Pokroky matematiky, fyziky a astronomie

Stephen A. Cook

Prehl'ad teórie výpočtovej složitosti

Pokroky matematiky, fyziky a astronomie, Vol. 32 (1987), No. 1, 12--29

Persistent URL: http://dml.cz/dmlcz/139874

Prehľad teórie výpočtovej zložitosti

Stephen A. Cook, Toronto

Stephen Arthur Cook je profesorom matematickej informatiky na Univerzite v Toronte a držiteľom Turingovej ceny za rok 1982. Túto cenu udeľuje spoločnosť ACM (Association for Computing Machinery) na počesť A. M. Turinga ako ocenenie najväčšieho prínosu do výpočtových vied. Držiteľ ceny pri preberaní prednesie prednášku z tej oblasti, v ktorej pracuje. Profesor Cook je zakladateľom teórie NP-úplnosti ako najvýznamnejšieho smeru výskumu v matematickej informatike za posledné desaťročie. Jeho ďalšie práce sa týkajú najmä teórie zložitosti, vzťahu času a priestoru vo výpočtoch a logiky programovacích jazykov.

Abstrakt: V článku sa podáva historický prehľad teórie výpočtovej zložitosti. Dôraz sa kladie na základné otázky určenia vlastnej výpočtovej zložitosti problému a dokazovania horných a dolných odhadov zložitosti problémov. Uvažujú sa aj pravdepodobnostné a paralelné výpočty.

Toto je druhá prednáška držiteľa Turingovej ceny o výpočtovej zložitosti. Prvú mal Michael Rabin v r. 1976*). Keď si teraz čítam Rabinov skvelý článok [62], udivuje ma, okrem iného, koľko aktivity sa odvtedy vynaložilo v tejto oblasti. V tomto stručnom prehľade chcem spomenúť pre mňa najdôležitejšie a najzaujímavejšie výsledky asi od r. 1960, kedy sa začala rozvíjať táto teória. V takej veľkej oblasti je výber materiálu nevyhnutne trochu osobný; ale dúfam, že som zaradil články, ktoré sú z každého hľadiska fundamentálne.

1. Prvé články

Predhistória tejto teórie začína vlastne u Alana Turinga. Vo svojom článku *O vypočítateľných číslach s použitím na rozhodovací problém* [85] zaviedol v r. 1937 Turing svoj známy Turingov stroj, ktorý umožňoval (dovtedy) najvhodnejšiu formalizáciu pojmu efektívne (alebo algoritmicky) vypočítateľnej funkcie. Keď už bol tento pojem precízne zvládnutý, bolo možné dokazovať, čo nemožno vypočítať na počítači. Turing v tom istom článku dokázal, že žiadny algoritmus (napr. Turingov stroj) nemôže pre

*) Michael Rabin a Dana Scott dostali Turingovu cenu v r. 1976.

S. A. Cook: *An overview of computational complexity*, Communications of the ACM, Vol. *26*, June 1983, No. 6, pp. 400—408.

danú ľubovoľnú formulu predikátového počtu v konečnom počte krokov rozhodnúť, či je táto formula splniteľná.

Akonáhle bola vyvinutá teória vysvetľujúca, ktoré problémy sa dajú a ktoré nedajú riešiť na počítači, bolo prirodzené sa pýtať na relatívnu výpočtovú obtiažnosť vypočítateľných funkcií. A práve to je predmetom teórie výpočtovej zložitosti. Rabin [59, 60], ako jeden z prvých (v r. 1960), explicitne nastolil túto všeobecnú otázku: čo to znamená, ak povieme, že f sa ťažšie vypočíta ako g? Rabin navrhol axiomatický rámec, ktorý poslúžil ako základ pre abstraktnú teóriu zložitosti rozvinutú Blumom [6] a ďalšími.

Svoj vplyv mal i druhý raný (1965) článok *O výpočtovej zložitosti algoritmov* [37] od J. Hartmanisa a R. E. Stearnsa*). Tento článok sa široko študoval a dal teórii zložitosti jej názov. Bol v ňom uvedený dôležitý pojem miery zložitosti definovaný ako čas výpočtu na viacpáskových Turingových strojoch a boli dokázané vety o hierarchii. V článku bola tiež položená zaujímavá, dodnes otvorená otázka. Či je ľubovoľné iracionálne číslo (také ako $\sqrt{2}$) vypočítateľné v reálnom čase, t. j. či existuje Turingov stroj, ktorý tlačí desiatkový rozvoj tohto čísla rýchlosťou jedna číslica za 100 krokov donekonečna.

Tretím zakladajúcim článkom (1965) bol článok Alana Cobhama *Vlastná výpočtová obtiažnosť funkcií* [15]. Cobham zdôrazňoval slovo vlastná (intrinsic), zaujímal sa teda o strojovo nezávislú teóriu. Položil otázku, či násobenie je ťažšie ako sčítanie a veril, že ju nemožno zodpovedať, kým sa dôkladne nevybuduje teória. Cobham tiež definoval a charakterizoval dôležitú triedu funkcií, ktorú nazval $\mathscr{L}$: sú to funkcie na prirodzených číslach, ktoré sa dajú vypočítať v čase ohraničenom polynómom v dĺžke vstupu, ak sa tento čas vyjadrí desiatkovo.

Tri ďalšie články, ktoré ovplyvnili spomínaných autorov ako i ďalších pracujúcich v teórii zložitosti (včítane mňa) sú od Yamadu [91], Benneta [4] a Ritchieho [66]. Je zaujímavé, že Rabin, Stearns, Bennet a Ritchie študovali v Princetone približne v rovnakom čase.

2. Počiatočné zdroje a koncepcie

Viacerí z raných autorov sa zaoberali otázkou, čo je pravou mierou zložitosti. Väčšina uvádzala ako prirodzenú odpoveď čas alebo priestor výpočtu, ale neboli presvedčení, či tieto miery sú jediné alebo či sú tie pravé. Napr. Cobham [15] navrhol „... nejaká miera, ktorá má vzťah ku fyzikálnemu pojmu práca (možno) vedie k najuspokojivejšej analýze". Rabin [60] zaviedol axiómy, ktoré by mala spĺňať miera zložitosti. S retrospektívou 20ročnej skúsenosti si teraz myslím, že je jasné, že čas a priestor — najmä čas — patria určite medzi najdôležitejšie miery zložitosti. Zdá sa, že prvoradou stránkou vhodnou na zhodnotenie efektívnosti algoritmu je čas chodu. Avšak v súčasnosti je stále jasnejšie, že paralelný čas a veľkosť hardwaru sú tiež dôležité miery zložitosti (pozri časť 6).

Ďalšou dôležitou mierou zložitosti, ktorá siaha v istej forme prinajmenšom k Shanno-

*) Pozri ďalší Hartmanisov článok [36] kvôli niektorým zaujímavým spomienkam.

novi [74] (1949), je zložitosť booleovských schém (kombinačná zložitosť). Tu je výhodné, ak sa predpokladá, že skúmaná funkcia f zobrazuje konečné reťazce bitov do konečných reťazcov bitov a zložitosť $C(n)$ funkcie f je veľkosť najmenšej booleovskej schémy, ktorá počíta f pre všetky vstupy dĺžky n. Táto veľmi prirodzená miera má úzky vzťah k času výpočtu (pozri [57a], [57b], [68b]) a má dobre rozvinutú vlastnú teóriu (pozri Savage [68a]).

Ďalšou otázkou, ktorú položil Cobham [15], je otázka, čo tvorí „krok" výpočtu. To vlastne znamená pýtať sa, aký je správny model počítača na meranie času výpočtu algoritmu. V literatúre sa bežne používajú viacpáskové Turingove stroje, ale tie majú z hľadiska efektívnej implementácie algoritmov umelé obmedzenia. Napr. nie je žiadny nutný dôvod, prečo by pamäťové médiá mali byť lineárne pásky. Prečo nie rovinné polia alebo stromy? Prečo nedovoliť pamäť s náhodným prístupom?

Od r. 1960 bolo naozaj navrhnutých nemálo modelov počítačov. Keďže reálne počítače majú pamäte s náhodným prístupom, zdá sa prirodzené dovoliť ich aj v modeli. Ale ako to urobiť, to je spletitá otázka. Ak stroj môže ukladať celé čísla v jednom kroku, musí existovať akési ohraničenie na ich veľkosť. (Ak sa číslo 2 umocní stokrát, je výsledok 2^{100} bitov, čo by sa neuložilo do všetkých svetových pamäťových médií.) V [19] som navrhol počítacie RAM-y*), v ktorých sa vždy, keď sa ukladá alebo vyberá číslo x, pripočítava k cene (počtu krokov) okolo $\log [x]$. To funguje, ale nie je to úplne presvedčivé. Populárnejší model s náhodným prístupom je ten, ktorý používajú Aho, Hopcroft a Ullman v [3], v ktorom každá operácia obsahujúca celé číslo má jednotkovú cenu, ale celé čísla nemôžu byť neodôvodnene veľké (napr. ich veľkosť môže byť ohraničená fixovaným polynómom v premennej, ktorou bude veľkosť vstupu). Pravdepodobne matematicky najviac uspokojivým modelom je Schönhageho stroj s modifikovaným ukladaním [69], na ktorý sa možno dívať ako na Turingov stroj, ktorý si tvorí vlastnú štruktúru ukladania alebo ako na RAM s jednotkovou cenou, ktorý môže iba kopírovať, pričítať alebo odčítať jednotku alebo ukladať a vyberať v jednom kroku. Schönhageho stroj je nepatrným zovšeobecnením stroja Kolmogorova-Uspenského, navrhnutého omnoho skôr [46] (1958), a zdá sa mi, že reprezentuje najvšeobecnejší stroj vykonávajúci ohraničené množstvo práce v jednom kroku, aký môže byť vôbec skonštruovaný. Problém je v tom, že je pravdepodobne trochu príliš silný (pozri časť 3 pod „násobenie veľkých čísiel").

Ak sa vrátim ku Cobhamovej otázke „čo je to krok", myslím, že za posledných 20 rokov sa vyjasnilo, že neexistuje jediná jasná odpoveď. Našťastie, konkurujúce modely počítača sa nelíšia neohraničene v čase výpočtu. Vo všeobecnosti môže každý simulovať ten ďalší, pričom čas výpočtu je najviac kvadratický (niektoré z prvých argumentov k tomuto efektu sú v [37]). Medzi vedúcimi modelmi s náhodným prístupom sa jedná iba o faktor logaritmu času výpočtu.

To viedlo okolo r. 1965 k vývoju posledného dôležitého pojmu — identifikácii tried

*) RAM je model počítača s náhodným prístupom, ktorý má vstupnú pásku (ktorú môže iba čítať), výstupnú pásku (na ktorú môže iba zapisovať) a pamäť, ktorá sa skladá z postupnosti rovnako dostupných registrov. Činnosť stroja sa riadi programom (súborom inštrukcií vybraných podobne ako v reálnych počítačoch), ktorý je uložený mimo neho. Pozn. prekl.

problémov riešiteľných v čase ohraničenom polynómom v premennej, ktorou je dĺžka vstupu. Algoritmy s polynomiálnym a exponenciálnym časom rozlišoval už v r. 1953 von Neumann [90]. Ale takáto trieda nebola formálne definovaná ani skúmaná, až kým Cobham [15] nezaviedol v r. 1964 (pozri časť 1) triedu funkcií $\mathscr{L}$. Cobham poukázal na to, že jeho trieda bola dobre definovaná nezávisle na vybranom modeli počítača a charakterizoval ju v duchu teórie rekurzívnych funkcií. Myšlienku, že vypočítateľnosť v polynomiálnom čase zhruba zodpovedá praktickej použiteľnosti, prvý raz publikoval Edmonds [27], ktorý nazval algoritmy s polynomiálnym časom „dobrými algoritmami". Terajšie štandardné označenie P pre triedu množín reťazcov rozpoznateľných v polynomiálnom čase zaviedol neskôr Karp [42].

Na začiatku sedemdesiatych rokov sa všeobecne akceptovalo v tejto oblasti ztotožnenie P s prakticky použiteľnými (alebo realizovateľnými) problémami. Nie je hneď zrejmé, prečo by to mala byť pravda, pretože algoritmus, ktorého čas výpočtu je exponenciálny ako $2^{0,0001n}$, je v praxi realizovateľný. Zdá sa však empirickým faktom, že problémy, ktoré sa prirodzene objavujú, nemávajú optimálne algoritmy s takýmito časmi výpočtu.*) Najpozoruhodnejším praktickým algoritmom, ktorý má najhorší čas výpočtu exponenciálny, je simplexový algoritmus pre lineárne programovanie. Smale [75, 76] sa to snažil vysvetliť tak, že chcel ukázať, že v istom zmysle je priemerný čas výpočtu rýchly, ale treba tiež poznamenať, že Chačijan [43] s použitím iného algoritmu ukázal, že lineárne programovanie patrí do triedy P**), takže naša všeobecná téza, že P sa rovná realizovateľným problémom, nie je narušená.

3. Horné odhady času

Značná časť výskumu v matematickej informatike pozostáva z navrhovania a analýzy obrovského množstva efektívnych algoritmov. Dôležité algoritmy (z hľadiska výpočtovej zložitosti) musia byť v nejakom zmysle špeciálne; vo všeobecnosti poskytujú prekvapujúco rýchlu cestu riešenia jednoduchých alebo dôležitých problémov. Nižšie uvádzame prehľad najzaujímavejších z nich vymyslených od r. 1960. (Len tak mimochodom, je zaujímavé zamyslieť sa nad tým, ktoré sú najdôležitejšie algoritmy všetkých čias. Základnými sú iste aritmetické operácie $+$, $-$, $*$ a $:$ nad desiatkovými číslami. Po nich navrhujem ako kandidátov rýchle triedenie a vyhľadávanie, Gaussovu elimináciu, Euklidov algoritmus a simplexovú metódu.)

Parameter n sa vzťahuje na veľkosť vstupu; časové ohraničenia sú ohraničeniami času v najhoršom prípade a týkajú sa viacpáskových Turingových strojov (alebo akéhokoľvek rozumného stroja s náhodným prístupom) okrem toho, kde je uvedené ináč.

1) *Rýchla Fourierova transformácia* [23], ktorá vyžaduje $O(n \cdot \log n)$***) aritmetických operácií, je jedným z najpoužívanejších algoritmov pri vedeckých výpočtoch.

*) Pozri [31], na str. 6—9 je o tom diskusia.

**) O porovnaní tohto algoritmu so simplexovou metódou pozri [100]. Pozn. prekl.

***) Označenie O sa používa pri horných odhadoch zložitosti. $f(n) = O(g(n)) \Leftrightarrow \exists c_0 > 0\ \exists n_0\ \forall n > n_0\ f(n) \leqq c_0 \cdot g(n)$. Pozn. prekl.

2) *Násobenie veľkých čísiel.* Elementárna školská metóda vyžaduje $O(n^2)$ bitových operácií na vynásobenie dvoch čísiel po n číslicach. V r. 1962 Karatsuba a Ofman [41] publikovali metódu vyžadujúcu iba $O(n^{1,59})$ krokov. Krátko nato Toom [84] ukázal, ako sa dá zostrojiť booleovská schéma veľkosti $O(n^{1+\varepsilon})$ pre ľubovoľne malé $\varepsilon > 0$, ktorá realizuje násobenie. V tom čase som bol absolventom na Harvarde a inšpirovaný Cobhamovou otázkou „Je násobenie ťažšie ako sčítanie?" som sa naivne pokúšal dokázať, že násobenie vyžaduje $\Omega(n^2)$*) krokov na viacpáskovom Turingovom stroji. Toomov článok ma značne prekvapil. S pomocou Stala Aanderaa [22] som sa obmedzil na dôkaz, že násobenie vyžaduje $\Omega(n \cdot \log n/(\text{loglog } n)^2)$ krokov**) na spriahnutom Turingovom stroji.***) Vo svojej dizertácii som tiež upozornil na to, že Toomovu metódu možno prispôsobiť na viacpáskové Turingove stroje tak, aby násobili za $O(n^{1+\varepsilon})$ krokov. Som si istý, že Tooma to určite neprekvapilo.

Súčasne najrýchlejší asymptotický čas výpočtu pre násobenie na viacpáskových Turingových strojoch je $O(n \cdot \log n \cdot \text{loglog } n)$ a odvodili ho Schönhage a Strassen [70] (1971) s použitím rýchlej Fourierovej transformácie. Nedávno však Schönhage [69] ukázal pomocou komplikovaných argumentov, že jeho stroje s modifikovaným ukladaním (pozri časť 2) môžu násobiť v čase $O(n)$ (lineárny čas!). Z toho musíme urobiť záver, že buď je násobenie jednoduchšie, ako sme si mysleli, alebo Schönhageho stroje podvádzajú.

3) *Násobenie matíc.* Bežná metóda na násobenie dvoch matíc rozmeru $n \times n$ vyžaduje $n^2(n-1)$ aritmetických operácií a v 50. a 60. rokoch sa vyskytli pokusy dokázať, že táto metóda je optimálna. Bolo preto prekvapením, keď Strassen [81] (1969) publikoval svoju metódu vyžadujúcu iba $4{,}7n^{2,81}$ operácií. Veľa práce sa vynaložilo na redukciu exponentu 2,81 a najlepší súčasne známy čas je $O(n^{2,496})$ operácií od Coppersmitha a Winograda [24]. Je tu stále dosť priestoru na postup, veď najlepší známy dolný odhad je $2n^2 - 1$ (pozri [13]).

4) *Najpočetnejšie párenia vo všeobecne neorientovaných grafoch.*****) Toto bol asi prvý problém, o ktorom bolo explicitne dokázané, že patrí do P a ktorého príslušnosť v P vyžaduje zložitý algoritmus. Tento výsledok predložil Edmonds vo svojom vplyvnom článku [27], v ktorom rozoberal pojem algoritmu s polynomiálnym časom (pozri časť 2). Poukázal aj na to, že jednoduchý pojem zväčšujúcej sa cesty, ktorý stačí pre prípad bipartitných grafov, nefunguje pre všeobecne neorientované grafy.

5) *Rozpoznanie prvočísiel.* Veľkou otázkou je, či tento problém je v P. Inými slovami, či existuje algoritmus, ktorý nám vždy povie, či ľubovoľné číslo o n číslicach je prvočíslo a zastaví sa po počte krokov ohraničenom polynómom v n. Gary Miller [53] (1976) ukázal, že taký algoritmus existuje, ale jeho správnosť závisí na rozšírenej Rieman-

*) Označenie Ω znamená, že ide o dolný odhad zložitosti. $f(n) = \Omega(g(n)) \Leftrightarrow \exists c_1 > 0\ \exists n_1\ \forall n > n_1\ f(n) \geqq c_1 \cdot g(n)$. Pozn. prekl.

**) Tento dolný odhad bol trochu zlepšený. Pozri [56] a [64].

***) Spriahnutý Turingov stroj je viacpáskový Turingov stroj so vstupnou páskou (obsahujúcou vstup), na ktorú sa nesmie zapisovať a čítacia hlava sa po nej môže pohybovať iba vpravo; ak sa môže pohybovať obojsmerne, ide o nespriahnutý Turingov stroj. Pozn. prekl.

****) Pod párením v grafe rozumieme takú podmnožinu množiny hrán grafu, že každý vrchol grafu je incidentný najviac s jednou hranou z tejto podmnožiny. Pozn. prekl.

novej hypotéze. Solovay a Strassen [77] vymysleli rýchly Monte Carlo algoritmus (pozri časť 5) na rozpoznávanie prvočísiel, ale ak vstupné číslo je zložené, je tu malé riziko, že algoritmus chybne povie, že je prvočíslom. Najlepší dokázateľne deterministický algoritmus je známy vďaka Adlemanovi, Pomerancovi a Rumelymu [2]; jeho čas je $n^{O(\log\log n)}$, čiže trochu horší ako polynomiálny. Jeho variácia od H. Cohena a H. W. Lenstru ml. [17] môže bežne spracovávať čísla do 100 desiatkových číslic približne za 45 sekúnd.

V súčasnosti sa o troch dôležitých problémoch ukázalo, že patria do triedy *P*. Prvým je lineárne programovanie, čo ukázal Chačijan [43] v r. 1979 (pozri [55] pre výklad a tiež [101]). Luks [50] v r. 1980 ukázal druhý taký problém: určiť, či dva grafy stupňa najviac d sú izomorfné. (Algoritmus je polynomiálny v počte vrcholov pre pevné d, ale je exponenciálny v d.) Tretím problémom je rozklad polynómov s racionálnymi koeficientami. To ukázali Lenstra, Lenstra a Lovász [48] v r. 1982 pre polynómy v jednej premennej. Ako vyplýva z Kalfotenovho výsledku [39], [40], možno to zovšeobecniť na polynómy v ľubovoľnom fixovanom počte premenných.

4. Dolné odhady

Skutočným orieškom v teórii zložitosti a problémom, ktorý ju posúva mimo analýzy algoritmov, je dokazovanie dolných odhadov zložitosti konkrétnych problémov. Je niečo veľmi uspokojujúce v tom, ak sa dokáže, že problém typu áno-nie sa nedá riešiť za n, n^2 alebo 2^n krokov, bez ohľadu na to, aký algoritmus sa použije. V dokazovaní dolných odhadov sa dosiahlo niekoľko dôležitých úspechov, ale otvorené otázky sú ešte dôležitejšie a tak trochu frustrujúce.

Všetky významné dolné odhady času a priestoru výpočtu sú založené na „diagonalizačných argumentoch". Argumenty diagonalizácie použil Turing a jeho súčasníci na dôkaz, že určité problémy nie sú algoritmicky riešiteľné. Pred r. 1960 boli tiež použité na definovanie hierarchie vypočítateľných 0-1 funkcií.*) Rabin [60] v r. 1960 dokázal, že pre každú rozumnú mieru zložitosti, akou je čas výpočtu alebo priestor (pamäť), umožní zvýšenie prípustného času alebo priestoru ap. vždy vypočítať viac 0-1 funkcií. V približne rovnakom čase Ritchie vo svojej dizertácii [65] definoval pomocou množstva prípustného priestoru špecifickú hierarchiu funkcií (ktorá, ako ukázal, je netriviálna pre 0-1 funkcie). O niečo neskôr Hartmanis a Stearns [37] rozviedli detailnejšie Rabinov výsledok pre čas na viacpáskových Turingových strojoch a Stearns, Hartmanis a Lewis [78] pre priestor.

4.1. Dôkazy nerealizovateľnosti prirodzene rozhodnuteľných problémov

Vyššie spomenuté hierarchické výsledky dávali dolné odhady času a priestoru potrebného na výpočet konkrétnych funkcií, ale všetky tieto funkcie sa zdali „vykonštruované". Napr. je ľahko vidieť, že funkcia $f(x, y)$, ktorá dáva prvú číslicu výstupu stroja x

*) Pozri napr. Grzegorczyk [35].

na vstupe y po $(|x| + |y|)^2$ krokoch, nemôže byť vypočítaná v čase $(|x| + |y|)^2$. Netriviálny dolný odhad pre „prirodzený" problém („prirodzený" v zmysle zaujímavosti, a to nielen pre výpočtové stroje) na všeobecných modeloch výpočtu sa nenašiel skôr ako v r. 1972, kedy Albert Meyer a Larry Stockmeyer [52] dokázali, že problém ekvivalencie pre regulárne výrazy s kvadrátom vyžaduje exponenciálny priestor, a teda exponenciálny čas. Krátko nato Meyer [51] našiel veľmi silný dolný odhad času potrebného na určenie pravdivosti formúl v určitej formálne rozhodnuteľnej teórii nazvanej WSIS (slabá monadická teória 2. rádu s následníkom). Dokázal, že každý počítač, ktorého čas výpočtu je ohraničený fixovaným počtom exponenciálnych funkcií (2^n, 2^{2^n}, $2^{2^{2^n}}$ atd.), nemôže konkrétne rozhodnúť WSIS. Meyerov ašpirant Stockmeyer vypočítal [79], že každá booleovská schéma (rozumej počítač), ktorá správne rozhoduje pravdivosť ľubovoľnej WSIS formuly dĺžky 616 symbolov, musí mať viac ako 10^{123} logických prvkov. Číslo 10^{123} bolo vybraté ako počet protónov, ktoré by mohli vyplniť celý známy vesmír. To je veľmi presvedčivý dôkaz nerealizovateľnosti.

Počnúc Meyerom a Stockmeyerom objavilo sa množstvo dolných odhadov zložitosti rozhodnuteľných formálnych teórií (pozri [29] a [80] pre ich zhrnutie). Jeden z najzaujímavejších je dvojito exponenciálny dolný odhad času potrebného na rozhodnuteľnosť Presburgerovej aritmetiky (teória prirodzených čísiel so sčítaním). To nie je ďaleko od najlepšieho známeho horného odhadu času pre túto teóriu, ktorý je trojito exponenciálny [54]. Najlepší horný odhad pre príslušný priestor je dvojito exponenciálny [29].

Napriek uvedeným úspechom je bilancia v dokazovaní dolných odhadov pre problémy menšej zložitosti veľmi zlá. Skutočne, nie je známy žiadny nelineárny dolný odhad času na univerzálnom výpočtovom modeli pre nijaký prirodzený problém v NP (pozri časť 4.4.), špeciálne ani pre jeden z 300 problémov vypísaných v [31].*) Samozrejme, diagonalizačnými argumentami sa dá dokázať existencia problémov v NP, ktoré si vyžadujú čas n^k pre každé fixované k. Avšak v prípade priestorových dolných odhadov nevieme ani, ako dokázať existenciu NP problémov, ktoré sa nedajú riešiť v priestore $O(\log n)$ na nespriahnutom Turingovom stroji (pozri časť 4.3.). To je pravda napriek tomu, že najlepšie známe priestorové horné odhady sú v mnohých prirodzených prípadoch v podstate lineárne v n.

4.2. Štrukturované dolné odhady

Hoci zatiaľ máme málo úspechov v dokazovaní zaujímavých dolných odhadov pre konkrétne problémy na univerzálnom modeli počítača, máme naozaj zaujímavé výsledky pre „štrukturované" modely. Pojem „štrukturované" zaviedol Borodin [9] pre počítače, ktoré sú zúžené na určité operácie vhodné na skúmaný problém. Jednoduchým príkladom je problém triedenia n čísiel. Bez veľkých ťažkostí sa dá dokázať (pozri [44]), že triedenie vyžaduje $n \cdot \log n$ porovnaní za predpokladu, že jedinou operáciou, ktorú

*) To už dnes nie je pravda. Najprv v r. 1983 dokázali Ďuriš, Galil, Paul a Schnitger dolný odhad $\Omega(n \cdot \log n)$ pre čas rozpoznávania konkrétneho jazyka na jednopáskovom nedeterministickom Turingovom stroji [93] a neskôr bol dokázaný dolný odhad času $\Omega(n^2)$ na tom istom modeli pre tri ďalšie problémy [94, 95, 96]. Pozn. prekl.

môže počítač robiť so vstupmi, je porovnávať ich po dvojiciach. Tento dolný odhad nehovorí nič o Turingových strojoch alebo booleovských schémach, ale bol rozšírený na RAM-y s jednotkovou cenou za predpokladu, že je zakázané delenie.

Druhý veľmi elegantný štrukturovaný dolný odhad podľa Strassena [82] (1973) tvrdí, že polynomiálna interpolácia, tj. nájdenie polynómu stupňa $n - 1$, ktorý prechádza n danými bodmi, vyžaduje $\Omega(n \,.\, \log n)$ násobení, ak sú povolené iba aritmetické operácie. Pritom je zaujímavé to, že Strassenov pôvodný dôkaz závisí na Bezoutovej teoréme, čo je hlboký výsledok v algebraickej geometrii. Nedávno Baur a Strassen [83] rozšírili tento dolný odhad tým, že ukázali, že aj výpočet stredného koeficientu interpolačného polynómu cez n bodov vyžaduje $\Omega(n \,.\, \log n)$ násobení.

Príťažlivosť všetkých týchto štrukturovaných výsledkov spočíva v tom, že dolné odhady sú blízke najlepším známym horným odhadom *), a najlepšie známé algoritmy môžu byť implementované na štrukturovaných modeloch, pre ktoré platia tieto dolné odhady. (Poznamenajme, že radixové triedenie (alebo triedenie grupovaním), o ktorom sa niekedy hovorí, že má lineárny čas, reálne vyžaduje aspoň $n \,.\, \log n$ krokov, ak sa predpokladá, že vstupné čísla majú dostatok číslic na to, aby všetky mohli byť rôzne.)

4.3. Dolné odhady súčinu času a priestoru

Ďalšou cestou zo slepej uličky dokazovania dolných odhadov času a priestoru je dokazovanie dolných odhadov času za predpokladu malého priestoru. Prvý takýto výsledok dokázal Cobham [16] v r. 1966, keď ukázal, že súčin času a priestoru pre rozpoznanie úplných štvorcov o n číslicliach na nespriahnutom Turingovom stroji musí byť $\Omega(n^2)$. (To isté platí pre n-symbolové palindromy.) Pritom je vstup napísaný na dvojsmernej iba čítanej vstupnej páske a použitý priestor sa podľa definície rovná počtu štvorcov prezeraných pracovnými páskami, ktoré má k dispozícii Turingov stroj. Ak je teda napr. priestor ohraničený na $O(\log^3 n)$ (čo je viac než dostatočné), potom čas musí byť aspoň $\Omega(n^2/\log^3 n)$ krokov.

Slabinou Cobhamovho výsledku je to, že hoci nespriahnutý Turingov stroj je vhodný na meranie času a priestoru jednotlivo, je príliš ohraničujúci, ak sa čas a priestor uvažujú spolu. Napr. palindromy môžu byť zrejme rozpoznané za $2n$ krokov a v konštantnom priestore, ak dve hlavy môžu súčasne prehliadať vstupnú pásku. Borodin a ja [10] sme čiastočne napravili túto slabinu, keď sme dokázali, že triedenie n prirodzených čísiel od 1 do n^2 vyžaduje časopriestorový súčin $\Omega(n^2/\log n)$. Dôkaz možno použiť na každý „univerzálny sekvenčný stroj" včítane nespriahnutých Turingových strojov s viacerými vstupnými hlavami alebo dokonca s ľubovoľným prístupom ku vstupnej páske. Bohužiaľ pre náš dôkaz je rozhodujúce, že triedenie vyžaduje veľa výstupných bitov a zostáva zaujímavou otvorenou otázkou, či podobný dolný odhad možno dokázať a uplatniť na problém rozpoznania množiny, akým je napr. rozpoznanie, či všetkých n vstupných čísiel je rôznych. Náš dolný odhad pre triedenie bol nedávno trochu zlepšený v [64].**)

*) Horné odhady pre interpoláciu uvádzajú Borodin a Munro v [12].

**) Možno ešte doplniť výsledok $\Omega(n^3)$ pre súčin druhej mocniny času a priestoru pri rozpoznávaní konkrétneho jazyka na k hlavových viacpáskových Turingových strojoch [97] a výsledok $\Omega(n^2)$ ($\Omega(n^{5/3})$) pre súčin času, priestoru a paralelizmu na jednohlavovom (viachlavovom) Turingovom stroji [98]. Pozn. prekl.

4.4. *NP*-úplnosť

Teória *NP*-úplnosti je určite najdôležitejšou vývojovou etapou v teórii výpočtovej zložitosti. Nebudem sa tu o nej rozširovať, pretože je teraz už dobre známa a je i predmetom učebníc. Naozaj skvelým miestom, kde si možno o nej prečítať, je kniha Gareya a Johnsona [31].

Trieda *NP* pozostáva zo všetkých množín rozpoznateľných v polynomiálnom čase na nedeterministickom Turingovom stroji. Pokiaľ viem, prvýkrát definoval matematicky ekvivalentnú triedu James Bennet v r. 1962 vo svojej doktorskej dizertácii. Bennet používal pre svoju triedu názov „rozšírené pozitívne rudimentárne relácie" a v definícii logické kvantifikátory namiesto počítacích strojov. Čítal som túto časť jeho práce a zistil som, že jeho triedu možno charakterizovat ako teraz už známu definíciu *NP*. Vo svojom článku z r. 1971 [18] som používal označenie $\mathscr{L}^+$ (podľa Cobhamovej triedy $\mathscr{L}$) a Karp dal tejto triede teraz akceptované meno *NP* v článku z r. 1972 [42]. Medzitým Edmonds v r. 1965 [28] celkom nezávisle na formálnom vývoji hovoril neformálne o problémoch s „dobrou charakterizáciou", čo je pojem v podstate ekvivalentný *NP*.

V r. 1971 [18] som zaviedol pojem *NP*-úplnosti a dokázal som, že 3-splniteľnosť a problém podgrafu sú *NP*-úplné.*) O rok neskôr dokázal Karp [42], že ďalších 21 problémov je *NP*-úplných, čím pôsobivo demonštroval dôležitosť veci. Nezávisle na tom a o niečo neskôr Leonid Levin [49] v ZSSR (teraz na Bostonskej univerzite) definoval podobný a silnejší pojem a dokázal, že 6 problémov je úplných v jeho zmysle. V sovietskej literatúre bol známy neformálny pojem „problém preberania" a Levin nazval svoje problémy „univerzálne problémy preberania".

Trieda *NP* obsahuje nesmierne množstvo praktických problémov, ktoré sa objavujú v obchode a priemysle (pozri [31] a tiež [99]). Dôkaz, že *NP* problém je *NP*-úplný, je dôkazom toho, že problém nie je v *P* (nemá deterministický algoritmus s polynomiálnym časom), iba ak by každý *NP* problém bol v *P*. Pretože splnenie poslednej podmienky by revolucionalizovalo matematickú informatiku, praktickým dôsledkom *NP*-úplnosti je dolný odhad. To je dôvod, prečo som zahrnul tieto otázky do časti o dolných odhodoch.

4.5. #*P*-úplnosť

Pojem *NP*-úplnosti sa vzťahuje na množiny a dôkaz, že množina je *NP*-úplná, sa obyčajne interpretuje ako dôkaz, že je prakticky nerealizovateľná. Existuje však veľa očividne nerealizovateľných funkcií, pre ktoré sa žiadny dôkaz *NP*-úplnosti nezdá relevantný. Leslie Valiant [86, 87] definoval pojem #*P*-úplnosti, aby pomohol napraviť túto situáciu. Dôkaz, že funkcia je #*P*-úplná, ukazuje, že sa zjavne nedá prakticky vypočítať, rovnako ako dôkaz, že množina je *NP*-úplná ukazuje, že sa zjavne prakticky

*) Problém splniteľnosti pre booleovské výrazy je problém určiť, či existuje také priradenie núl a jednotiek premenným vo výraze, že tento výraz nadobúda hodnotu 1. Špeciálne 3-splniteľnosť je splniteľnosť pre booleovské výrazy vyjadrené v tvare konjunktívnej normálnej formy, pričom žiadna elementárna disjunkcia neobsahuje viac ako 3 premenné. Pod problémom podgrafu myslí autor problém, či zadaný graf obsahuje kliku, t. j. úplný podgraf. Pozn. prekl.

nedá rozpoznať; totiž, ak je $\#P$-úplná funkcia vypočítateľná v polynomiálnom čase, potom $P = NP$.

Valiant ukázal viacero príkladov $\#P$-úplných funkcií, ale pravdepodobne najzaujímavejším je permanent celočíselnej matice. Permanent má definíciu formálne podobnú ako determinant*), ale kým determinant sa dá ľahko vypočítať Gaussovou eliminációu, z mnohých pokusov o nájdenie vhodných spôsobov na výpočet permanentu za posledných vyše 100 rokov sa ani jeden nepodaril. Valiant podal prvý presvedčivý dôvod tohto nezdaru, keď dokázal, že permanent je $\#P$-úplný.

5. Pravdepodobnostné algoritmy

Použitie náhodných čísiel na simuláciu alebo aproximáciu náhodných procesov je veľmi prirodzené a vo výpočtovej praxi bežne zaužívané. Ale myšlienka, že náhodné vstupy môžu byť veľmi užitočné v riešení deterministických kombinatorických problémov sa medzi matematickými informatikmi rozširuje omnoho pomalšie. V tejto časti sa obmedzím na pravdepodobnostné (v zmysle hádzania mincou) algoritmy s polynomiálnym časom, ktoré „riešia" (v rozumnom zmysle) problém, pre ktorý nie je známy žiadny deterministický algoritmus s polynomiálnym časom.

Zdá sa, že prvým takým algoritmom je Berekampov algoritmus [5] z r. 1970 na rozklad polynómu f nad poľom $GF(p)$ obsahujúcom p prvkov. Jeho algoritmus beží v čase, ktorý závisí polynomiálne na stupni polynómu f a log p a s pravdepodobnosťou aspoň jedna polovica nájde správny rozklad f na prvočinitele; inak skončí neúspechom. Pretože algoritmus možno opakovať ľubovoľný počet krát a neúspešné prípady sú všetky nezávislé, v praxi vždy rozloží polynóm v prijateľnom čase.

Pozoruhodnejším príkladom je algoritmus na rozpoznávanie prvočísiel od Solovaya a Strassena [77], podaný v r. 1974. Tento algoritmus beží v čase, ktorý závisí polynomiálne na dĺžke vstupu m a dáva výstup buď „prvočíslo", alebo „zložené číslo". Ak m je naozaj prvočíslo, potom je výstup určite „prvočíslo", ale ak m je zložené, potom s pravdepodobnosťou najviac jedna polovica môže byť odpoveď tiež „prvočíslo". Algoritmus sa môže opakovať ľubovoľný počet krát na vstupe m s nezávislými výsledkami. Ak teda odpoveď je stále „zložené číslo", užívateľ vie, že m je zložené; ak odpoveď je sústavne „prvočíslo", napr. po 100 behoch, má užívateľ dobrý náznak toho, že m je prvočíslo, pretože také výsledky by dalo každé fixované zložené číslo m s veľmi malou pravdepodobnosťou (menej ako 2^{-100}).

Rabin [61] vyvinul odlišný pravdepodobnostný algoritmus s vlastnosťami podobnými predošlému a pri skúškach na počítači zistil, že je veľmi rýchly. Číslo $2^{400} - 593$ bolo za niekoľko minút identifikované ako pravdepodobne prvočíslo.

Jednu zaujímavú aplikáciu pravdepodobnostných testerov prvočísiel navrhli Rivest, Shamir a Adleman [67a] vo svojom medzníkovom článku o kryptosystémoch so známym kľúčom v r. 1978. Ich systém vyžaduje generovanie veľkých 100číslicových náhodných

*) Ak A je matica rozmeru $n \times n$, permanentom sa nazýva veličina $\mathrm{Perm}\, A = \sum \prod_{i=1}^{n} A_{i,\sigma(i)}$, kde sa sumovanie robí cez množinu $n!$ permutácií na $(1, 2, \ldots, n)$. Pozn. prekl.

prvočisiel. Navrhli teda testovať náhodné čísla o 100 číslicach s použitím Solovayovej-Strassenovej metódy, pokým sa nenájde jedno, ktoré je pravdepodobne prvočíslom v zmysle načrtnutom vyššie. Keď už sa nájde 100číslicové náhodné číslo, ktoré je „pravdepodobne prvočíslom" a je dôležité vedieť to naisto, možno ho testovať naisto za 45 sekúnd novým veľmi výkonným deterministickým testerom prvočísiel Cohena a Lenstru [17] spomenutom v časti 3.

Trieda množín rozpoznateľných pravdepodobnostnými algoritmami v polynomiálnom čase v zmysle Solovaya a Strassena je v literatúre známa ako R (alebo niekedy RP). Teda množina patrí do R vtedy a len vtedy, ak je rozpoznateľná pravdepodobnostným algoritmom, ktorý sa vždy zastaví v polynomiálnom čase, nikdy neurobí chybu pre vstupy nepatriace do R a pre každý vstup z R dáva správnu odpoveď pri každom behu s pravdepodobnosťou aspoň $1/2$. Preto množina zložených čísiel je v R a všeobecne $P \subseteq R \subseteq NP$. Existujú ďalšie zaujímavé príklady množín z R, o ktorých nie je známe, či patria do P. Napr. Schwartz [71] ukázal, že množina nesingulárnych matíc, ktorých prvky sú polynómy vo viacerých premenných, patrí do R. Algoritmus vyhodnocuje polynómy v náhodne malých celočíselných hodnotách a vypočítava determinant výsledku. (Determinant zjavne nemožno počítať priamo, pretože počítané polynómy by mali vo všeobecnosti exponenciálne veľa členov).

Je pozoruhodnou otvorenou otázkou, či $R = P$. Je lákavé predpokladať, že áno, s prihliadnutím na filozofické argumenty, podľa ktorých náhodné hádzanie mincou nebude veľmi užitočné, ak sa hľadá odpoveď dobre definovaná ako áno alebo nie. Príbuznou otázkou je, či pravdepodobnostný algoritmus (ak ukážeme, že problém patrí do R) je na všetky praktické účely taký dobrý ako deterministický. Koniec koncov pravdepodobnostné algoritmy môžu bežať s použitím pseudonáhodných číselných generátorov dostupných na väčšine počítačov a pravdepodobnosť chyby 2^{-100} je zanedbateľná. Háčik je v tom, že pseudonáhodný číselný generátor neprodukuje naozaj náhodné čísla a nikto nevie, ako dobre budú fungovať pre daný pravdepodobnostný algoritmus. V skutočnosti skúsenosť ukazuje, že fungujú — zdá sa — dobre. Ale ak vždy fungujú dobre, potom $R = P$, pretože pseudonáhodné čísla sú generované deterministicky, takže skutočná náhodnosť by koniec koncov nepomohla. Ďalšia možnosť je použiť fyzikálny proces, napr. teplotný šum na generovanie náhodných čísiel. Ale je otvorenou otázkou vo filozofii vedy, nakoľko naozaj náhodná môže byť príroda.

Túto časť uzavriem pripomenutím jednej zaujímavej teorémy od Adlemana [1] o triede R. Je zrejmé [57b], že ak množina je v P, tak pre každé n existuje booleovská schéma o veľkosti ohraničenej fixovaným polynómom v n, ktorá určí, či ľubovoľný reťazec dĺžky n patrí do tejto množiny. Adleman dokázal, že to isté platí pre triedu R. Preto napríklad pre každé n existuje malá „počítačová schéma", ktorá správne a rýchlo testuje, či čísla o n číslicach sú prvočíslami. Háčik je v tom, že schémy nie sú uniformné v n; a vskutku, pre prípad 100 číslic nie je prakticky možné načrtnúť, ako zostrojiť takú schému.*)

*) O ďalšej teórii pravdepodobnostných výpočtov pozri Gill [32].

6. Synchrónne paralelné výpočty

S príchodom VLSI technológie, ktorou možno položiť jeden alebo viac procesorov na polcentimetrový čip, je prirodzené rozmýšľať o počítači budúcnosti zloženom z tisícov takých procesorov pracujúcich spolu paralelne na riešení jedného problému. Hoci ešte nebol postavený žiadny príliš veľký univerzálny počítač takéhoto druhu, také projekty sú už na ceste (pozri Schwartz [72]). To motivuje súčasný rozvoj veľmi príjemného odvetvia výpočtovej zložitosti: teóriu synchrónnych paralelných výpočtov veľkého rozsahu, v ktorej počet procesorov je zdrojom, ktorý je ohraničený parametrom $H(n)$ (H ako hardware) rovnako ako priestor je ohraničený parametrom $S(n)$ v sekvenčnej teórii zložitosti. Obyčajne je $H(n)$ fixovaný polynóm v n.

Bol navrhnutý dosť veľký počet modelov paralelných výpočtov (pozri prehľad v [21]) tak, ako je mnoho konkurujúcich sekvenčných modelov (pozri časť 2). Ale hlavní súperi sú dvaja. Prvým je trieda modelov so zdieľanou pamäťou, v ktorej veľa procesorov komunikuje cez pamäť s náhodným prístupom, ktorú majú spoločnú. Bolo publikovaných mnoho paralelných algoritmov pre takéto modely, pretože reálne paralelné stroje, keď sa postavia, môžu byť celkom také. Ale pre matematickú teóriu takéto modely nie sú veľmi uspokojujúce, pretože ich detailnejší popis je príliš nejednoznačný: Ako sa riešia čítacie a zapisovacie konflikty v spoločnej pamäti? Aké základné operácie sú dovolené pre každý procesor? Treba pripočítavať $\log H(n)$ časových jednotiek za prístup do spoločnej pamäte?

Preto dávam prednosť šikovnejšiemu modelu, ktorý rozobral Borodin v [8] (1977); podľa neho je paralelný počítač uniformný systém $\langle B(n)\rangle$ acyklických booleovských schém takých, že $B(n)$ má n vstupov (a teda spracováva vstupné reťazce dĺžky n). Potom $H(n)$ (veľkosť hardwaru) je jednoducho počet logických prvkov v $B(n)$ a $T(n)$ (čas paralelného výpočtu) je hĺbka schémy $B(n)$ (t. j. dĺžka najdlhšej cesty od vstupu k výstupu). Tento model má praktické opodstatnenie v tom, že pravdepodobne všetky reálne stroje (včítane tých so zdieľanou pamäťou) sú zložené z booleovských schém. Okrem toho, minimálna veľkosť a hĺbka booleovskej schémy potrebnej na výpočet funkcie sú prirodzenými matematickými problémami, ktoré sa uvažovali oveľa skôr, než existovala teória paralelných výpočtov.

Našťastie pre teóriu nie sú minimálne hodnoty hardwaru $H(n)$ a paralelného času $T(n)$ priveľmi odlišné pre rôzne konkurujúce modely paralelných počítačov. Špeciálne pre všetky modely je pravdivý zaujímavý univerzálny fakt, ktorý dokázali Pratt a Stockmeyer [58] pre jednotlivý model v r. 1974 a ktorý bol v [33] nazvaný „téza paralelného výpočtu": problém môže byť riešený v čase polynomiálnom v $T(n)$ na paralelnom stroji (s neohraničeným hardwarom) vtedy a len vtedy, ak môže byť riešený v priestore polynomiálnom v $T(n)$ na sekvenčnom stroji (s neohraničeným časom).

Základnou otázkou v paralelných výpočtoch je: ktorý problém možno riešiť podstatne rýchlejšie pri použití mnohých procesorov ako pri použití jedného. Nicholas Pippenger [57a] formalizoval túto otázku definovaním triedy (teraz nazývanou NC ako Nick's class) problémov riešiteľných ultrarýchlo [čas $T(n) = (\log n)^{O(1)}$] na paralelnom počítači s prijateľnou veľkosťou hardwaru [$H(n) = n^{O(1)}$]. Našťastie zostáva trieda NC tá istá, nezávisle od vybraného jednotlivého modelu počítača, a je ľahko vidieť, že NC

je podtrieda triedy FP funkcií vypočítateľných sekvenčne v polynomiálnom čase. Našu neformálnu otázku možno potom formalizovať takto: ktoré problémy z FP sú tiež v NC.

Je možné (hoci nepravdepodobné), že $NC = FP$, pretože dokázať, že $NC \neq FP$ by vyžadovalo náhly postup v teórii zložitosti (pozri koniec časti 4.1.). Keďže nevieme, ako dokázať, že funkcia, ktorá je v FP, nie je v NC, najlepšie je najprv dokázať, že f je úplná vzhľadom na logaritmický priestor pre FP. Takýto dôkaz je analogický dôkazu, že problém je NP-úplný a má praktický dôsledok v tom, že odrádza od úsilia o nájdenie superrýchlych paralelných algoritmov pre f. To preto, že ak f je úplná vzhľadom na logaritmický priestor pre FP a f patrí do NC, potom $FP = NC$, čo by bolo veľkým prekvapením.

Velmi malý pokrok sa urobil v klasifikovaní problémov v FP podľa toho, či sú v NC, alebo sú úplné vzhľadom na logaritmický priestor pre FP (samozrejme nemusia byť ani jedno). Prvý príklad problému úplného pre P som prezentoval v r. 1973 [20], hoci výsledok som neuvádzal ako výsledok úplnosti. Krátko nato Jones a Laaser [38] definovali takéto chápanie úplnosti a uviedli okolo päť príkladov včítane problému prázdnosti pre bezkontextové gramatiky. Pravdepodobne najjednoduchším problémom, o ktorom bolo dokázané, že je úplný pre FP, je takzvaný problém hodnoty schémy [47]: ak je daná booleovská schéma spolu s hodnotami jej vstupov, treba nájsť hodnotu výstupu. Pre mňa najzaujímavejším príkladom od Goldschlagera, Shawa a Staplesa [34] je nájdenie (párnosti) maximálneho toku cez danú sieť s (veľkými) celočíselnými kapacitami na hranách. Záujem pramení z dôvtipnosti dôkazu úplnosti. Nakoniec by som mal spomenúť, že lineárne programovanie je úplné pre FP. Zložitou úlohou v tomto prípade je ukázať, že problém je v P (pozri [43]), po ktorej je dôkaz úplnosti [26] bezprostredný.

Medzi problémami, o ktorých sa vie, že sú v NC, sú štyri aritmetické operácie ($+$, $-$, $*$, $:$) nad binárnymi číslami, triedenie, spojitosť grafu, maticové operácie (násobenie, inverzia, determinant, rang), najväčší spoloční deliteľia polynómov, bezkontextové jazyky a nájdenie minimálneho pokrývajúceho lesa grafu (pozri [11], [21], [63], [67b]). O veľkosti najpočetnejšieho párenia pre daný graf je známe [11], že je v „náhodnom" NC (NC, ktoré dovoľuje hádzanie mincou), hoci zostáva zaujímavým otvoreným problémom, či nájdenie aktuálneho najpočetnejšieho párenia je vôbec v náhodnom NC. Výsledky v [89] a [67b] poskytujú všeobecné metódy na to, ako ukázať, že problémy sú v NC.

Najzaujímavejším problémom v FP, o ktorom sa nevie, ani či je úplný pre FP, alebo je v (náhodnom) NC, je nájdenie najväčšieho spoločného deliteľa pre dve celé čísla. Je mnoho ďalších zaujímavých problémov, ktoré treba zatriediť včítane nájdenia najpočetnejšieho párenia alebo maximálnej kliky v grafe (pozri [88]).

7. Budúcnosť

Dovoľte mi znovu pripomenúť, že oblasť výpočtovej zložitosti je veľká a tento prehľad je stručný. Sú veľké časti tejto teórie, ktoré som vynechal celkom alebo som sa ich sotva dotkol. Moje ospravedlnenie patrí tým, čo bádajú v týchto oblastiach.

Jeden relatívne nový a vzrušujúci smer, ktorý Yao [92] nazýva „výpočtová informačná teória", je založený na Shannonovej klasickej informačnej teórii tým, že uvažuje informáciu dostupnú cez prakticky realizovateľný výpočet. Tento smer široko aktivizovali Diffie a Hellman [25] a Rivest, Shamir a Adleman [67a] svojimi článkami o kryptosystémoch so známym kľúčom, hoci jeho výpočtové korene siahajú ku Kolmogorovovi [45] a Chaitinovi [14a], [14b]. Oni ako prví s použitím teórie výpočtov vysvetlili, čo to znamená, ak sa povie, že jednotlivá konečná postupnosť je „náhodná". Zaujímavú myšlienku v tejto teórii uvažujú Shamir [73] a Blum s Micalim [7]: týka sa generovania pseudonáhodných postupností, v ktorých nasledujúce bity dokázateľne ťažko predpovedať pomocou predošlých bitov. Yao [92] dokazuje, že existencia takýchto postupností by mala pozitívne dôsledky na deterministickú zložitosť pravdepodobnostnej triedy R (pozri časť 5). Skutočne, výpočtová informačná teória sľubuje, že vrhne svetlo na úlohu náhodnosti vo výpočtoch.

Popri výpočtovej informačnej teórii môžeme očakávať zaujímavé nové výsledky o pravdepodobnostných algoritmoch, paralelných výpočtoch a (s trochou šťastia) dolných odhadoch. Čo sa týka dolných odhadov, jeden výrazný postup, pre ktorý vidím nejakú nádej v blízkej budúcnosti, by znamenal dôkaz, že nie každý problém v P je riešiteľný v priestore $O(\log n)$ a asi tiež $P \neq NP$. V každom prípade zostáva oblasť výpočtovej zložitosti veľmi živá a som zvedavý, čo prinesie budúcnosť.

Poďakovanie. *Som vďačný svojim kolegom v teórii zložitosti z Toronta za veľa užitočných poznámok a rád, najmä Alanovi Borodinovi, Joachimovi von zur Gathenovi, Silviovi Micalimu a Charlesovi Rackoffovi.*

Literatúra

[1] Adleman, L.: *Two theorems on random polynomial time.* Proc. 19th IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles (1978), 75—83.

[2] Adleman, L., Pomerance, C. a Rumley, R. S.: *On distinguishing prime numbers from composite numbers.* Annals of Math. *117*, (January 1983), 173—206.

[3] Aho, A. V., Hopcroft, J. E. a Ullman, J. D.: *The Design and Analysis of Computer Algorithms.* Addison-Wesley, Reasing, Mass 1974, ruský preklad *Postrojenije i analiz vyčisliteľnych algoritmov*, Mir, Moskva 1979.

[4] Bennet, J. H.: *On Spectra.* Doktorská dizertácia, Department of Mathematics, Princeton University, 1962.

[5] Berlekamp, E. R.: *Factoring polynomials over large finite fields.* Math. Comp. *24* (1970), 713—735.

[6] Blum, M.: *A machine independent theory of the complexity of recursive functions.* JACM *14*, 2 (April 1967), 322—336.

[7] Blum, M. a Micali, S.: *How to generate cryptographically strong sequences of pseudo random bits.* Proc. of 23rd IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles (1982), 112—117.

[8] Borodin, A.: *On relating time and space to size and depth.* SIAM J. Comp. *6* (1977), 733—744.

[9] Borodin, A.: *Structured vs. general models in computational complexity.* In: *Logic and Algorithmic Monographie no. 30 de L'Enseignement Mathématique.* Université de Genéve, 1982.

[10] Borodin, A. a Cook, S. A.: *A time-space trade-off for sorting on a general sequential model of computation.* SIAM J. Comp. *11* (1982), 287—297.

[11] BORODIN, A., VON ZUR GATHEN, J. a HOPCROFT, J.: *Fast parallel matrix and GCD computations.* Proc. of 23rd Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles (1982), 65—71.

[12] BORODIN, A. a MUNRO, I.: *The Computational Complexity of Algebraic and Numeric Problems.* Elsevier, New York, 1975.

[13] BROCKETT, R. W. a DOBKIN, D.: *On the optimal evaluation of a set of bilinear forms.* Linear Algebra and its Applications *19* (1978), 207—235.

[14a] CHAITIN, G. J.: *On the length of programs for computing finite binary sequences.* JACM *13*, 4 (Otcober 1966), 547—569; JACM *16*, 1 (January 1969), 145—159.

[14b] CHAITIN, G. J.: *A theory of program size formally identical to informational theory.* JACM *22*, 3 (July 1975), 329—340.

[15] COBHAM, A.: *The intrinsic computational difficulty of functions.* Proc. 1964 International Congress for Logic, Methodology and Philosophy of Sciences. Y. Bar-Hellel, Ed. North Holland, Amsterdam, 1965, 24—30.

[16] COBHAM, A.: *The recognition problem for the set of perfect squares.* IEEE Conference Record Seventh SWAT (1966), 78—87.

[17] COHEN, H. a LENSTRA, H. W. ML.: *Primarily testing and Jacobi sums. Report 82-18.* University of Amsterdam, Dept. of Math. 1982.

[18] COOK, S. A.: *The complexity of theorem proving procedures.* Proc. 3rd ACM Symp. on Theory of Computing. Shaker Heights, Ohio (May 3.—5. 1971), 151—158.

[19] COOK, S. A.: *Linear time simulation of deterministic two-way pushdown automata.* Proc. IFIP Congress 71, (Theoretical Foundations). North Holland, Amsterdam, 1972, 75—80.

[20] COOK, S. A.: *An observation on time-storage tradeoff.* JCSS *9* (1974), 308—316. Pôvodne In: Proc. 5th ACM Symp. on Theory of Computing. Austin TX (April 30—May 2, 1973), 29—33.

[21] COOK, S. A.: *Towards a complexity theory of synchronous parallel computation.* L'Enseignement Mathematique *XXVII* (1981), 99—124.

[22] COOK, S. A. a AANDERAA, S.: *On the minimum computation time of functions.* Trans. ASM *142* (1969), 291—314.

[23] COOLEY, J. M. a TUKEY, J. W.: *The algorithm for the machine calculation of complex Fourier series.* Math. Comput. *19*, (1965), 297—301.

[24] COPPERSMITH, D. a WINOGRAD, S.: *On the asymptotic complexity of matrix multiplication.* SIAM J. Comp. *11* (1982), 472—492.

[25] DIFFIE, W. a HELLMAN, M. E.: *New directions in cryptography.* IEEE Trans. on Inform. Theory, IT-22, *6* (1976), 644—654.

[26] DOBKIN, D., LIPTON, R. J. a REISS, S.: *Linear programming is log-space hard for P.* Inf. Processing Letters *8* (1979), 96—97.

[27] EDMONDS, J.: *Paths, trees, flowers.* Canad. J. Math. *17* (1965), 449—467.

[28] EDMONDS, J.: *Minimum partition of a matroid into independent subsets.* J. Res. Nat. Bur. Standards Sect. B, *69* (1965), 62—72.

[29] FERRANTE, J. a RACKOFF, C. W.: *The Computational Complexity of Logical Theories.* Lecture Notes in Mathematics *718*. Springer Verlag, New York, 1979.

[30] FISCHER, M. J. a RABIN, M. O.: *Super-exponential complexity of Presburger arithmetic.* In: Complexity of Computation, SIAM-AMS Proc. *7*, R. Karp, Ed., 1974, 27—42.

[31] GAREY, M. R. a JOHNSON, D. S.: *Computers and Intractability: A Guide to Theory of NP-Completness.* W. H. Freeman, San Francisco, 1979, ruský preklad *Vyčislitel'nyje mašiny i trudnorešaemyje zadači*, Mir, Moskva 1982.

[32] GILL, J.: *Computational complexity of probabilistic Turing machines.* SIAM J. Comput. *6* (1977), 675—695.

[33] GOLDSCHLAGER, L. M.: *Synchronous Parallel Computation.* Doktorská dizertácia. Dept. of Comp. Science University of Toronto, 1977. Pozri tiež JACM *29*, 4 (October 1982), 1073—1086.

[34] GOLDSCHLAGER, L. M., SHAW, R. A. a STAPLES, J.: *The maximum flow problem is log-space complete for P.* Theoretical Computer Science *21* (1982), 105—111.

[35] GRZEGORCZYK, A.: *Some classes of recursive functions.* Rozprawy Matematyczne, 1953.

[36] HARTMANIS, J.: *Observations about the development of theoretical computer science.* Annals Hist. Comput. *3*, 1 (Jan. 1981), 42—51.

[37] HARTMANIS, J. a STEARNS, R. E.: *On the computational complexity of algorithms.* Trans. AMS *117* (1965), 285—306.

[38] JONES, N. D. a LAASER, W. T.: *Complete problems for deterministic polynomial time.* Theoretical Computer Science *3* (1977), 105—117.

[39] KALFOTEN, E.: *A polynomial reduction from multivariate to bivariate integer polynomial factorization.* Proc. 14th ACM Symp. on Theory of Computing, San Francisco, CA (MAY 5—7 1982), 261—266.

[40] KALFOTEN, E.: *A polynomial time reduction from bivariate to univariate integral polynomial factorization.* Proc. 23rd IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles 1982, 57—64.

[41] KARATSUBA, A. a OFMAN, JU.: *Umnoženije mnogoznačnych čisel na avtomatach.* Dokl. Akad. Nauk SSSR, *145* : *2* (1962), 293—294.

[42] KARL, R. M.: *Reducibility among combinatorial problems.* In: *Complexity of Computer Computations*, R. E. MILLER a J. W. THATCHER, Eds. Plenum Press, New York, 1972, 85—104.

[43] CHAČIJAN, L. G.: *Polynomial'nyj algoritm v linejnom programirovaniji.* Dokl. Akad. Nauk SSSR, *244* : *5* (1979), 1093—1096.

[44] KNUTH, D. E.: *The art of Computer Programming, Vol. 3 Sorting and Searching.* Addison-Wesley, Reading, MA 1973, ruský preklad Iskusstvo programmirovanija dlja EVM, t. 3 Sortirovka i poisk, Mir, Moskva 1978.

[45] KOLMOGOROV, A. N.: *Tri podchoda k opredeleniju ponjatija „količestvo informaciji".* Probl. pered. informaciji *1* (1965), 3—11.

[46] KOLMOGOROV, A. N. a USPENSKIJ, V. A.: *K opredeleniju algoritma.* Uspechi matem. nauk *13* (1958), 3—28.

[47] LADNER, R. E.: *The circuit value problem is log-space complete for P.* SIGACT News 7, *1* (1975), 18—20.

[48] LENSTRA, A. K., LENSTRA, H. W. a LOVASZ, L.: *Factoring polynomials with rational coefficients.* Report 82-05, University of Amsterdam, Dept. of Math., 1982.

[49] LEVIN, L. A.: *Universal'nyje zadači perebora.* Probl. pered. informaciji *9* (1973), 115—116.

[50] LUKS, E. M.: *Isomorphism of graphs of bounded valence can be tested in polynomial time.* Proc. of 21st IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles (1980), 42—49.

[51] MEYER, A. R.: *Weak monadic second order theory of successor is not elementary recursive.* Lecture Notes in Mathematics *453*. Springer Verlag, New York, 1975, 132—154.

[52] MEYER, A. R. a STOCKMEYER, L. J.: *The equivalence problem for regular expressions with squaring requires exponential space.* Proc. 13th IEEE Symp. on Switching and Automata Theory (1972), 125—129.

[53] MILLER, G. L.: *Riemann's Hypothesis and tests for primality*, JCSS *13* (1976), 300—317.

[54] OPPEN, D. C.: $2^{2^{2^{pn}}}$ *upper bound on the complexity of Presburger arithmetic.* JCSS *16* (1978), 323—332.

[55] PAPADIMITROU, C. H. a STEIGLITZ, K.: *Combinatorial Optimization: Algorithms and Complexity.* Prentice-Hall, Englewood Cliffs, NJ 1982.

[56] PATERSON, M. S., FISCHER, M. J. a MEYER, A. R.: *An improved overlap argument for on-line multiplication.* SIAM-AMS Proc. *7*, Amer. Math. Society, Providence 1974, 97—111.

[57a] PIPPENGER, N.: *On simultaneous resource bounds* (preliminary version). Proc. 20th IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles (1979), 307—311.

[57b] PIPPENGER, N. J. a FISCHER, M. J.: *Relations among complexity measures.* JACM *26*, 2 (April 1979), 361—381.

[58] PRATT, V. R. a STOCKMEYER, L. J.: *A characterization of the power of vectore machines.* JCSS *12* (1976), 198—221, pôvodne In: Proc. 6th ACM Symp. on Theory of Computing, Seattle, WA (April 30—May 2, 1974), 122—134.

[59] RABIN, M. O.: *Speed of computation and classification of recursive sets.* Third Convention Sci. Soc. Izrael, 1959, 1—2.

[60] RABIN, M. O.: *Degree of difficulty of computing a function and a partial ordering of recursive sets.* Tech. rep. No. *1*, O. N. R. Jeruzalem, 1960.

[61] RABIN, M. O.: *Probabilistic algorithms.* In: *Algorithms and Complexity, New Directions and Recent Trends*, J. F. TRAUB, Ed., Academic Press, New York, 1976, 21—39.

[62] RABIN, M. O.: *Complexity of computations.* Comm. ACM *20*, 9 (September 1977), 625—633.

[63] REIF, J. H.: *Symmetric Complementation.* Proc. 14th ACM Symp. on Theory of Computing, San Francisco, CA (May 5—7, 1982), 201—214.

[64] REISCH, S. a SCHNITGER, G.: *Three applications of Kolmogorov complexity.* Proc. 23rd IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles, 1982, 45—52.

[65] RITCHIE, R. W.: *Classes of Recursive Functions of Predictable Complexity.* Doktorská dizertácia, Princeton University, 1960.

[66] RITCHIE, R. W.: *Classes of predictably computable functions.* Trans. AMS *106* (1963), 139—173.

[67a] RIVEST, R. L., SHAMIR, A. a ADLEMAN, L.: *A method for obtaining digital signatures and public-key cryptosystems.* Comm. ACM 21, *2* (February 1978), 120—126.

[67b] RUZZO, W. L.: *On uniform circuit complexity.* JCSS *22* (1981), 365—383.

[68a] SAVAGE, J. E.: *The Complexity of Computing.* Wiley, New York, 1976.

[68b] SCHNORR, C. P.: *The network complexity and the Turing machine complexity of finite functions.* Acta Informatica 7 (1976), 95—107.

[69] SCHÖNHAGE, A.: *Storage modification machines.* SIAM J. Comput. *9* (1980), 490—508.

[70] SCHÖNHAGE, A. a STRASSEN V.: *Schnelle Multiplication grosser Zahlen.* Computing 7 (1971), 281—292.

[71] SCHWARTZ, J. T.: *Probabilistic algorithms for verification of polynomial identities.* JACM *27*, 4 (October 1980), 701—717.

[72] SCHWARTZ, J. T.: *Ultracomputers.* ACM Trans. on Prog. Languages and Systems *2*, 4 (October 1980), 484—521.

[73] SHAMIR, A.: *On the generation of cryptographically strong pseudo random sequences.* 8th Int. Colloquium on Automata, Languages and Programming (July 1981), Lecture Notes in Computer Science No. *115*, Springer Verlag, New York, 544—550.

[74] SHANNON, C. E.: *The synthesis of two terminal switching circuits.* BSTJ *28* (1949), 59—98, ruský preklad v zb. ŠENNON K. E.: *Raboty po teorii informaciji i kibernetike.* IL Moskva 1963, 59 až 101.

[75] SMALE, S.: *On the average speed of the simplex method of linear programming.* Preprint, 1982.

[76] SMALE, S.: *The problem of the average speed of the simplex method.* Preprint, 1982.

[77] SLOVAY, R. a STRASSEN V.: *A fast Monte-Carlo test for primality.* SIAM J. Comput. *6* (1977), 84—85.

[78] STEARNS, R. E., HARTMANIS, J. a LEWIS, P. M. I.: *Hierarchies of memory limited computations.* 6th IEEE Symp. on Switching Circuit Theory and Logical Design, (1965), 179—190.

[79] STOCKMEYER, L. J.: *The complexity of decision problem in automata theory and logic.* Doktorská dizertácia. Dept. of Electrical Eng., MIT Cambridge, MA 1974, Report TR-133, MIT Laboratory for Computer Sicence.

[80] STOCKMEYER, L. J.: *Classifying the computational complexity of problems.* Research Report RC *7606* (1979), Math. Sciences Dept., IBM T. J. Watson Research Centre, Yorktown Heights, New York.

[81] STRASSEN, V.: *Gaussian elimination is not optimal.* Num. Math. *13* (1969), 354—356.

[82] STRASSEN, V.: *Die Berechnungskomplexität von elementarsymmetrischen Funktionen und von Interpolationskoeffizienten.* Numer. Math. *20* (1973), 238—251.

[83] STRASSEN, V. a BAUR W.: *The complexity of partial derivatives.* Preprint, 1982.
[84] TOOM, A. L.: *O složnosti schemy iz funkcional'nych elementov, realizirujuščej umnoženije celych čisel.* Dokl. Akad. Nauk SSSR *150* : *3* (1963), 496—498.
[85] TURING, A. M.: *On computable numbers with an application to the Entscheidungsproblem.* Proc. London Math. Soc. ser. 2, *42* (1936—7), 230—265. Oprava, tamže *43* (1937), 544—546.
[86] VALIANT, L. G.: *The complexity of enumeration and reliability problems.* SIAM J. Comput. *8* (1979), 410—421.
[87] VALIANT, L. G.: *The complexity of computing the permanent.* Theoretical Computer Science *8* (1979), 189—202.
[88] VALIANT, L. G.: *Parallel Computations.* Proc. 7th IBM Japan Symp. Academic 6 Scientific Programs, IBM Japan, Tokyo (1982).
[89] VALIANT, L. G., SKYUM, S., BERKOWITZ, S. a RACKOFF, C.: *Fast parallel computation on polynomials using few processors.* Preprint (Preliminary version In: Springer Lecture Notes in Computer Science, *118* (1981), 132—139).
[90] VON NEUMANN, J.: *A certain zero-sum two-person game equivalent to the optimal assignment problem.* Contributions to the Theory of Games II, H. W. KAHN a A. W. TUCKER, Eds. Princeton University Press, Princeton NJ, 1953.
[91] YAMADA, H.: *Real time computation and recursive functions not real time computable.* IRE Trans. on Electronic Computers, *EC-11* (1962), 753—760.
[92] YAO, A. C.: *Theory and applications of trapdoor functions.* (Extended abstract). Proc. 23rd IEEE Symp. on Foundations of Computer Science. IEEE Computer Society, Los Angeles (1982), 80—91.
[93] ĎURIŠ, P., GALIL, Z., PAUL, W. a REISCHUK, R.: *Two nonlinear lower bounds.* Proc. of 15th ACM Symp. on Theory of Computing, (1983), 127—132.
[94] MASS, W.: *Quadratic lower bounds for deterministic and non-deterministic one-tape Turing machines.* Proc. of 16th ACM Symp. on Theory of Computing, (1984).
[95] MING, LI.: *On one tape versus two stacks.* Tech. report 84—591, Január 1984, Dept. of Comp. Science, Cornell University, Ithaca, New York.
[96] FREIWALDS, R.: Krátky príspevok na MCFS'84 (nepublikované).
[97] ĎURIŠ, P. a GALIL, Z.: *A time-space tradeoff for language recognition.* Proc. of 22nd IEEE Symp. on Foundations of Computer Science, (1981), 53—57.
[98] HROMKOVIČ, J.: *Optimal time-space-parallelism tradeoffs for language recognition* (nepublikovaný rukopis).
[99] NEŠETŘIL, J.: *Kombinatorické konstrukce, jejich složitost a praktický význam.* PMFA *23* (1978), 1, 16—27.
[100] LOVÁSZ, L.: *A new linear programming Algorithm — better or worse than the simplex method?* The mathematical Intelligencer, Vol. *2*, No. 3, 1980, 141—146, preklad: *Je nový algoritmus lineárního programování lepší nebo horší než simplexová metoda?* PMFA *26* (1981), 4, 193—202.
[101] LAWLER, E. L.: *The great mathematical sputnik of 1979.* The Mathematical Intelligencer, Vol. *2*, No. 4, 1980, 190—198, preklad: *Velký matematický sputnik roku 1979.* PMFA *27* (1982), 1, 38—47.

(Citácie [93]—[101] boli doplnené pri preklade.)

Preložil Milan Ftáčnik

V tom, abychom opravdu bojovali a hledali, odvrhovali dosažené a experimentovali, odbočovali a nacházeli nové, nám brání jediná mocná síla — my sami. Táhne nás to na vyzkoušené obvyklé cesty. Protože neznámé končiny často odrazují. Stává se i to, že lidé poté, co směle pohlédli do neprobádaných prostor, ustupují, ohromeni mohutností spatřeného. Nejvíce strádají první objevitelé, dobyvatelé nového. Nemají na koho se ohlédnout a v sobě ne pokaždé hned naleznou přesvědčivou oporu.