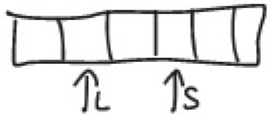
Aufgabe 5: Turing-Vollständigkeit (12 Punkte)

Eine 2-*vorlaufende Turingmaschine* ist eine rechnende Turingmaschine mit folgenden Modifikationen:

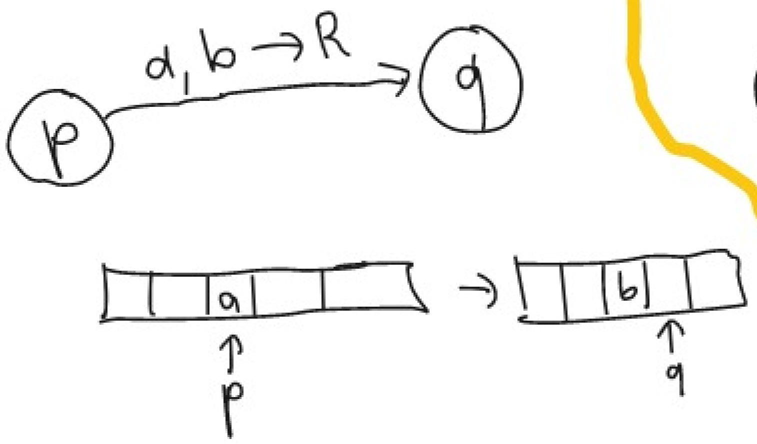
- Anstelle des Schreib-Lese-Kopfes gibt es einen Schreibkopf und einen Lesekopf.
- Der Schreibkopf steht 2 Zellen weiter rechts als der Lesekopf.
- Die Angabe der Bewegungsrichtung bei einem Übergang gilt für beide Köpfe gleichzeitig.
- Der Schreibkopf muss nicht bei jedem Übergang etwas schreiben; in diesem Fall wird anstelle des zu schreibenden Symbols am Übergang ε notiert.
- Zu Beginn bzw. Ende der Ausführung steht der Lesekopf auf dem ersten Symbol der Eingabe bzw. Ausgabe.



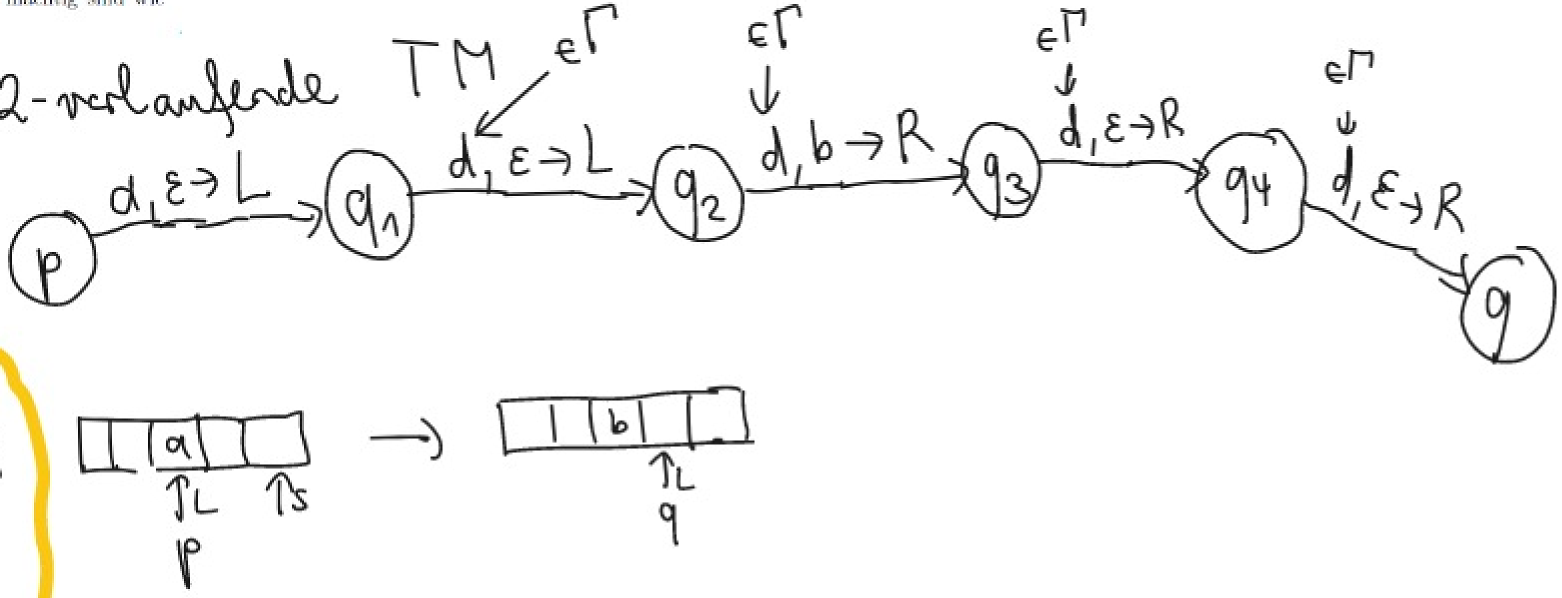
(a) **Beweis** (8 Punkte)

Zeigen Sie, dass 2-vorlaufende Turingmaschinen mindestens genauso mächtig sind wie rechnende Turingmaschinen.

rechnende TM



2-vorlaufende TM



miro

(b) **Halteproblem** (4 Punkte)

Ist das folgende Problem entscheidbar? Begründen Sie.

> 2-VorHalt
> **Gegeben:** Eine 2-vorlaufende TM $\mathcal{M}$ mit Bandalphabet $\Gamma := \{\square, 0, 1, 2\}$.
> **Gefragt:** Hält $\mathcal{M}$ bei Eingabe 2?

$H_2 = \{ w \in \{0,1\}^* \mid$ w Code eine 2-vorlaufende TM M und $M(2)\downarrow \}$

L_1, L_2 Sprachen und $L_1 \leq L_2$ (reduzierbar) und L_2 ist nicht entscheidbar

$\Rightarrow L_1$ nicht entscheidbar

$H_0 = \{ w \in \{0,1\}^* \mid$ w Code eine TM M und $M(\varepsilon)\downarrow \}$ (hält)

H_0 nicht entscheidbar nach VL

gesucht: Reduktion, d.h. totale und berechenbare Fkt $f: \Sigma^* \to \Sigma^*$ sodass

$\forall w \in \Sigma^*: w \in H_2 \Leftrightarrow f(w) \in H_0$

$f: \Sigma^* \to \Sigma^*$

$w \mapsto$ Code eine TM N, die folgendes tut: Bei Eingabe von ε simuliert M die TM M_w, die durch w codiert wird auf Eingabe 2

Sieb des Erasthotenes

false

~~1~~ 2 3 ~~4~~ 5 ~~6~~ 7 ~~8~~ ~~9~~ ~~10~~
11 ~~12~~ 13 ~~14~~ ~~15~~ ~~16~~ 17 ~~18~~ 19 ~~20~~
~~21~~ ~~22~~ 23 ~~24~~ ~~25~~ ~~26~~ ~~27~~ ~~28~~ 29 ~~30~~
31 ~~32~~ ~~33~~ ~~34~~ ~~35~~ ~~36~~ 37 ~~38~~ ~~39~~ ~~40~~

Aufgabe 7: Komplexität (14 Punkte)

(a) **Pseudopolynomieller Algorithmus** (8 Punkte)

> Primzahlmenge
> **Gegeben:** Endliche Menge natürlicher Zahlen $M \subseteq \mathbb{N}$.
> **Gefragt:** Sind alle Zahlen in M prim?

Geben Sie einen deterministischen Algorithmus an, der Primzahlmenge in **pseudopolynomieller Zeit** entscheidet, ohne einen Primzahltest als Blackbox zu verwenden. Begründen Sie.

Hinweis: Nehmen Sie an, dass Sie für zwei natürliche Zahlen $\alpha < \beta$ in Zeit $\mathcal{O}(\log \beta)$ entscheiden können, ob β von α geteilt wird.

Definition.
Sei a der größte in der Eingabe vorkommende Zahlenwert.
Ein **pseudopolynomieller Algorithmus** hat eine Laufzeit polynomiell in n und a, d.h. $\mathcal{O}(n^c \cdot a^d)$ für Konstanten $c, d \geq 0$.

```
public static boolean allPrim(int[] M) {
    int a = M[0];
    for (int i = 0; i < M.length; ++i) {
        if (M[i] > a) {
            a = M[i];
        }
    }
    boolean[] primes = new boolean[a+1];
    for (int i = 0; i < primes.length; ++i) {
        primes[i] = true;
    }
    primes[0] = false;
    primes[1] = false;
    for (int i = 2; i < primes.length; ++i) {
        if (!primes[i]) { continue; }
        for (k = i+i; k < primes.length; k = k+i) {
            primes[k] = false;
        }
    }
    for (int i = 0; i < M.length; ++i) {
        if (!primes[M[i]]) { return false; }
    }
    return true;
}
```

M.length

$O(n)$

$O(a)$

$O(a^2)$

$O(n)$

primes[4] = false $\Leftrightarrow$ 4 keine Primzahl

primes[5] = true $\Leftrightarrow$ 5 Primzahl

$V(2) = \{ k \cdot 2 \mid k \in \mathbb{N} \}$

primes = {false, false, true, true, false, true, false, true, false, false, false, true}
(indices 0 1 2 3 4 5 6 7 8 9 10 11)

M = { 3, 7, 10, 11 }

$O(n \cdot a^2)$

$c = 1, \; d = 2$

Aufgabe 8: *NP*-Vollständigkeit **(16 Punkte)**

Ein Rechteck $R = (x, y)$ ist gegeben durch seine beiden Kantenlängen $x, y \in \mathbb{N}$. Ein Rechteck heißt *nicht-trivial*, wenn beide Kanten mindestens Länge 2 haben. Der Flächeninhalt eines Rechtecks $R = (x, y)$ berechnet sich als $x \cdot y$.

SIEBTELFLÄCHE

Gegeben: Ein nicht-triviales Rechteck $Z = (x, y)$; Menge $\mathcal{R} = \{R_1, \ldots, R_n\}$ von nicht-trivialen Rechtecken mit $R_i = (x_i, y_i)$.

Gefragt: Gibt es eine Auswahl $A \subseteq \mathcal{R}$ von nicht-trivialen Rechtecken, sodass diese überlappungsfrei zu einem Rechteck angeordnet werden können, dessen Flächeninhalt genau ein Siebtel des Flächeninhalts von Z ist?

(a) Reduktion (12 Punkte)

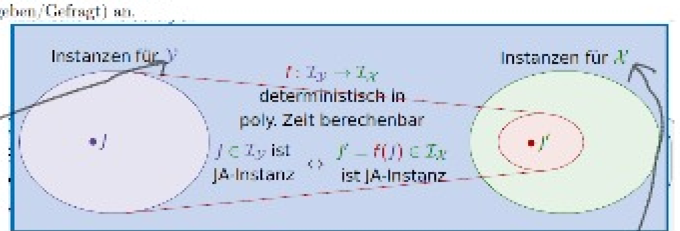
Zeigen Sie, dass SIEBTELFLÄCHE *NP*-schwer ist.

Das in Ihrer Reduktion verwendete Ausgangsproblem $\mathcal{X}$ muss aus der Vorlesung stammen. Geben Sie die Definition von $\mathcal{X}$ nach dem bekannten Schema (Gegeben/Gefragt) an.

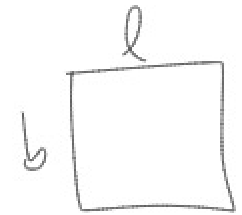
Pseudopolynomieller Algorithmus für SUBSETSUM. = Y

Gegeben. $A = \{a_1, \ldots, a_m\}$, $b \in \mathbb{N}$.

Gefragt. Existiert $S \subseteq A$ mit $[S] = b$?



Siebtelfläche = X



$F = l \cdot b$

$f : J_Y \to J_X$

$A = \{a_1, \ldots, a_m\}$, $b \in \mathbb{N}$ $\longmapsto$ $R_1 = (10, 10 \cdot a_1)$, $R_2 = (10, 10 \cdot a_2)$, $\ldots$, $R_m = (10, 10 \cdot a_m)$

$R = (7, b \cdot 100)$

Beispiel: $m = 7$

$a_1 + a_3 + a_5 = b$

$F_{R_1} + F_{R_3} + F_{R_5} =$
$= 100 \cdot a_1 + 100 \cdot a_3 + 100 \cdot a_5 = 100 \cdot (a_1 + a_3 + a_5) = 100 \cdot b \stackrel{!}{=} \frac{1}{7} \cdot F_R$

$F_R = 700 b$

$A = \{a_1, \ldots, a_m\}$, $b \in \mathbb{N}$ $= J$ Ja-Instanz in Y.

d.h. wir finden $S \subseteq A$ mit $[S] = b$

$S = \{a_{s_1}, \ldots, a_{s_n}\}$

$\Longrightarrow$ $f(J)$ ist Ja-Instanz in X, da

$F_{R_{s_1}} + F_{R_{s_2}} + \ldots + F_{R_{s_n}} = \frac{1}{7} \cdot F_R$

miro