Pokroky matematiky, fyziky a astronomie

B. A. Trachtěnbrot

Algoritmy a počítací stroje [Dokončení]

*Pokroky matematiky, fyziky a astronomie*, Vol. 2 (1957), No. 5, 544--552

Persistent URL: `http://dml.cz/dmlcz/137195`

*Tabulka 2.*

Koeficienty $D_6$

| Rozsah výběru $n$ | Přípustné průměrné procento zmetků za mezními rozměry | | | |
|---|---|---|---|---|
| | 2 % | 1 % | 0,5 % | 0,27 % |
| 2 | 0,48 | 0,43 | 0,40 | 0,37 |
| 3 | 51 | 46 | 43 | 40 |
| 4 | 54 | 48 | 44 | 42 |
| 5 | 55 | 50 | 46 | 43 |
| 6 | 57 | 51 | 47 | 44 |
| 7 | 58 | 52 | 48 | 45 |
| 8 | 59 | 53 | 49 | 45 |
| 9 | 59 | 54 | 49 | 46 |
| 10 | 60 | 54 | 50 | 47 |

*(Dokončení)*

# ALGORITMY A POČÍTACÍ STROJE*)

*(Dokončení)*

## § 5.

Turingův stroj má ve srovnání s elektronickými počítacími stroji jisté odchylné vlastnosti, protože byl navržen ještě před vznikem prvních elektronických počítacích strojů. Jsou to tyto odchylky:

I. V Turingově stroji je provedeno rozčlenění celého procesu na nejjednodušší elementární operace, namnoze ještě jednodušší než v elektronických samočinných počítačích.

II. Paměť Turingova stroje má tvar nekonečné pásky rozdělené na buňky (žádný skutečný stroj ovšem nemůže mít nekonečnou paměť, takže zde jde o idealisaci).

Uveďme si nyní podrobný popis Turingova stroje:

1. Turingův stroj disponuje konečným počtem symbolů $S_1, S_2, \ldots, S_k$, které představují tak zvanou z á k l a d n í a b e c e d u s t r o j e. V této abecedě se kodují jak údaje do stroje vkládané, tak dílčí údaje, které se v něm postupně vytvářejí. Dohodneme se na tom, že mezi symboly základní abecedy budeme počítat i „prázdný“ znak (třeba $S_1$), který má ten význam, že po vyslání do některé buňky paměti vymazává její obsah, takže pak buňka je prázdná. Říkáme o ní též, že uchovává prázdný znak. V libovolné fázi práce stroje může být v každé buňce paměti nejvýš jeden znak. Na začátku práce stroje se do pásky paměti uloží počáteční údaje (počáteční informace) zakodované určitým systémem znaků základní abecedy. Stroj pracuje v taktech jdoucích po sobě, během nichž se počáteční informace postupně přetváří v dílčí výsledné informace. Podle druhu počáteční informace $A$ mohou nastat tyto dva případy:

a) Po proběhnutí konečného počtu taktů se stroj zastaví a vyšle signál o svém zastavení. Na pásce se při tom objeví obraz určité informace $L$. V tom případě říkáme,

*) Referát z článku B. A. T r a c h t ě n b r o t, *Algoritmy i mašinnoje rešenije zadač*, Mat. v škole, 1956, č. 4 a 5.

že stroje lze užít ke zpracování dané informace $A$, kterou přetváří ve výslednou informaci $L$.

b) Ve stroji nastává neukončený proces, stroj se nikdy nezastaví ani sám nikdy nevydá signál k zastavení. V tom případě říkáme, že stroje nelze užít k zpracování dané počáteční informace $A$.

Říkáme, že stroj řeší určitou třídu úloh, jestliže jej můžeme vyždycky užít k zpracování informací, zobrazujících v určitém kodu podmínky každé jednotlivé úlohy dané třídy, a přetváří-li je v informace, které zobrazují v témže kodu řešení této úlohy.

2. Systém tříadresových příkazů, kterého se užívá v existujících elektronických strojích, je podmíněn možností provádět elementární operace za účasti obsahů tří buněk paměti. V některých elektronických počitačích se však užívá systému jednoadresových příkazů, který záleží v tom, že v každém taktu práce stroje se účastní jen jedna jediná buňka paměti. Budeme ji nazývat „prověřovanou“ buňkou v daném taktu.

Na příklad tříadresový příkaz k sečtení čísel z buněk $\beta$ a $\gamma$, a k dopravení výsledku do buňky $\delta$, můžeme za jistých podmínek nahradit třemi po sobě jdoucími jednoadresovými příkazy:

a) Přenést (do sčítače) číslo z buňky $\beta$;

b) Přenést (do sčítače) číslo z buňky $\gamma$;

c) Výsledek dopravit do buňky $\delta$.

V Turingově stroji je jak systém operací, tak systém jednoadresových příkazů velmi zjednodušen, a to v tomto smyslu: V každém jednotlivém taktu předpisuje příkaz záměnu jen jednoho znaku $S_i$ uloženého v prověřované buňce, libovolným jiným znakem $S_j$. Je-li $i = j$, nemění se obsah prověřované buňky. Případ $j = 1$ znamená, že obsah prověřované buňky se vymazává. Další zjednodušení záleží v tom, že při přechodu stroje z jednoho taktu do druhého nemůže se změnit adresa prověřované buňky více než o jedničku, což znamená, že se v druhém taktu prověřuje právě pravá nebo levá sousední buňka k buňce prověřované v předchozím taktu.

Zjednodušení tedy záleží na příklad v tom, že má-li stroj nalézti údaj uložený v buňce jakkoli vzdálené, provede to tím způsobem, že po řadě prověří všechny buňky až do té, jejíž obsah byl hledán. Tento postup je sice zdlouhavý, umožňuje však velké zjednodušení samé instrukce v tomto smyslu: V instrukčních příkazech lze místo adres prověřovaných buněk užít jen těchto tří posunových kodů:

$P_1$ .... prověřit pravou sousední buňku,

$P_{-1}$ ... prověřit levou sousední buňku,

$P_0$ .... pokračovat v prověřování téže buňky.

3. Elektronický samočinný počitač zpracovává číselné, v paměti uložené informace v aritmetickém bloku, který může být vždy jen v jednom z konečného počtu stavů: stav „sčítací“, stav „odčítací“, atd. Má-li být provedena určitá operace, zavedou se do aritmetického bloku z paměti nejen čísla, nad nimiž se má operace provést, nýbrž i signál, který zapne blok na žádaný druh operace [(na př. sčítání, viz *obr. 2b*)].

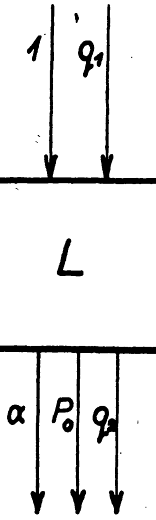
V Turingově stroji se na rozdíl od elektronických strojů zpracovávají informace nikoli v aritmetickém bloku, nýbrž v tak zvaném logickém bloku, který může být rovněž jen v jednom z konečného počtu stavů. Nechť symboly

$$q_1, q_2, \ldots, q_n \qquad (Q)$$

značí kody těchto stavů. Logický blok má dva vstupní kanály. Jedním z nich do něj vstupuje v každém taktu znak $S_i$ vycházející z prověřované buňky. Druhým kanálem vstupuje do logického bloku kod $q_l$ toho stavu, ve kterém musí blok v daném taktu být. Výstupním kanálem posílá logický blok do prověřované buňky příslušný přepracovaný znak $S_j$, který je zřejmě jednoznačnou funkcí vstupních signálů $S_i q_l$. Příkazy, které řídí práci stroje v každém taktu, mají tento tvar:

$$P_n q_l \, (n = -1, 0, 1; \quad l = 1, 2, \ldots, m),$$

kde první znak zaměňuje adresu prověřované buňky a druhý předpisuje logickému bloku určitý stav. Znaky $P_{-1}, P_0, P_1, q_1, q_2, \ldots, q_m$ představují tak zvanou pomocnou abecedu stroje. Zvláštností Turingova stroje je ta okolnost, že logický blok má též za úkol vytvářet v každém taktu příkaz, který se dopravuje do řídícího bloku k začátku




Obr. 4

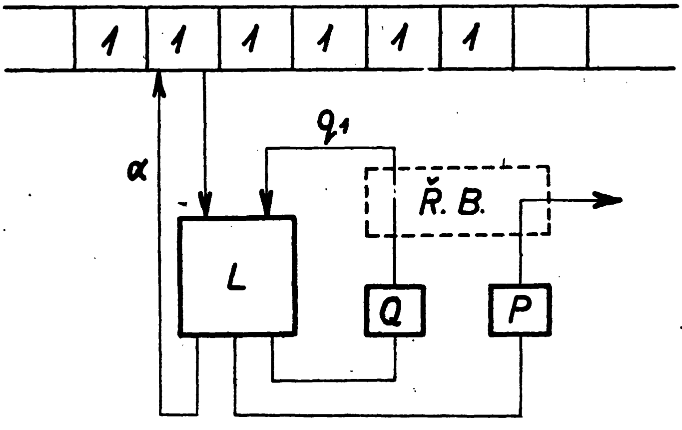
| | 1 | α | β | 0 |
|---|---|---|---|---|
| $q_1$ | $\alpha P_0 q_2$ | $\alpha P_{-1} q_1$ | $\beta P_{-1} q_1$ | $0 P_1 q_4$ |
| $q_2$ | $\beta P_0 q_1$ | $\alpha P_1 q_2$ | $\beta P_1 q_2$ | $0 P_{-1} q_3$ |
| $q_3$ | $1 P_1 q_1$ | $1 P_{-1} q_3$ | $0 P_{-1} q_3$ | $0 P_1 q_1$ |
| $q_4$ | $1 P_{-1} q_1$ | $0 P_1 q_4$ | $1 P_1 q_4$ | $0 P_0 q_5$ |
| $q_5$ | $1 P_0 q_5$ | $\alpha P_0 q_5$ | $\beta P_0 q_5$ | $0 P_0 q_6$ |

Obr. 5

dalšího taktu. Kromě kanálu pro výstup znaku $S_j$ má tedy logický blok ještě dva další kanály pro výstup kodů dvojice řídících příkazů (viz *obr. 4*). Podstatné je přitom to, že výstupní údaje $S_j P_m q_{l'}$ závisí výlučně na tom, jaké znaky $S_i q_l$ byly v témže taktu vloženy do bloku. To znamená, že v logickém bloku se tvoří funkce, která přiřazuje každé dvojici znaků $S_i q_l$ (všech těchto dvojic je $k \cdot m$) trojici znaků $S_j P_m q_{l'}$. Tuto funkci, kterou nazýváme logickou funkcí stroje, můžeme nejlépe zobrazit ve tvaru obdélníkové tabulky, jejíž řádky jsou označeny stavovými kody a sloupce znaky základní abecedy. V každém okénku tabulky je zapsána odpovídající trojice symbolů. Tuto tabulku, zobrazující logickou funkci stroje, budeme nazývat funkčním schematem stroje (viz *obr. 5*).

Z uvedeného výkladu je již patrno, že práce Turingova stroje je úplně určena logickou funkcí, kterou vytváří logický blok. Jinými slovy, dva Turingovy stroje, které mají stejné funkční schema, jsou s hlediska pracovního totožné. Na druhé straně, všechny Turingovy stroje jsou založeny na témže základním principu, a to v tom smyslu, že mají stejné strukturální schema, které popisuje schematicky složení stroje a vzájemné působení jeho orgánů (viz *obr. 6*). V tomto schematu je paměť zobrazena jednak buňkami nekonečné pásky, v nichž se ukládají informace zakodované v symbolice základní abecedy $S$, jednak dalšími dvěma buňkami, v nichž se ukládá řadící příkaz instrukce. Přitom buňka $Q$ uchovává stavový kod a buňka $P$ uchovává posunový kod. V těchto dvou buňkách se přechodně podrží kody $P_n q_{l'}$, vystupující z logického bloku $L$ po proběhnutí určitého taktu práce stroje, a to až do začátku dalšího taktu, kdy vstoupí do řídícího bloku. V této situaci jsou již funkce řídícího bloku velmi zjednodušeny. Jde v podstatě o to, zajistit posun pásky vždy o jednu buňku v souhlase s tím, jak postupuje posunový kod.

Popišme si nyní postup práce Turingova stroje. Před spuštěním stroje se přenese na pásku počáteční informace (na našem obrázku posloupnost jedniček) a zvolí se určitá výchozí buňka paměti (na obrázku buňka obsahující druhou jedničku zleva). Do buněk




*Obr. 6*

$Q$ a $P$ se uloží kody počátečního stavu a počátečního posunu (na příklad $q_1$ a $P_0$). Po spuštění stroje probíhá již celý proces, určený jednoznačně funkčním schematem stroje, úplně automaticky.

Popišme si podrobně, co se děje ve stroji v případě, že je dáno funkční schema podle *obr. 5*.

P r v n í  t a k t. Prověří se znak 1 ve výchozí buňce (neboť je předepsán posun $P_0$) za stavu $q_1$. Výsledkem je, že jednička se nahradí znakem $\alpha$, a do buněk $P$ a $Q$ se vyšle řadící příkaz $P_0 q_2$, který se zde uloží do příštího taktu.

D r u h ý  t a k t. Prověří se znak $\alpha$ v téže výchozí buňce jako dříve (posun $P_0$) za stavu $q_2$. Výsledkem je, že znak $\alpha$ se nezmění, a vyšle se příkaz $P_1 q_2$.

T ř e t í  t a k t. Prověří se znak 1 v pravé sousední buňce (posun $P_1$) za stavu $q_2$. Výsledkem je, že jednička se zamění znakem $\beta$, a vyšle se příkaz $P_0 q_1$.

Tímto způsobem pracuje stroj podle funkčního schematu na *obr. 5* dále, a to až do té doby, kdy se dostane do stavu $q_5$. V tomto okamžiku se zastaví.

## § 6.

Popišme si nyní proces probíhající v Turingově stroji při výpočtu největšího společného dělitele dvou daných přirozených čísel, to jest průběh řešení Euklidova algoritmu.

V § 3 jsme si již uvedli popis provádění Euklidova algoritmu upraveného pro výpočet elektronickým počítačem. V této úpravě se rozčleňuje celý proces na dva typy operací, a to operaci srovnávání dvojic čísel, a za druhé operaci odčítání menšího čísla od většího.

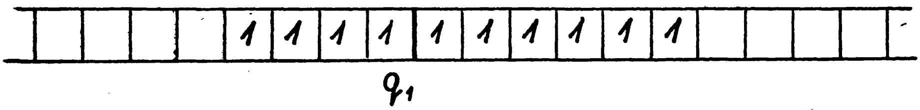
Ve stroji Turingově se však každá z těchto dvou operací rozčleňuje ještě dále vždy na celý cyklus jednodušších operací. Celý proces pak sestává ze dvou navzájem se střídajících cyklů, a to cyklu „srovnávacího" a cyklu „odčítacího". Celý proces se uskutečňuje v Turingově stroji s funkčním schematem právě podle *obr. 5* a se základní abecedou

$$1, \alpha, \beta, 0.$$

Libovolné přirozené číslo $n$ bude v této abecedě zakodováno takto:

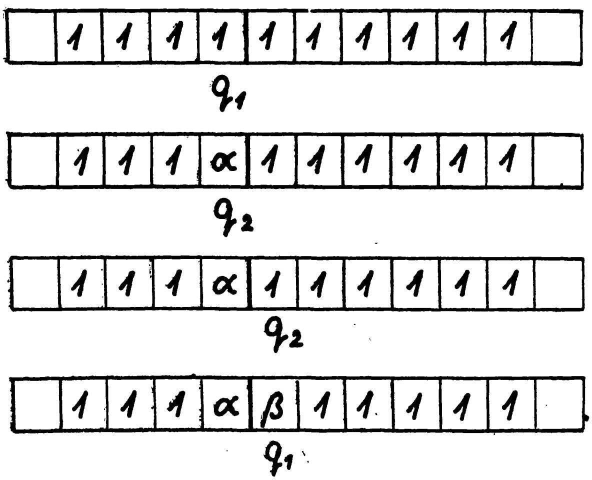
$$\underbrace{1\,1 \ldots 1}_{n\text{-krát}} \quad \text{(t. j. } n \text{ jedniček zapsaných v } n \text{ sousedních buňkách).}$$

Symbol 0 hraje roli prázdného znaku (viz § 5). Symboly $\alpha$ a $\beta$ pak hrají roli těch grafických značek, které si dělá výpočtář (třeba ve tvaru čárek) k zapamatování určitých okolností, které vzniknou během procesu.



*Obr. 7*

Při řešení úlohy postupujeme tak, že nejprve daná přirozená čísla $m$ a $n$, pro něž máme vypočíst největšího společného dělitele, zapíšeme pomocí posloupností jedniček v $m + n$ sousedních buňkách. Za první prověřovanou buňku vezmeme tu buňku, v níž je zapsána poslední jednička prvého čísla $m$. Další obecný popis doprovodíme obrázky, na nichž je ilustrován případ nalezení největšího společného dělitele čísel $m = 4$, $n = 6$. Jednotlivé fáze výpočtu budeme přitom ilustrovat tak zvanými konfiguracemi, při čemž $n$-tou konfigurací nazveme schema zobrazující paměťovou pásku stroje, na níž byla zakodována ta informace, která na ní postupně vznikla k začátku $n$-tého taktu. V každé konfiguraci je vždy pod prověřovanou buňkou zapsán stavový kod, který je vysílán do logického bloku na začátku tohoto taktu. Prvá konfigurace je zobrazena na *obr. 7.*



*Obr. 8*

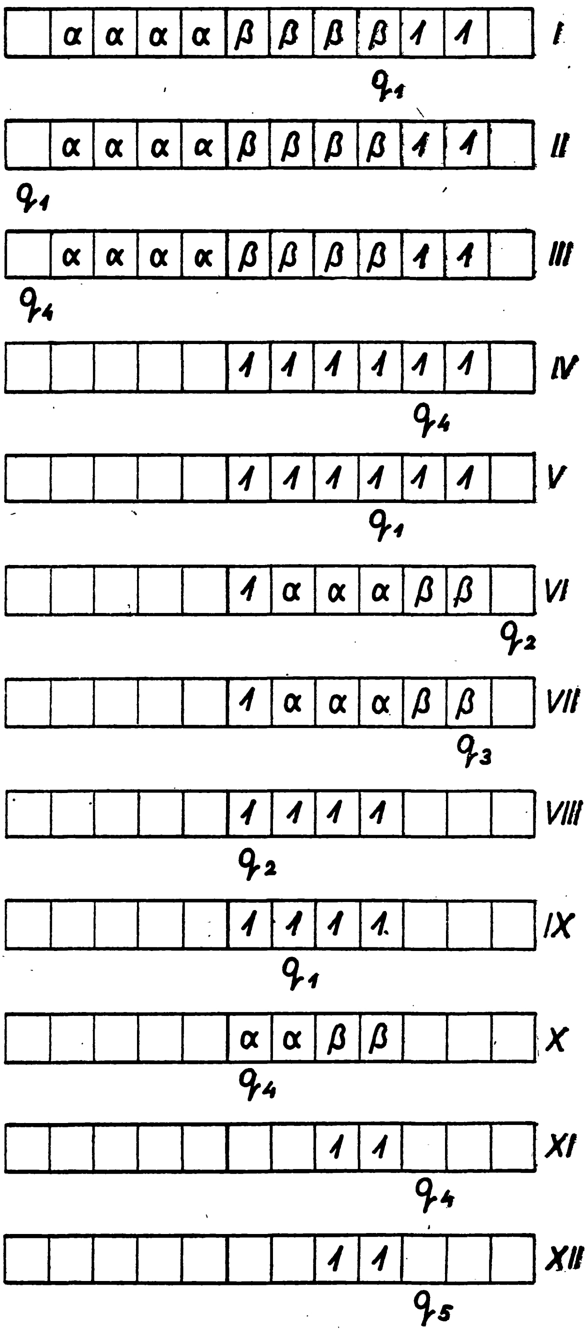
Ve srovnávacím cyklu jsou zúčastněny pouze stavy $q_1$ a $q_2$.

v cyklu odčítacím jsou zúčastněny jen stavy $q_3$ a $q_4$. Stav $q_5$ je stavem pracovního klidu čili stavem, kdy sestroj zastavuje. Objeví-li-se tento stav, znamená to v podstatě totéž jako vydání signálu o tom, že bylo již dosaženo hledaného výsledku.

Popišme si nyní podrobně celý proces.

Stroj nejprve srovnává čísla, zobrazená na pásce, aby zjistil, které z nich je větší. Při tom pracuje v podstatě stejně jako člověk, který má za úkol srovnat dvě dlouhé posloupnosti jedniček, které se nedají na první pohled přehlédnout. V takovémto případě si člověk postupně označuje určitým grafickým způsobem každou jedničku v obou posloupnostech (na př. odškrtává si je), takže pak po proběhnutí celé jedné posloupnosti snadno zjistí, která z obou posloupností má větší počet jedniček. Stroj postupuje tak, že nahradí nejprve výchozí jedničku (v našem případě poslední jedničku prvého čísla) symbolem $\alpha$, pak zamění první jedničku druhého čísla symbolem $\beta$. Nato se vrátí k jedničkám prvého čísla a zamění ještě jednu z nich symbolem $\alpha$; potom zamění další symetrickou jedničku druhého čísla symbolem $\beta$ atd.

V prvních čtyřech taktech tedy na pásce stroje vznikají konfigurace podle *obr. 8*, odkud je patrno, že stroj nakonec označil po jedné jedničce každého čísla, to jest poslední jedničku prvého čísla a první jedničku druhého čísla. Po provedení posunu směrem vlevo se označí další nejbližší jednička prvého čísla. Po několika dalších taktech,




Obr. 9

kdy se současně též označují symetrické jedničky druhého čísla, vzniká na pásce stroje konfigurace I. z *obr. 9*, z níž poznáváme, že prvé číslo bylo už úplně „vyčerpáno", avšak druhé ještě nikoli. Protože na levé straně stroj už nenašel žádnou další jedničku, vzniká konfigurace II., kdy je srovnávací cyklus skončen. Další takt již dává konfiguraci III. za účasti stavu $q_4$, kdy začíná odčítací cyklus. Ze 4. řádku funkčního schematu na *obr. 5* poznáváme, že nyní začíná posun směrem vpravo, při čemž se symboly $\alpha$ nahrazují nulami (t. j. všechna $\alpha$ se vymazávají) a všechny symboly $\beta$ se zaměňují jedničkami. Jakmile byl poslední symbol $\beta$ zaměněn jedničkou, objevuje se na pásce stroje konfigurace IV., a dále hned konfigurace V.

Tak vidíme, že po skončení srovnávacího cyklu proběhl i cyklus odčítání prvého čísla od čísla druhého, jehož výsledkem bylo, že menší číslo $m$ bylo smazáno a větší číslo $n$ bylo rozděleno ve dvě části: $m$ a $n - m$; přitom za výchozí jedničku k dalšímu vyšetřování se bere poslední jednička prvého z těchto dvou čísel. To vše znamená, že původní úloha pro dvojici čísel $m$, $n$ byla převedena na tutéž úlohu pro čísla $m$, $n - m$.

V dalším postupu se musí opět opakovat srovnávací cyklus, končící nyní vyčerpáním druhého čísla (t. j. čísla napravo), které je tentokrát menší než prvé. To nastane v tom okamžiku, kdy stroj po označení tří jedniček prvého čísla nenajde již žádnou další symetrickou jedničku druhého čísla, a na pásce stroje vznikne konfigurace VI. V příštím taktu dostaneme konfiguraci VII., čímž začíná cyklus odčítání druhého čísla od čísla prvého. Přitom se vymazávají všechny symboly $\beta$, a všechny symboly $\alpha$ se nahradí jedničkami. Po provení záměny nejkrajnějšího levého symbolu $\alpha$ jedničkou se objeví na pásce stroje již konfigurace VIII., a pak hned konfigurace IX., čímž je odčítací cyklus skončen. Nyní by měl opět následovat srovnávací cyklus, dále pak opět cyklus odčítací atd. až do té doby, kdy dostaneme dvě stejná čísla (v našem příkladě to právě nastalo). Po skončení posledního srovnávacího cyklu končí též celý proces. V našem příkladě dostaneme po srovnání a po odečtení konfiguraci XI. a nakonec výslednou konfiguraci XII., po níž se stroj zastavuje.

Na základě podrobnějšího vyšetřování se ukazuje, že všechny v matematice dosud známé algoritmy je možno realisovat v určitém stroji Turingově. Tato významná skutečnost spolu s ostatními výše uvedenými okolnostmi ospravedlňuje přijetí základní definice theorie algoritmů, podle níž se dosud nepřesný pojem algoritmu ztotožňuje s přesným pojmem funkčního schematu příslušného Turingova stroje.

## § 7.

Zpřesnění pojmu algoritmu, k němuž jsme právě došli, umožňuje nám nyní definovat každý algoritmus (a spolu s ním i každý stroj, kterým se tento algoritmus realisuje) pomocí tak zvané šifry algoritmu (čili šifry příslušného stroje Turingova). K této šifře dospějeme touto úpravou funkčního schematu stroje:

1. Místo dvojrozměrného zobrazení funkčního schematu stroje (t. j. tabulky o $m$-řádcích a $k$-sloupcích) užijeme jednorozměrného zobrazení v jednom řádku. K tomu stačí, přepíšeme-li všechny pětice symbolů (v počtu $m \cdot k$) vedle sebe, při čemž prvý symbol pětice označuje vždy řádek tabulky, druhý označuje sloupec tabulky a zbývající tři jsou symboly zapsané v tom okénku tabulky, které je pronikem uvedeného řádku a sloupce. Na příklad místo tabulky na *obr. 5* tak vzniká jediný řádek symbolů

$$q_1 1\, \alpha\, P_0 q_2 q_1\, \alpha\alpha\, \mathrm{P}_{-1} q_1 q_1\, \beta\beta\, \mathrm{P}_{-1} q_1 \ldots \quad (*)$$

Je samozřejmé, že z tohoto zápisu můžeme opět kdykoli znovu sestavit původní tabulku.

2. Pro funkční schema stroje nejsou vůbec podstatné grafické zápisy těch symbolů, které v něm vystupují. Důležité je, aby různé objekty byly označeny různými symboly. Proto můžeme každý symbol v řádku (*) zakodovat na příklad jistou posloupností jedniček a nul tak, že různé symboly zakodujeme různými posloupnostmi; řádek (*) pak přejde v jiný řádek (**), v němž jsou jen jedničky a nuly. K tomu, aby bylo možno přetvořit řádek (*) v řádek (**), je nutno zvolit takový kod,

a) aby bylo možno řádek (**) zcela jednoznačně rozčlenit na posloupnosti, představující kody jednotlivých symbolů z řádku (*);

b) aby každý ze symbolů $P_0 P_{-1}, P_1$, které mají zvláštní určení (nulový posun, posun vlevo a vpravo), bylo možno dobře rozlišit.

Oběma podmínkám lze vyhovět tímto kodováním:

I. Každý symbol se vyznačí posloupností tvaru

$$\underbrace{100\ldots01}_{i\text{-nul}} \qquad (i = 1, 2, \ldots, m + k + 3)\,,$$

takže počet všech možných symbolů je $m + k + 3$.

II. Kody 101, 1001, 10001 se vyhradí jednou provždy znakům $P_0$, $P_{-1}$, $P_1$.

III. Ostatní symboly v počtu $m + k$ se pak zakodují posloupnostmi, které obsahují 4, 5, 6, ..., $m + k + 3$ nul. Vzájemné přiřazení kodů a symbolů můžeme provést libovolným způsobem. Na příklad můžeme přiřadit nejkratší kod prvému symbolu v řádku (*), různému od symbolů $P_0$, $P_{-1}$, $P_1$, a další kody pak postupně dalším symbolům z řádku (*).

V tomto přiřazení se pak řádek (*) zakoduje touto posloupností:

$$\underbrace{100001}_{q_1}\quad \underbrace{1000001}_{1}\quad \underbrace{10000001}_{\alpha}\quad \underbrace{101}_{P_0}\quad \underbrace{100000001}_{q_2}\quad \underbrace{100001}_{q_1}\ldots \qquad (**)$$

Tuto posloupnost složenou z jedniček a z nul budeme nazývat *šifrou funkčního schematu stroje*. Původní tvar schematu můžeme podle dané šifry vždy opět snadno vytvořit. Proto můžeme vždycky schema stroje (nebo algoritmu) dokonale definovat jeho šifrou.

*Universální stroj Turingův.* V dosavadních výkladech jsme vycházeli z předpokladu, že různé algoritmy se realisují též v různých Turingových strojích, jejichž funkční schemata se navzájem liší. Je však možné sestrojit universální stroj Turingův, který je schopen realisovat libovolný algoritmus, to znamená, že je schopen vykonat práci libovolného stroje Turingova. K objasnění této věci si představme tento experiment: Předpokládejme, že na pásku stroje $A$ je vložena počáteční informace $B$, a nechť je někomu uloženo, aby ukázal, v jakou informaci přetváří stroj $A$ tuto informaci. Je-li dotyčná osoba seznámena s principem práce Turingova stroje, stačí sděliti jí kromě počáteční informace ještě funkční schema $A$ nebo šifru stroje. Tato osoba pak při řešení dané úlohy napodobuje práci stroje, a může dospět k stejnému výsledku jako stroj. To znamená, že tento člověk je schopen vykonat práci libovolného Turingova stroje, pokud je mu známa jeho šifra. Samotný proces napodobení práce stroje podle funkčního schematu může býti předepsán ve tvaru přesného předpisu, který je možno sdělit i člověku, který vůbec není seznámen s theorií Turingových strojů. Představme si nyní, že místo člověka, který má k disposici tento „algoritmus napodobení", postavíme Turingův stroj. Tento stroj však je právě universálním Turingovým strojem. Jestliže nějaký speciální Turingův stroj *A* řeší určitou úlohu, pak také universální Turingův stroj je schopen tuto úlohu rozřešit ovšem za předpokladu, že kromě vstupních dat úlohy bude na jeho pásku vložena i šifra stroje čili šifra příslušného algoritmu. V souvislosti s tím je možno užít funkčního schematu čili šifry *A* určitého algoritmu dvojím způsobem, plynoucím ze dvou hledisek:

1. Šifra *A* popisuje funkci logického bloku speciálního stroje Turingova, který realisuje zašifrovaný algoritmus. To je hledisko, z něhož jsme dosud vycházeli.
2. Šifra *A* popisuje instrukci, vloženou na pásku universálního stroje Turingova, kde má sloužit k uskutečnění příslušného algoritmu.

Soudobé elektronické samočinné počítací stroje se staví stejně jako universální stroje Turingovy, do jejichž paměti se společně se vstupními daty úlohy vkládají též instrukce k jejímu řešení.

Závěrem je možno říci, že je vždycky možno (alespoň theoreticky) zkonstruovat takový samočinný počítač, který je schopen realisovat algoritmus k řešení určité dané úlohy nebo dokonce počítač, který je schopen realisovat algoritmy k řešení většího počtu úloh téhož typu.

Na druhé straně však theorie algoritmů ukazuje, že není možné každou matematickou nebo logickou úlohu algoritmisovat, což úzce souvisí s tím, že lidské myšlení nelze vždy úplně spoutat do tuhých algoritmických předpisů.

Je nesporně velmi významným pokrokem, můžeme-li určitou úlohu nebo určitou třídu úloh téhož typu v celém jejich rozsahu řešit pomocí samočinného počítacího stroje, i když zde zůstávají úlohy, které nelze v celém rozsahu řešit strojově a u nichž se musíme spokojit v podstatě klasickým způsobem řešení.

*František Martan*

# POZNÁMKY K EXPERIMENTÁLNÍMU STUDIU FYSIKÁLNÍCH VLASTNOSTÍ VULKANISÁTŮ KAUČUKU

VÁCLAV MÜLLER

Hlavní úlohou experimentálního studia vulkanisátů kaučuku je určit z jejich uvažovaného počtu (vulkanisáty různé složením, dobou vulkanisace resp. jinými specifickými vlastnostmi) ty vulkanisáty, které se hodí svými fysikálními i chemickými vlastnostmi nejlépe pro předpokládané použití v praxi.

Při výběru vhodných experimentálních metod je nutno hlavně dbáti toho, jak dalece lze zachytit použitím zvolené metody základní mechanické resp. elektrické vlastnosti vulkanisátů, jaký je vztah mezi výsledky získanými různými měrnými metodami a do jaké míry lze tyto vztahy vysvětlit s přihlédnutím ke stejným základním vlastnostem zkoumaného vulkanisátu. Dále záleží na tom, jak přesně lze zjišťovat rozptyly výsledků fysikálních zkoušek a s jakou matematickou jistotou lze činit výsledná tvrzení z vyplývajících výsledků.

Pro uspokojivé objasnění mechanických vlastností vulkanisátů kaučuku je nutno se zmínit stručně o Alfreyově modelové theorii, pokud je to nutné k posouzení zkušebních metod v rámci tohoto článku.