = 1, \end{cases}$$
wobei $u$ der Knoten in $C$ ist, von dem aus eine Kante nach $v$ führt ($u$ wird der Vorgängerknoten von $v$ genannt).
- Ist $v$ ein $\wedge$-Gatter, so ist

$$val_v(x) = \begin{cases} 1 & \text{falls } val_{u_1}(x) = 1 = val_{u_2}(x) \\ 0 & \text{sonst} \end{cases},$$

wobei $u_1$ und $u_2$ die beiden Vorgängerknoten von $v$ in $C$ sind.

- Ist $v$ ein $\vee$-Gatter, so ist

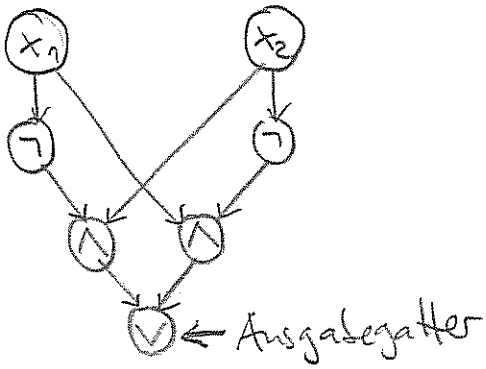
$$val_v(x) = \begin{cases} 0 & \text{falls } val_{u_1}(x) = 0 = val_{u_2}(x) \\ 1 & \text{sonst (d.h. } val_{u_1}(x) = 1 \text{ oder } val_{u_2}(x) = 1) \end{cases}$$

wobei $u_1$ und $u_2$ die beiden Vorgängerknoten von $v$ in $C$ sind.

(d) Ein SK $C$ mit $n$ Eingabegattern berechnet die Boolesche Funktion $f_C: \{0,1\}^n \to \{0,1\}$ mit $f_C(x) := C(x)$ f.a. $x \in \{0,1\}^n$

Beispiel 6.2

a) Der SK




berechnet die 2-stellige XOR-Funktion (mit $XOR(x_1,x_2) = 1 \iff x_1 \neq x_2$).

(b) Der Syntaxbaum einer aussagenlogischen Formel kann als SK aufgefasst werden, bei dem jeder Knoten, der kein Eingabegatter ist, den Aus-Grad $\leq 1$ hat.

## Notation 6.3

(a) In Zeichnungen von Skeen werden oft die Richtungen von Kanten oft weggelassen — Kanten verlaufen dann immer von oben nach unten

(b) Im Zusammenhang mit Skeen werden die Begriffe "Ein-Grad" und "Aus-Grad" auch oft "fan-in" bzw "fan-out" genannt.

## Theorem 6.4 (Shannon, 1949)

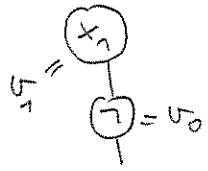
Für jedes $n \geq 1$ und jedes $f: \{0,1\}^n \to \{0,1\}$ gibt es einen SK $C_f$ der Größe $\leq 2 \cdot \frac{2^n}{n}$, der $f$ berechnet.

Beweis:

**Behauptung 1:** Für jedes $\ell \geq 1$ gibt es einen SK $C_\ell$ mit Eingabegatter $x_1, \dots, x_\ell$ und insgesamt $\leq 3 \cdot 2^\ell$ Gattern, in dem es für jedes $a \in \{0,1\}^\ell$ einen Knoten $v_a$ gibt, der genau dann den Wert 1 hat, wenn an den Eingabegatter $x_1 \dots x_\ell$ die Werte $a_1 \dots a_\ell$ eingegeben werden

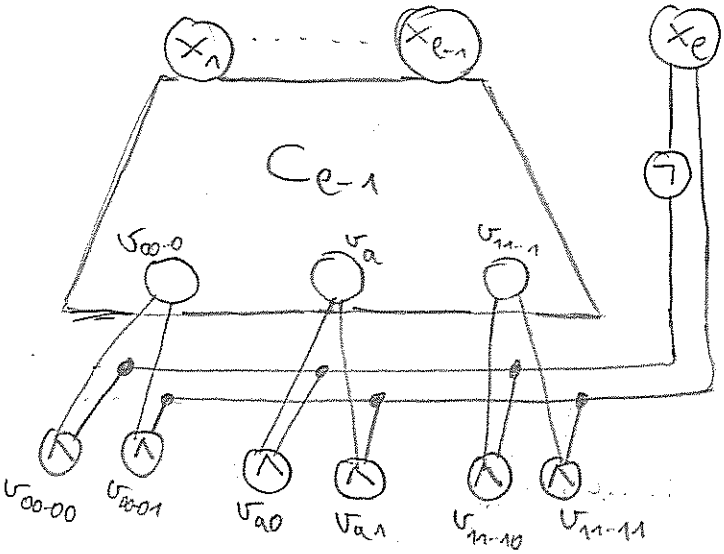
Beweis: Per Induktion nach $\ell$

$\ell = 1$: $C_1$ sieht folgendermaßen aus:




klar: $|C_1| = 2 \leq 3 \cdot 2^1$

$\ell - 1 \to \ell$: $C_\ell$ sieht folgendermaßen aus:




Man sieht leicht, dass für jedes $a \in \{0,1\}^{\ell-1}$ die Knoten $v_{a0}$ und $v_{a1}$ das Gewünschte leisten.

Außerdem gilt:

$$|C_\ell| = |C_{\ell-1}| + 2^\ell + 2$$

Auflösen der Rekursionsgleichung liefert:

$$|C_\ell| = 2 + 2^\ell + |C_{\ell-1}| = 2 \cdot 2 + (2^\ell + 2^{\ell-1}) + |C_{\ell-2}| = 3 \cdot 2 + (2^\ell + 2^{\ell-1} + 2^{\ell-2}) + |C_{\ell-3}|$$
$$= \dots = i \cdot 2 + (2^\ell + 2^{\ell-1} + 2^{\ell-2} + \dots + 2^{\ell-(i-1)}) + |C_{\ell-i}| = \dots = (\ell-1) \cdot 2 + (2^\ell + 2^{\ell-1} + \dots + 2^2) + |C_1|$$
$$= 2\ell - 2 + 2^{\ell+1} - 1 - 2^0 - 2^1 + |C_1| = 2\ell + 2^{\ell+1} - 6 + 2 = 2 \cdot 2^\ell + \underbrace{2\ell - 4}_{> 2^\ell} < 3 \cdot 2^\ell$$

□ Beh. 1

**Behauptung 2:** Für jedes $k \geq 1$ gibt es einen SK $D_k$ mit Eingabegattern $x_1, \dots, x_k$ und insgesamt $\leq 2 \cdot 2^k \cdot 2^{2^k}$ Gattern, in dem es für jede Funktion $g: \{0,1\}^k \to \{0,1\}$ einen Knoten $v_g$ gibt, der die Funktion $g$ berechnet (d.h. $v_g$ hat genau dann den Wert 1, wenn an den Eingabegattern $x_1, \dots, x_k$ Werte $a_1 \dots a_k \in \{0,1\}^k$ eingegeben werden, für die gilt: $g(a_1 \dots a_k) = 1$)

**Beweis:**

Für jedes $g: \{0,1\}^k \to \{0,1\}$ gilt:

$$g(x_1, \dots, x_k) = \bigvee_{\substack{a \in \{0,1\}^k: \\ g(a)=1}} \text{"} x_1 \dots x_k = a \text{"}$$

Zum Bau des SKs $D_k$ erweitern wir den SK $C_k$ aus Behauptung 1 wie folgt:

Für jedes $g: \{0,1\}^k \to \{0,1\}$ fügen wir eine Kette von (höchstens $2^k - 1$) $\vee$-Gattern ein, durch die die Ver-ODER-ung sämtlicher Knoten $v_a$ für $a \in \{0,1\}^k$ mit $g(a) = 1$ berechnet wird. Das letzte dieser $\vee$-Gatter ist der Knoten $v_g$.

Man sieht leicht, dass der so erhaltene SK $D_k$ das Gewünschte leistet. Außerdem gilt:

$$|D_k| \leq |C_k| + \underbrace{2^{2^k}}_{\substack{\text{Anzahl der Fkten} \\ g: \{0,1\}^k \to \{0,1\}}} \cdot 2^k \underset{\text{Beh 1}}{\leq} 3 \cdot 2^k + 2^{2^k} \cdot 2^k \leq \underbrace{2 \cdot 2^k \cdot 2^{2^k}}_{\substack{3 \cdot 2^k < 2^{2^k} \cdot 2^k \\ \text{f.a. } k \geq 1}}$$

□ Beh 2

## Abschluss des Beweises:

Sei $n \geq 1$ und sei $f: \{0,1\}^n \to \{0,1\}$.

Ziel: Konstruktion eines SK $C_f$ der Größe $\leq 22 \cdot \frac{2^n}{n}$, der $f$ berechnet.

Wir wählen $k := \log n - 2$ und $\ell := n - k$.

Sei $D_k$ der SK aus Beh 2, und

sei $C'_\ell$ der SK aus Beh 1, der an Stelle von $x_1, \dots, x_\ell$ die Gatter $x_{k+1}, \dots, \underbrace{x_{k+\ell}}_{=n}$ als Eingabegatter hat.

Für jedes $a \in \{0,1\}^\ell$ sei die Funktion

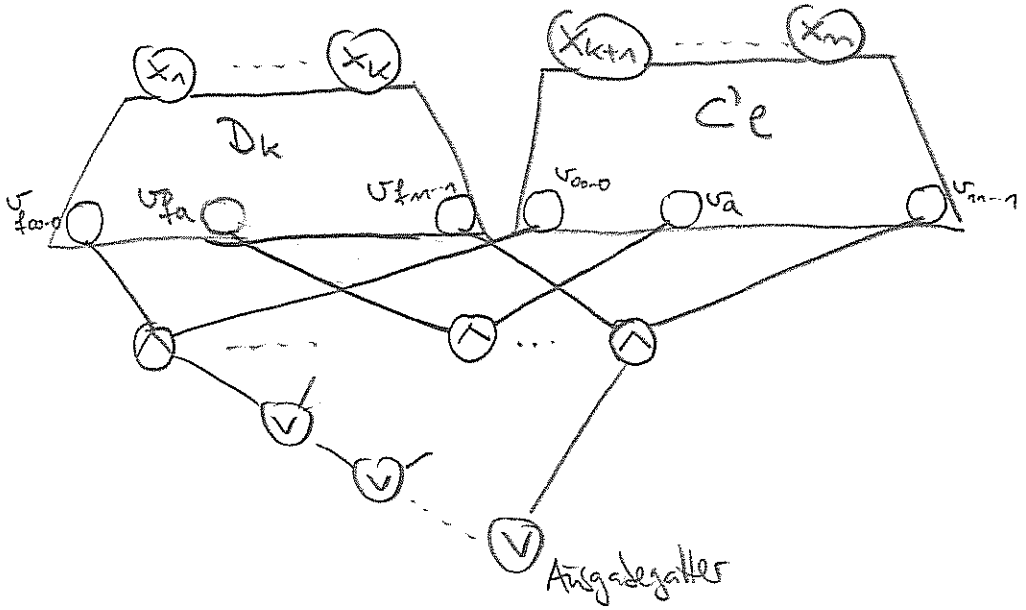
$f_a: \{0,1\}^k \to \{0,1\}$ definiert durch

$$f_a(x_1, \dots, x_k) := f(x_1, \dots, x_k, a_1, \dots, a_\ell)$$

Man sieht leicht, dass folgendes gilt:

$$\circledast \quad f(x_1, \dots, x_n) = \bigvee_{a \in \{0,1\}^\ell} \left( f_a(x_1, \dots, x_k) \wedge "x_{k+1} \dots x_n = a" \right)$$

Sei $C_f$ der folgende SK, der $\circledast$ realisiert und daher $f$ berechnet:




$\} 2^\ell$ $\wedge$-Gatter

$\} 2^\ell - 1$ $\vee$-Gatter

Es gilt:

$$|C_f| = |D_k| + |C'_\ell| + 2^\ell + 2^\ell - 1 \underset{\text{Beh 1+2}}{\leq} 2 \cdot 2^k \cdot 2^{2^k} + 3 \cdot 2^\ell + 2^\ell + 2^\ell - 1$$

Also: $|C_\ell| \leq 2 \cdot 2^k \cdot 2^{2^k} + 5 \cdot 2^\ell$ (*)

Wegen $k = \log n - 2$ und $\ell = n - k = n - \log n + 2$ gilt:

$$5 \cdot 2^\ell = 5 \cdot 2^{n - \log n + 2} = 5 \cdot 2^2 \cdot 2^{n - \log n} = 5 \cdot 4 \cdot \frac{2^n}{n} = 20 \cdot \frac{2^n}{n} \quad (**)$$

und

$$2 \cdot 2^k \cdot 2^{2^k} = 2 \cdot 2^{\log n - 2} \cdot 2^{2^{\log n - 2}} = 2^{\log n - 1} \cdot 2^{\frac{n}{4}} = 2^{\frac{n}{4} + \log n - 1} \quad (***)$$

[Behauptung 3: F.a. $n \geq 1$ gilt: $\frac{n}{4} + \log n - 1 \leq n - \log n + 1$

Beweis:

$$\frac{n}{4} + \log n - 1 \leq n - \log n + 1$$

$$\Leftrightarrow \quad 2 \log n \leq \frac{3}{4} n + 2$$

$$\Leftrightarrow \quad \frac{8}{3} \log n \leq n + \frac{8}{3}$$

$$\Leftrightarrow \quad 2^{\frac{8}{3} \log n} \leq 2^{n + \frac{8}{3}}$$

$$\Leftrightarrow \quad n^{8/3} \leq 2^{8/3} \cdot 2^n$$

$$\Leftarrow \quad n^3 \leq 2^2 \cdot 2^n = 4 \cdot 2^n$$

Somit sind wir fertig, sobald bewiesen ist, dass f.a. $n \geq 1$ gilt: $n^3 \leq 4 \cdot 2^n$. Rest: Übung!

□ Beh 3

Wegen Beh 3 und (***) gilt:

$$2 \cdot 2^k \cdot 2^{2^k} \leq 2^{n - \log n + 1} = 2 \cdot 2^{n - \log n} = 2 \cdot \frac{2^n}{n}.$$

Mit (*) und (**) folgt: $|C_\ell| \leq 2 \cdot \frac{2^n}{n} + 20 \frac{2^n}{n} = 22 \cdot \frac{2^n}{n}$. □

Das nächste Theorem zeigt, dass die Konstruktion des Schaltkreises $C_\ell$ im obigen Beweis fast optimal ist.

**Theorem 6.5** (Shannon, 1949)

Für jedes $n \geq 1$ gibt es eine Funktion $f: \{0,1\}^n \to \{0,1\}$, die durch keinen SK der Größe $\leq \frac{1}{10} \cdot \frac{2^n}{n}$ berechnet wird.

**Beweis:** Wir nützen ein einfaches Zählargument.

Sei $F_n$ die Anzahl der Funktionen $f: \{0,1\}^n \to \{0,1\}$, d.h. $F_n = 2^{2^n}$.

Für $s := \frac{1}{10} \cdot \frac{2^n}{n}$ sei $K_s$ die Anzahl der SKe der Größe $\leq s$.

Wir zeigen im Folgenden, dass $K_s < 2^{2^n} = F_n$ ist.

Um $K_s$ abzuschätzen, repräsentieren wir jeden SK $C$ der Größe $\leq s$ wie folgt durch einen Bitstring $b(C)$:

- Nummeriere die Knoten von $C$ mit den Zahlen $0, 1, \ldots, |C|-1 \leq s$. Der Ausgabeknoten bekommt die Nummer 0.
- Für jeden Knoten $i \in \{0, \ldots, |C|-1\}$ sei $b(i)$ der wie folgt definierte Bitstring:

  (1) Die ersten $\log(n+5)$ Bits geben an, von welcher Sorte aus $\{x_1, \ldots, x_n, \wedge, \vee, \neg, 0, 1\}$ der Knoten $i$ ist

  (2) Die nächsten 0, $\log s$ oder $2\log s$ Bits geben die 0, 1 oder 2 Knoten an, von denen aus Kanten zum Knoten $i$ hinführen

  Beachte: Für (1) und (2) werden $\leq \log(n+5) + 2\log s$ Bits benötigt. Das ist $\leq 9 \cdot \log s$, denn:

  $\log(n+5) \leq 7 \log s \iff n+5 \leq 2^7 \cdot s \iff n+5 \leq 128 \cdot s \iff$

  $n+5 \leq \frac{128}{10} \cdot \frac{2^n}{n} \impliedby n^2+5n \leq 10 \cdot 2^n$ — und dies gilt f.a. $n \geq 0$.

  (3) Wir füllen $b(i)$ mit Oen auf zu einem Bitstring der Länge $9 \log s$

- Wir konkatenieren die Bitstrings $b(i)$ für $i = 0, 1, \dots, |C|-1$ und füllen das Ganze mit Oen auf einem Bitstring $b(C)$ der Länge $s \cdot g\log s$

  Beachte: Aus $b(C)$ lässt sich eindeutig der SK $C$ rekonstruieren.

Insbesondere gilt:

$K_s \overset{\text{Def}}{=}$ Anzahl der SKe der Größe $\leq s$ $\leq$ Anzahl der Bitstrings der Länge $g s \log s$ $\leq 2^{g s \log s}$.

Somit gilt:

$$K_s \leq 2^{g s \log s} \overset{s = \frac{1}{10}\frac{2^n}{n}}{=} 2^{\frac{g}{10}\cdot\frac{2^n}{n}\log\left(\frac{2^n}{10n}\right)} = 2^{2^n\left(\frac{g}{10n}\log\left(\frac{2^n}{10n}\right)\right)}$$

$$\leq 2^{2^n\left(\frac{g}{10n}\cdot\log(2^n)\right)}$$

$$= 2^{2^n\left(\frac{g}{10n}\cdot n\right)} = 2^{2^n\cdot\frac{g}{10}} < 2^{2^n} = \#_n$$

□

# 6.3 Die Klasse SIZE(S(n))

## Definition 6.6 (Die Klasse SIZE(S(n)))

(a) Eine Schaltkreisfamilie (kurz: SK-Familie) ist eine Folge $(C_n)_{n\in\mathbb{N}}$, wobei für jedes $n\in\mathbb{N}$ $C_n$ ein SK mit $n$ Eingabegattern ist

(b) Die von einer SK-Familie $(C_n)_{n\in\mathbb{N}}$ berechnete (oder definierte) Sprache ist

$$L := \{ x\in\{0,1\}^* : n\in\mathbb{N},\ C_n(x) = 1 \}$$

(c) Sei $S: \mathbb{N}\to\mathbb{N}$.
Eine SK-Familie $(C_n)_{n\in\mathbb{N}}$ hat Größe $S(n)$, falls $\forall\, n\in\mathbb{N}$ gilt $|C_n| \le S(n)$.

(d) Sei $S:\mathbb{N}\to\mathbb{N}$.

$$\mathrm{SIZE}(S(n)) := \{ L\subseteq\{0,1\}^* : \text{Es gibt eine SK-Familie der Größe } S(n), \text{ die } L \text{ berechnet} \}$$

## Bemerkung 6.7

Aus Theorem 6.4 folgt, dass für jede Funktion $S:\mathbb{N}\to\mathbb{N}$ mit $S(n) \ge 22\cdot\frac{2^n}{n}$ (für $n\ge 1$) gilt:

$$\mathrm{SIZE}(S(n)) = \{ L : L\subseteq\{0,1\}^* \}$$

(d.h. alle Sprachen liegen in $\mathrm{SIZE}(S(n))$).

Aus Shannons Theoremen 6.4 und 6.5 erhalten wir folgenden Hierarchiesatz:

**Theorem 6.8** (Nichtuniformer Hierarchiesatz)

Seien $s: \mathbb{N} \to \mathbb{N}$ und $S: \mathbb{N} \to \mathbb{N}$ s.d. f.a. hinreichend großen $n \in \mathbb{N}$ gilt:

(1) $n \leq s(n) \leq \frac{1}{10} \cdot \frac{2^n}{n}$ und

(2) $440 \cdot s(n) \leq S(n)$.

Dann gilt:

$$SIZE(s(n)) \subsetneq SIZE(S(n))$$

Beweis: Sei $n$ hinreichend groß, s.d. (1) und (2) gilt.

Fall 1: $S(n) \geq 22 \cdot \frac{2^n}{n}$

Wegen Theorem 6.4 kann jedes $f: \{0,1\}^n \to \{0,1\}$ durch einen SK der Größe $S(n)$ berechnet werden.
Wegen Theorem 6.5 gibt es ein $f: \{0,1\}^n \to \{0,1\}$, das durch keinen SK der Größe $s(n) \leq \frac{1}{10} \cdot \frac{2^n}{n}$ berechnet wird
Somit ist $SIZE(s(n)) \subsetneq SIZE(S(n))$.

Fall 2: $S(n) < 22 \cdot \frac{2^n}{n}$

Wähle $\ell \in \{1, \dots, n\}$ maximal, s.d. gilt: ⊛

$$S(n) \geq 22 \cdot \frac{2^\ell}{\ell} - (n-\ell)$$

Beachte: Es gilt $\ell \in \{1, \dots, n-1\}$, denn:

$22 \cdot \frac{2^1}{1} - (n-1) = 44 - n + 1 \leq 45 < 440 \cdot 1 \leq 440 \cdot s(n) \leq S(n)$ und

$22 \cdot \frac{2^n}{n} - (n-n) = 22 \cdot \frac{2^n}{n} > S(n)$, da wir in Fall 2 sind

Gemäß Theorem 6.5 und 6.4 gibt es ein $f: \{0,1\}^\ell \to \{0,1\}$, das durch keinen SK der Größe $\frac{1}{10}\frac{2^\ell}{\ell}$, aber durch einen SK der Größe $22 \cdot \frac{2^\ell}{\ell}$ berechnet werden kann.

Sei $g: \{0,1\}^n \to \{0,1\}$ mit $g(x_1 \dots x_n) := f(x_1, \dots, x_\ell)$.

Klar: $g$ wird durch einen SK der Größe $\leq$

$$\underbrace{22 \cdot \frac{2^\ell}{\ell}}_{\text{SK für } f} + \underbrace{(n-\ell)}_{\substack{\text{zusätzliche} \\ \text{Eingabegatter} \\ \text{für } x_{\ell+1}, \dots, x_n}} \overset{\circledast}{\leq} S(n)$$

Behauptung: $g$ wird durch keinen SK der Größe $s(n)$ berechnet.

Beweis: Angenommen doch, dann wird auch $f$ durch diesen SK berechnet (indem die Eingabegatter $x_{\ell+1}, \dots, x_n$ durch ein mit 0 markiertes Gatter ersetzt werden).

Wir zeigen, dass $s(n) \leq \frac{1}{10} \cdot \frac{2^\ell}{\ell}$ ist — ein Widerspruch zur Wahl von $f$.

Wegen $\circledast$ (Maximalität von $\ell$) gilt:

$$S'(n) < 22 \cdot \frac{2^{\ell+1}}{\ell+1} - \underbrace{(n-(\ell+1))}_{\geq 0, \text{ da } \ell+1 \leq n} \leq 22 \cdot \frac{2^{\ell+1}}{\ell} = 44 \cdot \frac{2^\ell}{\ell}$$

V1 (2)

$$440 \cdot s(n)$$

Somit ist $s(n) \leq \frac{44}{440} \cdot \frac{2^\ell}{\ell} = \frac{1}{10} \cdot \frac{2^\ell}{\ell}$. □ Behauptung.

Wir haben gezeigt, dass $g$ durch einen SK der Größe $S(n)$, nicht aber durch einen SK der Größe $s(n)$ berechnet wird.

Somit ist $\text{SIZE}(s(n)) \subsetneq \text{SIZE}(S(n))$.

□

## Folgerung 6.9

- $SIZE(n) \subsetneq SIZE(500n)$ und
- $SIZE(n) \subsetneq SIZE(n^2) \subsetneq SIZE(n^3) \subsetneq \dots$
- $\bigcup_{c \geq 1} SIZE(n^c) \subsetneq SIZE\left(\frac{2^n}{n}\right)$

## 6.4 Die Klasse P/poly

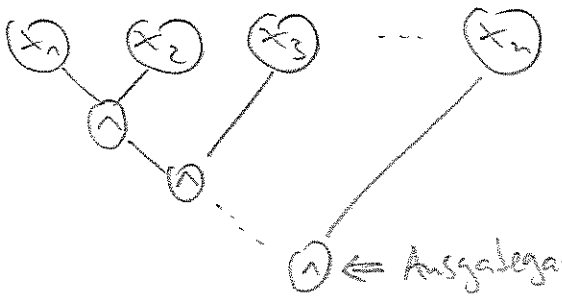
### Definition 6.10 (P/poly)

$$P/poly := \bigcup_{c \geq 1} SIZE(n^c)$$

ist die Klasse aller Sprachen, die durch SK-Familien polynomieller Größe berechnet werden.

### Beispiel 6.11

Die Sprache $\{1^n : n \in \mathbb{N}\}$ wird durch eine SK-Familie der Größe $2n-1$ berechnet und gehört daher zu P/poly.
Der SK $C_n$ dieser SK-Familie ist einfach die Konjunktion aller $n$ Eingabegatter:



## Bemerkung 6.12

(a) Jede unäre Sprache $L$ (d.h. $L \subseteq \{1\}^*$) gehört zu P/poly (und wird durch eine Sk-Familie der Größe $O(n)$ berechnet):
Die Sk-Familie $(C_n)_{n\in\mathbb{N}}$, die $L$ berechnet, kann wie folgt gewählt werden:
F.a. $n\in\mathbb{N}$ gilt:

- Falls $1^n \in L$, so ist $C_n$ der Sk aus Beispiel 6.11
- Falls $1^n \notin L$, so ist $C_n$ ein Sk, der immer 0 ausgibt — etwa:


(b) P/poly enthält einige unentscheidbare Sprachen — z.B. die unäre Sprache

$$\text{UHALT} := \{1^n : \text{die Binärdarstellung von } n \text{ repräsentiert eine TM } M, \text{ die bei leerer Eingabe anhält}\}.$$

(c) Wegen Theorem 6.5 bzw. dem Hierarchiesatz Theorem 6.8 gibt es Sprachen, die nicht in P/poly liegen.