Richard M. Karp
Kombinatorika, zložitosť a náhodnost

Pokroky matematiky, fyziky a astronomie, Vol. 34 (1989), No. 6, 313--335

Persistent URL: http://dml.cz/dmlcz/137849

Kombinatorika, zložitosť a náhodnosť

Držiteľ Turingovej ceny za rok 1985 predstavuje svoj pohľad na vývoj v oblasti, ktorá sa začala nazývať teoretická informatika (computer science).

Richard M. Karp

Táto prednáška je venovaná pamiatke môjho otca, Abrahama Louisa Karpa.

Je to pre mňa česť a zároveň radosť, že som sa stal držiteľom tohoročnej Turingovej ceny. Je príjemné získať takéto ocenenie, ale zistil som, že moje najväčšie uspokojenie ako výskumníka pramení z práce v samotnom výskume a z priateľov, ktorých som počas toho získal. Chcel by som s vami prejsť tých 25 rokov, ktoré som prežil ako výskumník v oblasti kombinatorických algoritmov a výpočtovej zložitosti, a povedať vám o niektorých pojmoch a myšlienkach, ktoré sa mi zdali dôležité, a o niektorých ľuďoch, ktorí ma inšpirovali a ovplyvnili.

Začiatky

Môj vstup do oblasti počítačov bol pomerne náhodný. Keď som skončil v roku 1955 Harvard College s diplomom matematika, stál som pred rozhodnutím, čo robiť ďalej. Len si zarábať na živobytie, to ma nelákalo, takže prirodzenou voľbou bola ašpirantúra. Jedna možnosť bola pokračovať v kariére v matematike, ale v matematike vtedy vrcholil dôraz na abstraktnosť a všeobecnosť a zdalo sa, že konkrétna a aplikovateľná matematika, ktorú som mal najradšej, nie je v móde.

A tak som bol takmer zákonite prijatý na ašpirantúru v Harvardskom výpočtovom laboratóriu. Mnohé z predmetov, ktoré sa mali stať základom učebných plánov matematickej informatiky, sa vtedy ešte nevyučovali, a tak som si vybral eklektickú zmes prednášok: teóriu logických schém, numerickú analýzu, aplikovanú matematiku, pravdepodobnosť a štatistiku, operačný výskum, elektroniku a matematickú lingvistiku. Hoci učebným plánom sa dalo všeličo vytýkať, najmä čo do hĺbky a vzájomných súvislostí, prevládal veľmi špeciálny pocit súdržnosti: vedeli sme, že sa stávame svedkami novej vedeckej disciplíny zameranej na počítače. Zistil som, že nachádzam krásu a eleganciu v štruktúre algoritmov a že mám schopnosti pre diskrétnu matematiku, ktorá tvorila

RICHARD M. KARP: *Combinatorics, Complexity and Randomness.* Communications of the ACM, February 1986, Volume 29, Number 2, pp. 98—109. Reprinted by permissions of the Association for Computing Machinery and Addison-Wesley Publishing Company.

Preložil MILAN FTÁČNIK

základ štúdia počítačov a výpočtov. Tak som sa dostal viac menej náhodou do oblasti, ktorá mi bola veľmi po chuti.

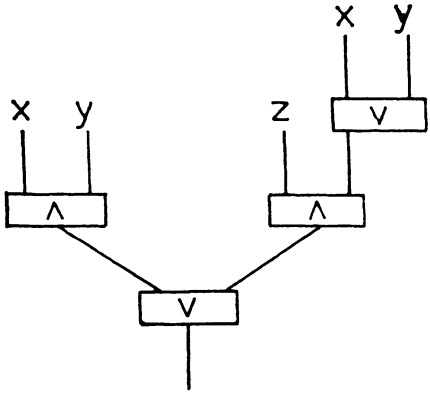
Ľahké a ťažké kombinatorické problémy

Od samého začiatku som mal veľký záujem o kombinatorické problémy hľadania – problémy, ktoré sa podobajú skladačke, hre. V tejto hre musí človek poskladať časti určitej štruktúry určeným spôsobom. Takéto problémy obsahujú prehľadávanie konečnej, ale ohromne veľkej štrukturovanej množiny možných riešení, modelov alebo systémov, aby sa našiel jeden, ktorý vyhovuje zadanej množine podmienok. Niekoľkými príkladmi takýchto problémov sú rozmiestnenie a poprepájanie súčastí integrovaného obvodu na čípe, rozpis Národnej futbalovej ligy či stanovenie cestovného poriadku školských autobusov.

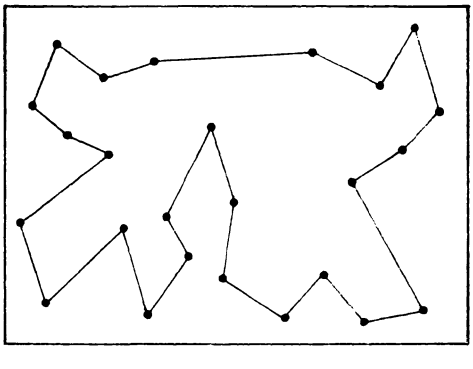
V každom z týchto kombinatorických problémov sa skrýva možnosť kombinatorickej explózie. Pretože počet možností, ktoré treba prehľadať, ohromne a neohraničene rastie, možno pritom naraziť na veľké množstvo výpočtov, ak len sa na prehľadávanie priestoru možných riešení nepoužije nejaká lesť. Rád by som začal technickú časť tohto rozprávania tým, že poviem o mojich prvých zápasoch s kombinatorickou explóziou.

Moja prvá porážka v tomto zmysle prišla krátko po tom, ako som začal pracovať vo výskumnom centre IBM Yorktown Heights v roku 1959. Pridelili ma do skupiny, ktorú viedol J. P. Roth, vážený odborník v algebraickej topológii, ktorý významne prispel ku teórii logických schém. Poslaním našej skupiny bolo vytvoriť program pre počítač na automatickú syntézu logických schém. Vstupom programu bola množina booleovských formúl udávajúca, ako závisia výstupy schémy na jej vstupoch: predpokladalo sa, že program bude generovať logickú schému s použitím minimálneho množstva logických hradiel. Obr. 1. ukazuje schému pre väčšinovú funkciu troch premenných: výstup je jednotkou vždy, keď aspoň dve z troch premenných x, y a z sú jednotky.

Program, ktorý sme vytvorili, obsahoval veľa elegantných skratiek a zjemnení, ale



Obr. 1. Schéma pre väčšinovú funkciu



Obr. 2. Cesta obchodného cestujúceho

jeho základný mechanizmus jednoducho vyčísľoval možné schémy v poradí ich vzrastajúcej ceny. Počet schém, ktoré mal program prehľadať, so zväčšovaním množstva vstupných premenných ohromne rástol; z toho vyplývalo, že sa nikdy nemôžeme dostať za riešenie jednoduchých problémov. Dnes sa náš optimizmus, to, že sme vôbec skúšali enumeračný prístup, môže zdať úplne naivný, ale neboli sme jediní, čo sa chytili do tejto pasce: veľa práce v automatickom dokazovaní viet za posledné dve desaťročia začalo počiatočnou vlnou nadšenia, keď sa podarilo úspešne riešiť jednoduché problémy, po čom nasledovala dezilúzia, keď už vážnosť kombinatorickej explózie bola úplne očividná.

Približne v rovnakom čase som začal s Michaelom Heldom z IBM pracovať na probléme obchodného cestujúceho. Tento problém dostal svoje meno podľa situácie, v ktorej sa obchodný cestujúci chystá navštíviť všetky mestá vo svojej oblasti, začať a skončiť chce vo svojom bydlisku a chce minimalizovať svoje cestovné výdavky. V špeciálnom prípade, keď sú tieto mestá bodmi v rovine a cestovné výdavky sa rovnajú euklidovskej vzdialenosti, problém jednoducho znamená nájsť polygón s minimálnym obvodom prechádzajúci všetkými mestami (pozri obr. 2). Niekoľko rokov predtým sa Georgeovi Dantzigovi, Raymondovi Fulkersonovi a Selmerovi Johnsonovi z Rand Corporation podarilo použitím ručných a automatických výpočtov vyriešiť problém pre 49 miest a my sme dúfali, že prekonáme ich rekord.

Napriek svojmu nevinnému vzhľadu obsahoval problém obchodného cestujúceho potenciálne kombinatorickú explóziu, pretože počet možných ciest cez n miest v rovine je $(n-1)!/2$, čo je veľmi rýchlo rastúca funkcia v n. Napríklad ak počet miest je iba 20, čas potrebný na vymenovanie všetkých ciest hrubou silou, pri pomere milión ciest za sekundu, by bol viac ako tisíc rokov.

Held a ja sme skúšali množstvo prístupov k problému obchodného cestujúceho. Začali sme znovuobjavením zrýchlenia založeného na dynamickom programovaní, na ktoré pôvodne upozornil Richard Bellman. Metóda dynamického programovania redukovala čas prehľadávania na $n^2 2^n$, ale táto funkcia tiež explozívne rastie a metóda je prakticky ohraničená na problémy s najviac 16 mestami. Po čase sme sa vzdali myšlienky riešiť problém presne a experimentovali sme s metódami lokálneho prehľadávania, ktoré viedli k dobrým, aj keď nie optimálnym cestám. V týchto metódach sa začne s nejakou cestou a opakovane sa hľadajú lokálne zmeny, ktoré ju vylepšia. Proces pokračuje, kým sa nenájde cesta, ktorá sa žiadnou lokálnou zmenou nedá vylepšiť. Naše metódy lokálneho vylepšovania boli dosť ťažkopádne a neskôr našli Shen Lin a Brian Kernighan z Bell Labs omnoho lepšie. Takéto automatické metódy sú často v praxi veľmi užitočné, ak sa striktne nepožaduje optimálne riešenie, ale nikto nemôže zaručiť, ako dobre budú fungovať.

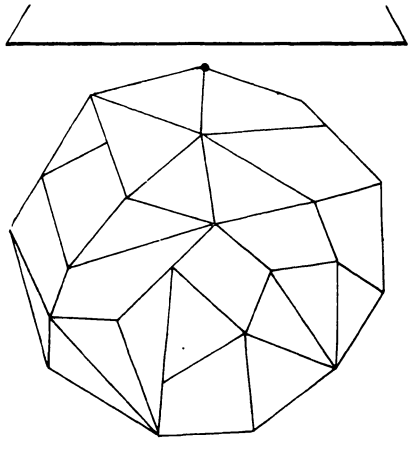
Potom sme začali skúmať metódy ohraničeného vetvenia. Takéto metódy sú v zásade enumeratívnej povahy, ale efektívnosť získavajú odseknutím veľkých častí priestoru možných riešení. Robí sa to vypočítaním dolného odhadu ceny každej cesty, ktorá obsahuje určité spojenia a nemôže obsahovať určité iné; ak je dolný odhad dostatočne veľký, plynie z toho, že žiadna taká cesta nemôže byť optimálna. Po dlhej sérii neúspešných experimentov sme Held a ja náhodou narazili na silnú metódu získavania dolných odhadov. Táto ohraničujúca technika nám umožnila podstatne obmedziť prehľadávanie, takže sme boli schopní riešiť problémy až so 65 mestami. Myslím, že žiadny z mojich

teoretických výsledkov neposkytol také veľké napätie pri pohľade na čísla vychádzajúce z počítača ako v tú noc, keď sme Held a ja testovali našu ohraničujúcu techniku. Neskôr sme zistili, že naša metóda bola variantom starej techniky nazývanej Lagrangeova relaxácia, ktorá sa teraz rutinne používa na získavanie dolných odhadov pri technikách ohraničujúceho vetvenia.

Na krátky čas bol náš program svetovým preborníkom v riešení problému obchodného cestujúceho, ale dnes už existujú pôsobivejšie metódy. Sú založené na technike nazývanej polyedrálna kombinatorika, ktorá sa snaží previesť prípady problému obchodného cestujúceho na veľké problémy lineárneho programovania. Takéto metódy môžu riešiť problémy s viac ako 300 mestami, ale tento prístup úplne neeliminuje kombinatorickú explóziu, pretože čas potrebný na riešenie problému stále rastie exponenciálne ako funkcia počtu miest.

Problém obchodného cestujúceho zostáva fascinujúcou záhadou. Bola publikovaná viac ako 400stranová kniha pokrývajúca väčšinu toho, čo sa vie o tomto nepostihnuteľnom probléme. Neskôr budeme hovoriť o teórii **NP**-úplnosti, ktorá poskytuje dôkaz, že podstatou problému obchodného cestujúceho je praktická neriešiteľnosť; žiadne úsilie o návrh dômyselných algoritmov nemôže preto úplne obísť kombinatorickú explóziu, ktorá sa skrýva v tomto probléme.

Začiatkom 60. rokov malo IBM Research Laboratory v Yorktown Heights vynikajúcu skupinu kombinatorických matematikov a pod ich vedením som sa naučil dôležité techniky na riešenie určitých kombinatorických problémov s vyhnutím sa kombinatorickej explózii. Napríklad som sa zoznámil s Dantzigovým slávnym simplexovým algoritmom lineárneho programovania. Problémom lineárneho programovania je nájsť bod na mnohostene v mnohorozmernom priestore, ktorý je najbližšie k danej externej nadrovine (mnohosten je zovšeobecnením polygónu z dvojrozmerného priestoru alebo obvyklé polyedrálne teleso v trojrozmernom priestore a nadrovina je zovšeobecnením priamky v rovine alebo roviny v trojrozmernom priestore). Najbližší bod v nadrovine je vždy rohovým bodom alebo vrcholom mnohostenu (pozri obr. 3). V praxi sa možno spoľahnúť, že simplexová metóda nájde potrebný vrchol veľmi rýchlo.

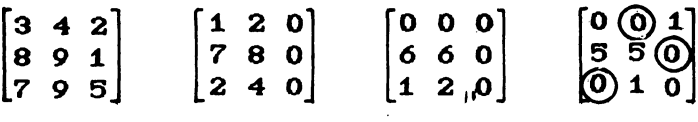


Obr. 3. Problém lineárneho programovania

Naučil som sa tiež krásnu teóriu o tokoch v sieťach od Lestera Forda a Fulkersona. Táto teória sa zaoberá množstvom, ktorým sa také látky ako olej, plyn, elektrina alebo bity informácie môžu presúvať cez sieť, v ktorej každé spojenie má určitú kapacitu, obmedzujúcu množstvo látky cez ňu prepravovanej. Viaceré kombinatorické problémy, ktoré na prvý pohľad nemajú žiadny vzťah ku látkam prepravovaným cez siete, možno zmeniť na problémy tokov v sieťach; príslušná teória umožňuje riešiť takéto problémy elegantne a efektívne len s využitím aritmetických operácií sčítania a odčítania.

Dovoľte mi ilustrovať túto krásnu teóriu náčrtom takzvaného maďarského algoritmu na riešenie jedného problému kombinatorickej optimalizácie známeho ako svadobný problém. Problém obsahuje spoločnosť pozostávajúcu z n mužov a n žien. Problémom je, ako vytvoriť páry mužov a žien s minimálnou cenou, ak každej dvojici je priradená určitá cena. Tieto ceny sú zadané maticou $n \times n$, v ktorej každý riadok zodpovedá jednému mužovi a každý stĺpec jednej žene. Každé popárenie n mužov a n žien vo všeobecnosti zodpovedá výberu n prvkov matice, pričom žiadne dva z nich neležia v rovnakom riadku alebo stĺpci: cenou popárenia je suma n vybraných prvkov. Počet možných popárení je $n!$, funkcia, ktorá rastie tak rýchlo, že vyčísľovanie hrubou silou je nám málo platné. Obrázok 4a znázorňuje príklad rozmeru 3×3, v ktorom vidíme, že cena vytvorenia páru z tretieho muža a druhej ženy sa rovná 9; je to prvok v treťom riadku a v druhom stĺpci v danej matici.

Kľúčovým momentom, ktorý je základom maďarského algoritmu, je všimnúť si, že problém sa nezmení, ak sa odpočíta rovnaká konštanta od všetkých prvkov v určitom riadku matice. S využitím tejto možnosti meniť matice snaží sa algoritmus vytvoriť takú maticu, v ktorej sú všetky prvky nezáporné, takže každé úplné popárenie má nezápornú celkovú cenu a existuje v nej úplné popárenie, v ktorom sú všetky prvky nulové. Takéto popárenie je zjavne optimálne pre cenovú maticu, ktorú sme vytvorili, a práve tak je optimálne aj pre pôvodnú cenovú maticu. V našom príklade 3×3 algoritmus začína tým, že v každom riadku odpočíta najmenší prvok v tomto riadku od všetkých prvkov riadku. Tak sa vytvorí matica, v ktorej každý riadok obsahuje aspoň jednu nulu (obr. 4b). Na vytvorenie nuly v každom stĺpci odpočíta potom algoritmus vo všetkých stĺpcoch, ktoré ešte neobsahujú nulu, najmenší prvok v tomto stĺpci (obr. 4c). V tomto príklade ležia všetky nuly výslednej matice v prvom riadku alebo treťom stĺpci; keďže úplné popárenie obsahuje iba jeden prvok z každého riadku alebo stĺpca, ešte stále nie je možné nájsť úplné popárenie obsahujúce iba nulové prvky. Na vytvorenie takéhoto popárenia je nevyhnutné vytvoriť nulu v ľavej dolnej časti matice. V tomto prípade vytvorí algoritmus nulu odpočítaním jednotky od prvého a druhého stĺpca a pridaním jednotky k prvému riadku (obr. 4d). Vo výslednej nezápornej matici tvoria zakrúžkované prvky úplné popárenie s nulovou cenou a toto popárenie je preto optimálne, a to tak v tejto ako aj v pôvodnej matici.



Obr. 4. Prípad svadobného problému

Tento algoritmus je omnoho obratnejší a efektívnejší ako vyčísľovanie hrubou silou. Čas potrebný na riešenie svadobného problému rastie s treťou mocninou n, t. j. počtu riadkov a stĺpcov matice. Dôsledkom je, že je možné riešiť príklady s tisíckami riadkov a stĺpcov.

Generácia vedcov, ktorí rozpracovali teóriu lineárneho programovania a tokov v sieťach mali pragmatický vzťah ku základom výpočtovej zložitosti: algoritmus sa považoval za efektívny, ak v praxi bežal dostatočne rýchlo a nebolo osobitne dôležité dokazovať, že je rýchly vo všetkých možných prípadoch. V roku 1967 som si všimol, že štandardný algoritmus na riešenie určitých problémov tokov na sieťach má teoretickú dieru, ktorá spôsobila, že beží veľmi pomaly na určitých vhodne vymyslených príkladoch. Zistil som, že nie je ťažké opraviť túto dieru, a hovoril som o tomto výsledku na kombinatorickom seminári v Princetone. Ľudia z Princetonu ma informovali, že Jack Edmonds, odborník z National Bureau of Standards, prezentoval veľmi podobné výsledky na rovnakom seminári počas minulého týždňa.

Ako výsledok tejto zhody sme Edmonds a ja začali pracovať spoločne na teoretickej efektívnosti algoritmov tokov na sieťach a po čase sme publikovali spoločný článok. Ale hlavným výsledkom našej spolupráce bolo posilnenie niektorých ideí o výpočtovej zložitosti, ku ktorým som sa tápavo prepracovával a ktoré mali výrazný vplyv na budúce zameranie môjho výskumu. Edmonds bol majster svojho odboru, ktorý využil idey spojené s lineárnym programovaním na vytvorenie ohromujúcich algoritmov pre množstvo kombinatorických problémov. A okrem bohatej skúsenosti v konštruovaní algoritmov bol o niečo vpredu pred svojimi súčasníkmi v ďalšom významnom aspekte: vytvoril jasné a precízne chápanie toho, čo to znamená, že algoritmus je efektívny. V jeho článkoch je vyložený názor, že algoritmus možno považovať za „dobrý", ak čas jeho behu je ohraničený polynomiálnou funkciou vo veľkosti vstupu a nie napríklad exponenciálnou funkciou. Podľa Edmondsovho prístupu napríklad maďarský algoritmus pre svadobný problém je dobrým algoritmom, pretože čas jeho behu rastie s treťou mocninou veľkosti vstupu. Ale pokiaľ vieme, neexistuje možno žiadny dobrý algoritmus pre problém obchodného cestujúceho, pretože všetky algoritmy, čo sa vyskúšali, vykazovali exponenciálny čas behu vzhľadom na veľkosť problému. Edmondsova definícia nám dala jasnú predstavu, ako definovať hranicu medzi ľahkými a ťažkými kombinatorickými problémami a po prvý raz utvorila, aspoň v mojom ponímaní, možnosť, že raz snáď prídeme s teorémou, ktorá dokáže alebo vyvráti domnienku, že samou podstatou problému obchodného cestujúceho je praktická neriešiteľnosť.

Cesta k NP-úplnosti

Popri pokroku v oblasti kombinatorických algoritmov zbieral v šesťdesiatych rokov sily iný smer výskumu — teória výpočtovej zložitosti. Základy tejto teórie položila v tridsiatych rokoch skupina logikov (včítane Alana Turinga), ktorí sa zaoberali existenciou alebo neexistenciou automatických procedúr pre rozhodovanie, či matematické tvrdenia sú pravdivé, alebo nie.

Turing a ostatní priekopníci teórie výpočtov boli prví, ktorí dokázali, že určité dobre

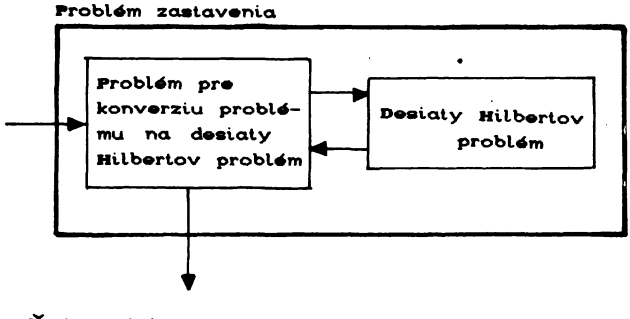
definované matematické problémy sú nerozhodnuteľné, to znamená, že principiálne nemôže existovať algoritmus schopný riešiť všetky prípady takýchto problémov. Prvým príkladom takéhoto problému bol problém zastavenia, ktorý je v podstate otázkou o ladení počítačových programov. Vstupom problému zastavenia je počítačový program spolu so vstupnými údajmi; problémom je rozhodnúť, či sa program nakoniec zastaví. Ako je možné, že neexistuje algoritmus na taký dobre definovaný problém? Ťažkosti vyplývajú z možnosti neohraničeného hľadania. Obvyklým riešením je proste spustiť program, kým sa nezastaví. Ale kedy už je logické vzdať sa, rozhodnúť sa, že program sa nezastaví? Zdá sa, že neexistuje spôsob, ako stanoviť limit na množstvo potrebného hľadania. S použitím techniky zvanej diagonalizácia skonštruoval Turing dôkaz, že neexistuje algoritmus, ktorý môže úspešne riešiť všetky prípady problému zastavenia.

S postupom rokov sa našli nerozhodnuteľné problémy takmer v každom odvetví matematiky. Príkladom z teórie čísiel je problém riešenia diofantických rovníc. Ak je daná polynomická rovnica, ako napr.

$$4xy^2 + 2xy^2z^3 - 11x^3y^2z^2 = -1164\,,$$

existuje jej celočíselné riešenie? Problém nájsť všeobecnú rozhodovaciu procedúru na riešenie takých diofantických rovníc prvýkrát predložil David Hilbert v roku 1900 a tento problém sa stal známy ako desiaty Hilbertov problém; zostal otvorený až do roku 1971, kedy sa dokázalo, že žiadna taká rozhodovacia procedúra nemôže existovať.

Jedným zo základných prostriedkov používaných pri vytyčovaní hranice medzi riešiteľnými a neriešiteľnými problémami je pojem redukovateľnosti, ktorý vystúpil do popredia prácami logika Emila Posta. Hovoríme, že problém A je redukovateľný na problém B, ak za predpokladu, že je známy podprogram na riešenie problému B, môžeme skonštruovať algoritmus na riešenie problému A. Ako príklad: medzníkovým výsledkom je, že problém zastavenia je redukovateľný na desiaty Hilbertov problém (pozri obr. 5). Z toho vyplýva, že desiaty Hilbertov problém je nerozhodnuteľný, pretože inak by sme boli schopní použiť túto redukciu na odvodenie algoritmu pre problém zastavenia, o ktorom sa vie, že je nerozhodnuteľný. Pojem redukovateľnosti sa nám zíde znova, ak budeme rozoberať **NP**-úplnosť a **P-NP** problém.



Obr. 5. Problém zastavenia je redukovateľný na desiaty Hilbertov problém

Ďalšou dôležitou témou, ktorú teória zložitosti zdedila z teórie výpočtov, je rozdiel medzi schopnosťou riešiť problém a schopnosťou vyskúšať nejaké riešenie. Aj keď neexistuje žiadny všeobecný postup, ako nájsť riešenie diofantickej rovnice, je ľahké

vyskúšať navrhované riešenie. Napríklad preveriť, či $x = 3$, $y = 2$, $z = -1$ tvoria riešenie diofantickej rovnice uvedenej vyššie sa dá jednoducho dosadením a trochou aritmetiky. Ako uvidíme, rozdiel medzi riešením a preverovaním je vlastne podstatou **P-NP** problému.

Niektoré z najstarších odvetví teoretickej informatiky majú svôj pôvod v abstraktných zariadeniach a ďalších formalizmoch teórie výpočtov. Jedným z najdôležitejších týchto odvetví je teória výpočtovej zložitosti. Namiesto jednoduchej otázky, či problém je rozhodnuteľný ako taký sa teória zložitosti pýta, nakoľko je ťažké problém riešiť. Inými slovami: teória zložitosti sa zaoberá schopnosťami univerzálnych výpočtových zariadení, ako sú napr. Turingove stroje, ak sú dané obmedzenia na čas ich výpočtu alebo množstvo pamäte, ktorú môžu použiť. Prvé náznaky teórie zložitosti možno nájsť v článkoch z rokov 1959 a 1960 publikovaných Michaelom Rabinom, Robertom McNaughtonom a Hideo Yamadom, ale až článok Jurija Hartmanisa a Richarda Stearnsa znamenal začiatok modernej éry teórie zložitosti. S použitím Turingovho stroja ako modelu abstraktného počítača poskytli Hartmanis a Stearns presnú definíciu „triedy zložitosti" pozostávajúcej z problémov riešiteľných v počte krokov, ktorý je ohraničený danou funkciou vo veľkosti vstupu n. Adaptáciou techniky diagonalizácie, ktorú Turing použil na dôkaz nerozhodnuteľnosti problému zastavenia, dokázali množstvo zaujímavých výsledkov o štruktúre tried zložitosti. Všetci, ktorí sme čítali ich článok, sme začali chápať, že teraz máme dostatočný formálny rámec na hľadanie odpovedí na otázky, ktoré už skôr predložil Edmonds v intuitívnej podobe — na otázky, či napríklad je problém obchodného cestujúceho riešiteľný v polynomiálnom čase.

V tom istom roku som sa učil teóriu výpočtov z vynikajúcej knihy Hartleyho Rogersa, ktorý bol mojím učiteľom na Harvarde. Pamätám sa na svoje vtedajšie úvahy, či pojem redukovateľnosti, taký dominujúci v teórii výpočtov, môže hrať úlohu v teórii zložitosti, ale nevidel som, ako skonštruovať prepojenie. Približne v rovnakom čase bol Michael Rabin, ktorý mal dostať Turingovu cenu v roku 1976, na návšteve v IBM Research Laboratory v Yorktown Heights; bolo to počas dovolenky z Hebrejskej univerzity v Jerusaleme. Stalo sa, že sme bývali v rovnakej budove na okraji New York City a zvykli sme si spoločne tráviť dlhú cestu do Yorktown Heights. Rabin je skutočne originálny mysliteľ, jeden zo zakladateľov aj teórie automatov, aj teórie zložitosti, a počas našich denných diskusií pozdĺž Sawmill River Parkway som získal omnoho širší pohľad na logiku, teóriu výpočtov a teóriu abstraktných výpočtových zariadení.

V roku 1968, zrejme pod vplyvom celkových sociálnych nepokojov, ktoré zachvátili národ, som sa rozhodol prejsť na University of California v Berkeley, kde sa čosi dialo. Roky v IBM boli rozhodujúce pre môj vývoj ako vedca. Možnosť pracovať s takými vynikajúcimi vedcami ako Alan Hoffman, Raymond Miller, Arnold Rosenberg a Shmuel Winograd bola jednoducho neoceniteľná. Nový okruh mojich kolegov zahŕňal Michaela Harrisona, uznávaného odborníka na teóriu jazykov, ktorý ma zverboval pre Berkeley, Eugena Lawlera, experta na kombinatorickú optimalizáciu, Manuela Bluma, zakladateľa teórie zložitosti, ktorý prešiel na nevyriešené problémy na rozhraní teórie čísiel a kryptografie, a Stephena Cooka, ktorého práca v teórii zložitosti ma mala ovplyvniť o niekoľko rokov neskôr. Na matematickej katedre boli: Julia Robinson, ktorej práce na desiatom Hilbertovom probléme mali čoskoro priniesť ovocie, Robert Solovay,

známy logik, ktorý neskôr objavil dôležitý náhodný algoritmus na testovanie, či číslo je prvočíslom, a Steve Smale, ktorého priekopnícka práca o pravdepodobnostnej analýze lineárneho programovania ma mala ovplyvniť o niekoľko rokov neskôr. A za zálivom v Stanforde bol Dantzig, otec lineárneho programovania, Donald Knuth, zakladateľ oblasti dátových štruktúr a analýzy algoritmov, rovnako ako Robert Tarjan, vtedy ašpirant, a John Hopcroft, na vedeckej dovolenke z Cornellu, ktorý brilantne aplikoval techniky dátových štruktúr na analýzu grafových algoritmov.

V roku 1971 publikoval Cook, ktorý medzitým prešiel na Univerzitu v Toronte, svoj historický článok *On the complexity of theorem-proving procedures* (O zložitosti procedúr na dokazovanie teorém). Cook rozoberal triedy problémov, ktoré teraz voláme **P** a **NP**, a zaviedol pojem, ktorý teraz označujeme **NP**-úplnosť. Neformálne povedané, trieda **P** pozostáva zo všetkých tých problémov, ktoré možno riešiť v polynomiálnom čase. Svadobný problém patrí teda do **P**, pretože maďarský algoritmus rieši jeho zadanie o veľkosti n za približne n^3 krokov, ale zdá sa, že problém obchodného cestujúceho neleží v **P**, pretože každá známa metóda riešenia vyžaduje exponenciálny čas. Ak akceptujeme tvrdenie, že výpočtový problém nie je prakticky riešiteľný, kým nemáme polynomiálny algoritmus na jeho riešenie, potom všetky riešiteľné problémy ležia v **P**. Trieda **NP** obsahuje všetky tie problémy, pre ktoré možno navrhované riešenie preveriť v polynomiálnom čase. Uvažujme napríklad verziu problému obchodného cestujúceho, v ktorej sú ako vstupné údaje zadané vzdialenosti medzi všetkými dvojicami miest a ešte „cieľové číslo" T, a úlohou je určiť, či existuje cesta dĺžky menšej, alebo rovnej ako T. Zdá sa nesmierne ťažké určiť, či taká cesta existuje, ale ak niekto navrhne nejakú cestu, ľahko overíme, či jej dĺžka je menšia, alebo rovná ako T; táto verzia problému obchodného cestujúceho leží teda v triede **NP**. Podobne pomocou zavedenia cieľového čísla T možno ukázať, že všetky kombinatorické problémy obvykle uvažované v obchode, vede alebo inžinierstve majú verzie, ktoré ležia v triede **NP**.

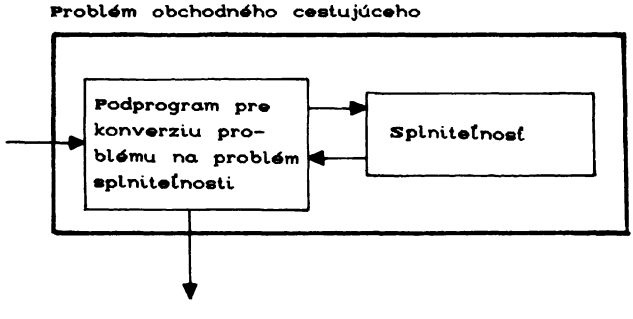
Preto **NP** je trieda, do ktorej väčšinou padnú kombinatorické problémy; vnútri **NP** leží **P**, trieda problémov, ktoré majú efektívne riešenie. Fundamentálnou otázkou je: aký je vzťah medzi triedami **P** a **NP**? Je jasné, že **P** je podmnožinou **NP**, a otázka, na ktorú upriamil pozornosť Cook, je, či **P** a **NP** môžu byť tou istou triedou. Ak by sa **P** rovnalo **NP**, malo by to úžasné dôsledky: znamenalo by to, že každý problém, pre ktorý je ľahké preveriť riešenie, by bol aj ľahko riešiteľný; znamenalo by to, že kedykoľvek existuje krátky dôkaz nejakej teorémy, rýchlo by ho aj našla uniformná procedúra; to by znamenalo, že všetky obvyklé kombinatorické optimalizačné problémy by boli riešiteľné v polynomiálnom čase. Stručne povedané by to znamenalo, že hrozbu kombinatorickej explózie možno odvrátiť. Ale napriek týmto heuristickým dôkazom, že by bolo príliš dobre, ak by sa **P** a **NP** rovnali, sa zatiaľ nenašiel žiadny dôkaz, že $\mathbf{P} \neq \mathbf{NP}$, a niektorí odborníci dokonca veria, že sa žiadny ani nenájde.

Najdôležitejším výsledkom Cookovho článku bolo, že ukázal, že $\mathbf{P} = \mathbf{NP}$ vtedy a len vtedy, ak určitý výpočtový problém zvaný problém splniteľnosti leží v **P**. Problém splniteľnosti pochádza z matematickej logiky a má aplikácie v teórii obvodov, ale možno ho formulovať ako jednoduchú kombinatorickú hru: sú dané viaceré postupnosti veľkých a malých písmen: je možné vybrať z každej postupnosti písmeno bez toho, aby sme vybrali aj veľkú aj malú verziu akéhokoľvek písmena? Ak postupnosti sú *Abc*, *Bc*, *aB* a *ac*,

je možné napríklad vybrať A z prvej postupnosti, B z druhej a tretej a c zo štvrtej; všimnite si, že to isté písmeno možno vybrať viackrát, za predpokladu, že nevyberieme aj jeho malú aj veľkú verziu. Príklad, kde neexistuje spôsob požadovaného výberu, je daný štyrmi postupnosťami: AB, Ab, aB a ab.

Problém splniteľnosti je zjavne v **NP**, pretože je jednoduché preveriť, či navrhovaný výber písmen spĺňa podmienky problému: Cook dokázal, že ak je problém splniteľnosti riešiteľný v polynomiálnom čase, potom každý problém z **NP** je riešiteľný v polynomiálnom čase, z čoho **P** = **NP**. Je teda zrejmé, že tento zdanlivo bizarný a bezvýznamný problém je typickým kombinatorickým problémom, pretože obsahuje kľúč k efektívnemu riešeniu všetkých problémov v **NP**.

Cookov dôkaz bol založený na pojme redukovateľnosti, s ktorým sme sa stretli pri rozoberaní teórie výpočtov. Ukázal, že každý konkrétny prípad problému v **NP** možno transformovať na zodpovedajúci prípad problému splniteľnosti, a to tak, že pôvodný problém má riešenie práve vtedy, ak ho má problém splniteľnosti. Túto transformáciu naviac možno uskutočniť v polynomiálnom čase. Inými slovami, problém splniteľnosti je dostatočne všeobecný na to, aby zachytil štruktúru akéhokoľvek problému v **NP**. Z toho vyplýva, že ak by sme vedeli riešiť problém splniteľnosti v polynomiálnom čase, boli by sme schopní skonštruovať polynomiálny algoritmus na riešenie akéhokoľvek problému v **NP**. Tento algoritmus by pozostával z dvoch častí: polynomiálnej transformačnej procedúry, ktorá prevedie prípad daného problému na prípad problému splniteľnosti a polynomiálneho podprogramu na riešenie vlastného problému splniteľnosti (pozri obr. 6).



Obr. 6. Problém obchodného cestujúceho je redukovateľný v polynomiálnom čase na problém splniteľnosti

Po prečítaní Cookovho článku som si okamžite uvedomil, že jeho pojem typického kombinatorického problému bol formalizáciou myšlienky, ktorá bola dlho súčasťou práce v kombinatorickej optimalizácii. Odborníci v tejto oblasti vedia, že problém celočíselného programovania, čo je v podstate problém rozhodnúť, či systém lineárnych nerovností má celočíselné riešenie, je dostatočne všeobecný na to, aby vyjadril obmedzenia akéhokoľvek z bežne uvažovaných problémov kombinatorickej optimalizácie. Dantzig v roku 1960 publikoval článok na túto tému. Pretože Cook sa zaujímal o dokazovanie viet viac ako o kombinatorickú optimalizáciu, vybral si iný typický problém, hoci základná myšlienka bola rovnaká. Avšak bol tu kľúčový rozdiel: použitím aparátu teórie zložitosti vytvoril Cook rámec, v ktorom sa typická podstata daného problému mohla stať teorémou a nielen neformálnou hypotézou. Je zaujímavé, že Leonid Levin,

ktorý bol vtedy v Leningrade a teraz je profesorom na Bostonskej univerzite, nezávisle objavil v podstate tie isté myšleneky. Jeho typický problém bol založený na vypĺňaní konečných útvarov v rovine kockami domina.

Rozhodol som sa preskúmať, či určitá trieda kombinatorických problémov, o ktorej sa dlho verilo, že je prakticky neriešiteľná, je tiež typická v Cookovom zmysle. Nazval som také problémy „polynomiálne úplné", ale tento termín bol prekonaný presnejším termínom „NP-úplný". Problém je NP-úplný, ak patrí do triedy NP a každý problém v NP je naň polynomiálne redukovateľný. Na dôkaz toho, že daný problém v NP je NP-úplný, stačí ukázať, že nejaký problém, o ktorom sa už vie, že je NP-úplný, je polynomiálne redukovateľný na daný problém. Skonštruovaním série polynomiálnych redukcií som ukázal, že väčšina klasických problémov ukladania, pokrývania, porovnávania, rozkladania, prepájania a plánovania, ktoré vznikajú v kombinatorickej optimalizácii, sú NP-úplné. Prezentoval som tieto výsledky v roku 1972 v článku nazvanom *Reducibility among combinatorial problems* (Redukovateľnosť medzi kombinatorickými problémami). Moje prvé výsledky boli rýchlo zosilnené a rozšírené ďalšími odborníkmi a počas niekoľkých rokov bolo o stovkách rozličných problémov, objavujúcich sa vlastne v každej oblasti, kde sa niečo počíta, ukázané, že sú NP-úplné.

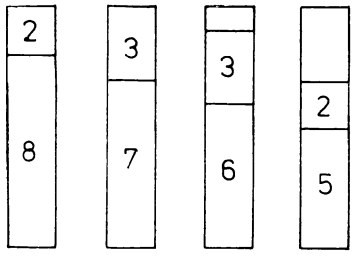
Zvládnutie NP-úplnych problémov

Za svoj výskum NP-úplných problémov som bol odmenený administratívnym miestom. Od roku 1973 do roku 1975 som viedol novovytvorené oddelenie computer science v Berkeley a pri týchto povinnostiach mi nezvyšovalo veľa času na výskum. Výsledkom bolo, že som stál trochu v ústraní počas veľmi aktívneho obdobia, kedy sa našlo veľa príkladov NP-úplných problémov a urobili sa prvé pokusy odstrániť negatívne dôsledky NP-úplnosti.

Výsledky o NP-úplnosti dokázané začiatkom 70. rokov ukázali, že ak $P \neq NP$, tak väčšina problémov kombinatorickej optimalizácie, ktorá sa objavuje v obchode, vede a inžinierstve, je prakticky nerealizovateľná: žiadna metóda na ich riešenie nemôže úplne obísť kombinatorickú explóziu. Ako potom zvládnuť takéto problémy v praxi? Jeden možný prístup vyplýva z toho, že často je dostatočne dobré takmer optimálne riešenie: obchodný cestujúci sa pravdepodobne uspokojí s cestou, ktorá bude o niekoľko percent dlhšia ako optimálna. Využívajúc tento prístup začali odborníci hľadať polynomiálne algoritmy, ktoré by zaručovali takmer optimálne riešenia pre NP-úplné problémy kombinatorickej optimalizácie. Vo väčšine prípadov bola záruka úspešnosti aproximačných algoritmov v tvare horného odhadu pomeru ceny riešenia produkovaného algoritmom a ceny optimálneho riešenia.

Niektoré z najzaujímavejších prác o aproximačných algoritmoch so zárukou úspešnosti sa zaoberajú jednorozmerným problémom ukladania do poličiek. V tomto probléme treba skupinu položiek rozličnej veľkosti vložiť do poličiek, ktoré majú všetky rovnakú kapacitu. Cieľom je minimalizovať počet poličiek potrebných na uloženie za takého obmedzenia, že suma veľkostí položiek uložených do ľubovoľnej poličky neprekročí jej kapacitu. Séria článkov publikovaných v polovici 70. rokov o aproximačných algorit-

moch pre ukladanie do poličiek vyvrcholila prácou Davida Johnsona, v ktorej analyzoval zostupný algoritmus prvého uloženia. V tomto jednoduchom algoritme sa položky uvažujú zostupne podľa veľkosti a každá položka, ktorá je na rade, sa umiestni do prvej poličky, ktorá je schopná ju akceptovať. V príklade na obr. 7 sú napríklad štyri poličky, každá s kapacitou 10 a osem položiek s veľkosťou od 2 do 8. Johnson ukázal, že táto jednoduchá metóda zaručuje relatívnu chybu najviac 2/9; inými slovami, počet potrebných poličiek bude najviac o 22 percent väčší ako počet poličiek v optimálnom riešení. Po niekoľko rokov sa tieto výsledky stále zlepšovali a nakoniec sa ukázalo, že relatívnu chybu možno zmenšiť, nakoľko len chceme, hoci polynomiálne algoritmy, ktoré sú na to potrebné, už nemajú jednoduchosť algoritmu, ktorý analyzoval Johnson.



Obr. 7. Uloženie vytvorené zostupným algoritmom prvého uloženia

Výskum v oblasti polynomiálnych aproximačných algoritmov odhalil zaujímavé rozdiely medzi **NP**-úplnými problémami kombinatorickej optimalizácie. Pre niektoré problémy možno relatívnu chybu zmenšiť, nakoľko len chceme, pre ďalšie ju možno znížiť na určitú úroveň, ale zdá sa, že nie ďalej; ďalšie problémy odolávajú pokusom nájsť algoritmus s ohraničenou relatívnou chybou a nakoniec sú určité problémy, pre ktoré by existencia polynomiálneho aproximačného algoritmu s ohraničenou relatívnou chybou implikovala rovnosť $\mathbf{P} = \mathbf{NP}$.

Počas ročnej vedeckej dovolenky, ktorá nasledovala po období môjho administrovania, som začal premýšľať o priepasti medzi teóriou a praxou v oblasti kombinatorickej optimalizácie. Zo strany teórie boli správy neradostné. Takmer všetky problémy, ktoré bolo treba riešiť, boli **NP**-úplné a vo väčšine prípadov polynomiálne aproximačné algoritmy nemohli dať také záruky úspešnosti, ktoré by sa dali použiť v praxi. Na druhej strane bolo mnoho algoritmov, ktoré na prvý pohľad dokonale fungovali v praxi, aj keď im chýbalo teoretické odôvodnenie. Napríklad Lin a Kernighan vymysleli veľmi úspešnú stratégiu lokálneho zlepšovania pre problém obchodného cestujúceho. Ich algoritmus jednoducho začal s nejakou náhodnou cestou a zlepšoval ju pridávaním a odoberaním niekoľkých spojení, kým nevytvoril cestu, ktorú už takýmito lokálnymi zmenami nebolo možné zlepšiť. V špeciálne vykonštruovaných prípadoch sa ich algoritmus správal katastrofálne, ale v praktických prípadoch bolo možné veriť, že dáva takmer optimálne riešenie. Podobná situácia vládla pri simplexovom algoritme, jednej z najdôležitejších výpočtových metód: spoľahlivo riešil veľké problémy lineárneho programovania, ktoré sa objavili pri aplikáciách, napriek tomu, že určité umelo vykonštruované prípady bol nútený riešiť v exponenciálnom počte krokov.

Zdalo sa, že úspech takýchto neexaktných či „z prsta vycucaných" algoritmov bol

empirickým fenoménom, ktorý bolo treba vysvetliť. A zdalo sa tiež, že vysvetlenie tohto fenoménu bude nevyhnutne vyžadovať rozchod s tradičnými paradigmami teórie zložitosti, ktoré ohodnocujú algoritmus podľa jeho správania na najhoršom možnom vstupe, ktorý môže dostať. Tradičná analýza najhoršieho prípadu – dominantný smer v teórii zložitosti – zodpovedá scenáru, v ktorom sú prípady problému, ktoré sa majú riešiť, vytvárané nejakým nekonečne inteligentným protivníkom, ktorý pozná štruktúru algoritmu a vyberá vstupy, ktoré ho donútia k maximálnemu výkonu. Tento scenár vedie k záveru, že simplexová metóda a algoritmus Lina-Kernighana sú beznádejne chybné. Ja som začal presadzovať iný prístup, v ktorom sa predpokladá, že vstupy prichádzajú od užívateľa, ktorý ich jednoducho vyberá podľa nejakého rozumného rozdelenia pravdepodobnosti a nesnaží sa ani mariť algoritmus, ani mu pomáhať.

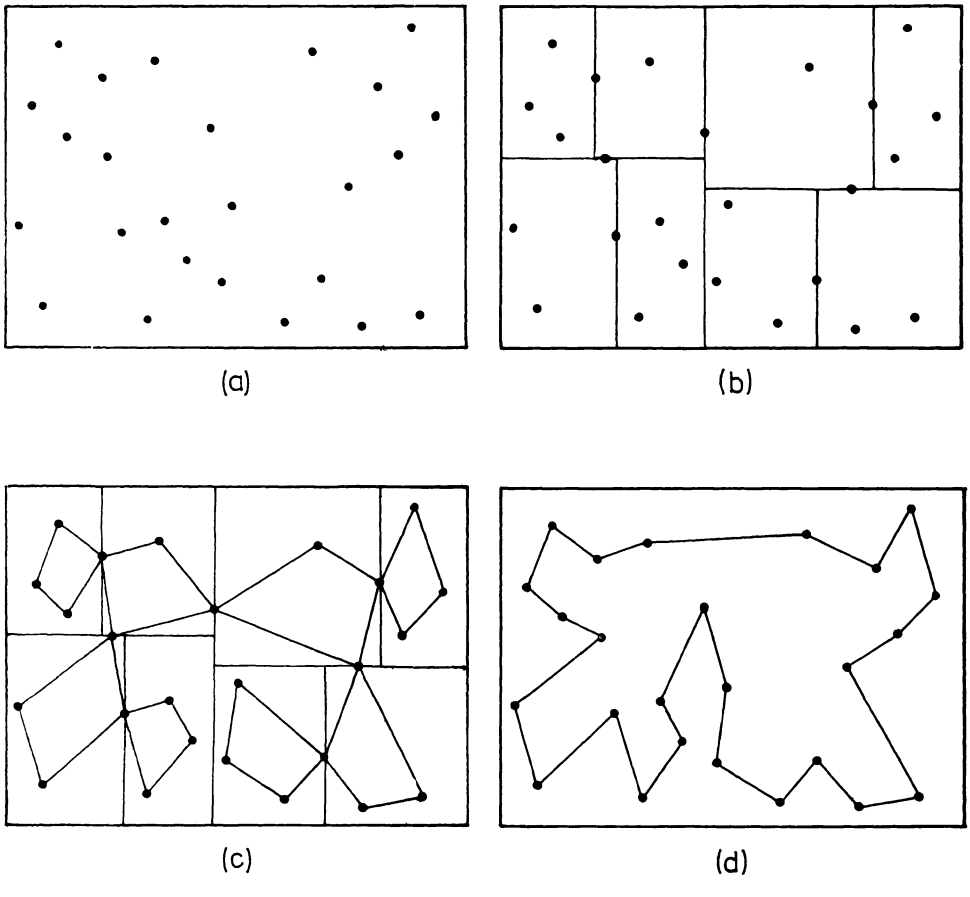
V roku 1975 som sa rozhodol zahryznúť do kyslého jablka a dať sa do výskumu pravdepodobnostnej analýzy kombinatorických algoritmov. Musím povedať, že toto rozhodnutie vyžadovalo určitú odvahu, pretože tento smer výskumu mal svojich odporcov, ktorý celkom správne dôvodili, že neexistuje spôsob, akým sa dá zistiť, aké vstupy sa budú predkladať algoritmu, a že najlepšou zárukou, ak vôbec možno nejakú dostať, je záruka najhoršieho prípadu. Napriek tomu som cítil, že v prípade NP-úplných algoritmov by sme nedostali záruky najhoršieho prípadu, ktoré sme chceli, a že pravdepodobnostný prístup je najlepší a možno jediný spôsob, ako pochopiť, prečo heuristické kombinatorické algoritmy v praxi tak dobre fungujú.

Pravdepodobnostná analýza začína od predpokladu, že jednotlivé prípady problému sa vytvárajú podľa určitého rozdelenia pravdepodobnosti. Napríklad v prípade obchodného cestujúceho jeden možný predpoklad je, že lokalizácia n miest sa odvodzuje nezávisle z rovnomerného rozdelenia nad jednotkovým štvorcom. Podľa predpokladu môžeme študovať rozdelenie pravdepodobnosti dĺžky optimálnej cesty alebo dĺžky cesty produkovanej určitým algoritmom. V ideálnom prípade je cieľom dokázať, že nejaký jednoduchý algoritmus produkuje optimálne alebo takmer optimálne riešenie s vysokou pravdepodobnosťou. Samozrejme takýto výsledok má zmysel iba v tom prípade, ak sa rozdelenie pravdepodobnosti jednotlivých prípadov problémov podobá prípadom, ktoré sa objavujú v reálnom živote, alebo ak je pravdepodobnostná analýza dostatočne robustná na to, aby bola pravdivá pre široký okruh rozdelení pravdepodobnosti.

Medzi najúžasnejšie javy teórie pravdepodobnosti patria zákony veľkých čísiel, ktoré nám hovoria, že kumulatívny efekt veľkého počtu náhodných udalostí je vysoko predvídateľný, aj keď výsledky jednotlivých udalostí sú vysoko nepredvídateľné. Môžeme iste napríklad predvídať, že v dlhej sérii hodov obyčajnou mincou bude približne polovica výsledkov hlava. Pravdepodobnostná analýza odhalila, že podobný jav riadi správanie mnohých algoritmov kombinatorickej optimalizácie, ak sa ich vstupné údaje vyberajú podľa jednoduchého rozdelenia pravdepodobnosti: beh algoritmu sa s veľkou pravdepodobnosťou vyvíja vysoko predvídateľným spôsobom a vytvorené riešenie je takmer optimálne. Článok od Beardwooda, Haltona a Hammersleyho z roku 1960 napríklad ukázal, že ak sa n miest v probléme obchodného cestujúceho vyberá nezávisle podľa rovnomerného rozdelenia z jednotkového štvorca, potom, ak n je veľmi veľké, je dĺžka optimálnej cesty takmer určite veľmi blízko určitej absolútnej konštante krát druhá

odmocnina počtu miest. Motivovaný ich výsledkom som ukázal, že ak počet miest bude extrémne veľký, bude jednoduchý algoritmus rozdeľuj-a-panuj takmer určite produkovať cestu, ktorej dĺžka je veľmi blízka dĺžke optimálnej cesty (pozri obr. 8). Algoritmus začína rozdeľovaním oblasti, kde ležia mestá na obdĺžniky, z ktorých každý obsahuje malý počet miest. Potom vytvára optimálnu cestu cez mestá v každom obdĺžniku. Zjednotenie týchto malých ciest presne pripomína celkovú cestu obchodného cestujúceho, ale odlišuje sa od nej extra návštevami tých miest, ktoré ležia na okraji obdĺžnikov. Nakoniec algoritmus vykonáva určitý druh lokálnej chirurgie na odstránenie týchto redundancií a na vytvorenie výslednej cesty.

Možno citovať mnohé ďalšie príklady, v ktorých jednoduché aproximačné algoritmy skoro určite dávajú takmer optimálne riešenie pre náhodné veľké prípady NP-úplných optimalizačných problémov. Napríklad moja študentka Sally Floyd vychádzajúc zo skorších prác o poličkovom ukladaní od Bentleyho, Johnsona, Leightona, McGeocha a McGeocha, nedávno ukázala, že ak sa ukladané položky vyberajú nezávisle podľa



Obr. 8. Algoritmus rozdeľuj-a-panuj pre problém obchodného cestujúceho v rovine

rovnomerného rozdelenia nad intervalom $[0,1/2]$, potom zostupný algoritmus prvého uloženia skoro určite, nezávisle od toho, koľko je položiek, vytvorí uloženie s menej ako 10 poličkami nevyužitého priestoru.

Jedným z najpozoruhodnejších použití pravdepodobnostnej analýzy bolo jej použitie na problém lineárneho programovania. Geometricky sa tento problém blíži k nájdeniu vrcholov mnohostenu čo najbližšie k určitej externej nadrovine. Algebraicky je ekvivalentný minimalizácii lineárnej funkcie s ohľadom na obmedzenia v tvare lineárnych nerovností. Lineárna funkcia meria vzdialenosť ku nadrovine a ohraničenia lineárnych nerovností zodpovedajú nadrovinám, ktoré ohraničujú mnohosten.

Simplexový algoritmus pre problém lineárneho programovania je metóda postupu smerom do vrchu. Opakovane sa posúva od vrcholu k susednému vrcholu, stále sa približujúc bližšie k externej nadrovine. Algoritmus skončí vtedy, keď sa dostane do vrcholu, ktorý je bližšie ku tejto nadrovine ako ktorýkoľvek susedný vrchol; je zaručené, že taký vrchol je optimálnym riešením. V najhoršom prípade vyžaduje simplexový algoritmus toľko iterácií, že ich počet rastie exponenciálne s počtom lineárnych nerovností potrebných na opísanie mnohostenu, ale v praxi je počet iterácií zriedkakedy väčší ako troj- alebo štvornásobok počtu lineárnych nerovností.

Karl-Heinz Borgwart z NSR a Steve Smale z Berkeley boli prvými odborníkmi, ktorí použili pravdepodobnostnú analýzu na vysvetlenie bezdôvodného úspechu simplexového algoritmu a jeho variantov. Ich analýza závisela na ohodnotení určitých viacrozmerných integrálov. S mojou ohraničenou prípravou v matematickej analýze sa mi ich metódy zdali nepreniknuteľné. Našťastie jeden z mojich kolegov v Berkeley, Ilan Adler, navrhol prístup, ktorý sľuboval uskutočnenie pravdepodobnostnej analýzy vlastne bez výpočtov: použili by sa určité princípy symetrie na vykonanie požadovaného priemerovania a z toho magicky vzíde odpoveď.

Pokračujúc v tejto línii výskumu ukázali sme v roku 1983 Adler, Ron Shamir a ja, že pri rozumne širokej množine pravdepodobnostných predpokladov rastie očakávaný počet iterácií vykonávaných určitou verziou simplexového algoritmu s druhou mocninou počtu lineárnych nerovností. Rovnaký výsledok získal Michael Todd cez viacrozmerné integrály a tiež Adler a Nimrod Meggido. Verím tomu, že tieto výsledky značne prispejú k chápaniu toho, prečo simplexová metóda funguje tak dobre.

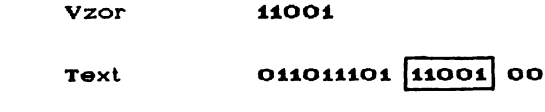
Pravdepodobnostná analýza kombinatorických optimalizačných algoritmov bola hlavnou témou v mojom výskume viac ako desaťročie. V roku 1975, keď som sa prvý raz dal na tento smer výskumu, bolo veľmi málo príkladov na tento typ analýzy. Teraz sú jej venované stovky článkov a všetky klasické problémy kombinatorickej optimalizácie boli podrobené pravdepodobnostnej analýze. Výsledky nám značne pomáhajú porozumieť, do akej miery možno tieto problémy využiť v praxi. Napriek tomu považujem tento pokus iba za čiastočne úspešný. Pretože sme limitovaní našimi technikami, pokračujeme v práci s najjednoduchšími pravdepodobnostnými modelmi a aj vtedy sú mnohé z najzaujímavejších a najúspešnejších algoritmov mimo rámca našej analýzy. Keď je všetko povedané a urobené, návrh praktických kombinatorických optimalizačných algoritmov zostáva viac umením, ako by bol vedou.

Náhodné algoritmy

Algoritmy, ktoré môžu hádzať mincou v priebehu svojho výpočtu, sa navrhovali z času na čas už od počiatkov počítačov, ale systematické štúdium takýchto náhodných algoritmov začalo až okolo roku 1976. Záujem o tieto otázky bol podnietený dvoma prekvapivo efektívnymi náhodnými algoritmami na testovanie, či číslo n je prvočíslo: jeden z týchto algoritmov navrhli Solovay a Volker Strassen a druhý Rabin. Následný Rabinov článok obsahoval ďalšie príklady a motiváciu pre systematické štúdium náhodných algoritmov a doktorská práca Johna Gilla pod vedením môjho kolegu Bluma položila základy všeobecnej teórie náhodných algoritmov.

Aby sme pochopili výhody hádzania mincou, vráťme sa znovu ku scenáru spojenému s analýzou najhoršieho prípadu, v ktorom vševedúci protivník vyberá také prípady, ktoré preveria daný algoritmus čo najprísnejšie. Náhodnosť spôsobí, že správanie algoritmu je nepredvídateľné, aj keď ide o fixovaný prípad, a to môže urobiť ťažkým, ak nie úplne nemožným pre súpera vybrať taký prípad problému, ktorý by pravdepodobne spôsobil ťažkosti. Tu je užitočná analógia s futbalom, v ktorej algoritmus zodpovedá ofenzívnemu tímu a súper defenzívnemu. Deterministický algoritmus je ako tím, ktorého herný plán je úplne predvídateľný, čím dovoľuje druhému tímu dopredu si pripraviť obranné varianty. Ako každý obranca vie, trocha obmeny v obrannom pláne je nevyhnutná na to, aby si defenzívny tím zachoval česť.

Ako konkrétnu ilustráciu výhod hádzania mincou uvediem jednoduchý náhodný algoritmus porovnania vzoru, ktorý sme vymysleli Rabin a ja v roku 1980. Algoritmus porovnania vzoru je fundamentálnym problémom v spracovaní textu. Ak je daný n-bitový reťazec zvaný vzor a omnoho dlhší reťazec bitov nazývaný text, problém je určiť, či sa vzor nachádza ako súvislý blok v texte (obr. 9). Metóda hrubej sily na riešenie tohto problému je porovnávať vzor priamo s každým n-bitovým blokom v texte. V najhoršom prípade je čas výpočtu pri tejto metóde úmerný súčinu dĺžky vzoru a dĺžky textu. V mnohých aplikáciách spracovania textu je táto metóda neakceptovateľne pomalá, ak vzor nie je veľmi krátky.



Obr. 9. Problém porovnania vzoru

Naša metóda obchádza tieto ťažkosti jednoduchým hašovacím trikom. Definujeme „odtlačkovú funkciu", ktorá priradí každému reťazcu n bitov omnoho kratší reťazec zvaný odtlačok. Odtlačková funkcia je vybraná tak, že je možné prechádzať cez text a rýchlo počítať odtlačok každeho bloku dlhého n bitov. Potom namiesto porovnávania každého vzoru s každým takým blokom textu, porovnávame odtlačok vzoru s odtlačkom každého takého bloku. Ak sa odtlačok vzoru líši od odtlačku každého bloku, potom vieme, že sa vzor ako blok nenachádza v texte.

Metóda porovnávania krátkych odtlačkov namiesto dlhých reťazcov podstatne redukuje čas výpočtu, ale obsahuje možnosť chybných porovnaní, ak určitý blok textu má

rovnaký odtlačok ako vzor napriek tomu, že vzor a blok textu nie sú rovnaké. Chybné porovnania sú vážny problém: pre každú jednotlivú voľbu odtlačkovej funkcie skutočne môže protivník skonštruovať taký príklad vzoru a textu, že chybné porovnanie sa objaví v každej pozícii textu. Preto je potrebná ešte nejaká rezervná metóda na obranu proti chybným porovnaniam a výhody odtlačkovej metódy sa zdajú stratené.

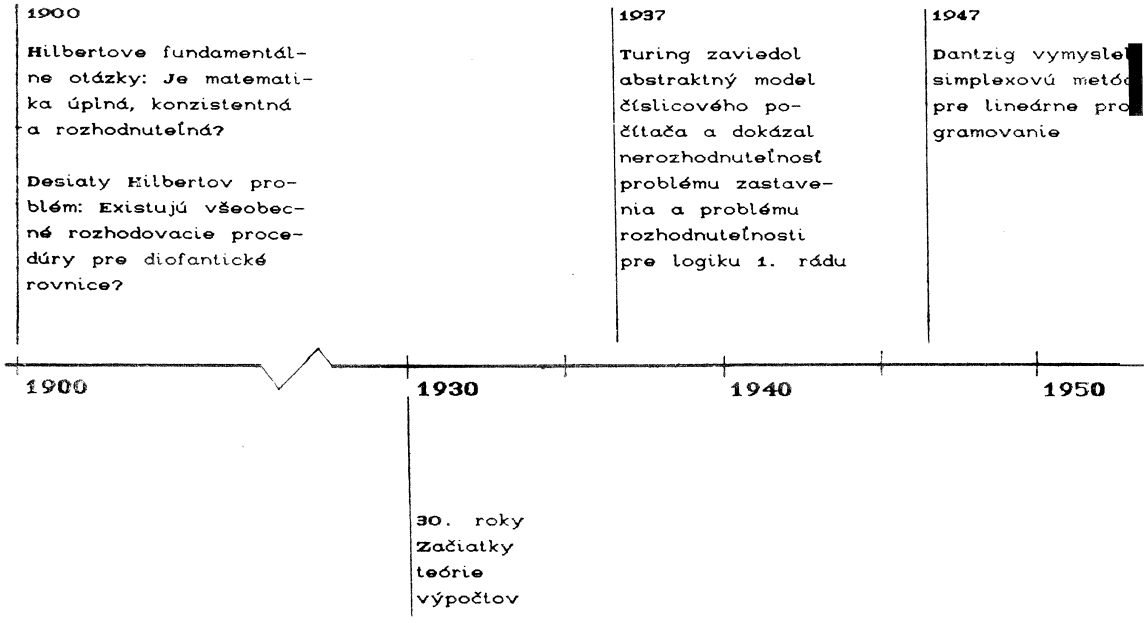
Našťastie výhody tejto metódy možno obnoviť cez náhodnosť. Namiesto práce s jednou odtlačkovou funkciou má náhodná metóda k dispozícii veľkú skupinu rozličných ľahko vypočítateľných odtlačkových funkcií. Ak je zadaný prípad problému, pozostávajúci zo vzoru a textu, algoritmus náhodne vyberie odtlačkovú funkciu z tejto veľkej skupiny a použije túto funkciu na testovanie zhody medzi vzorom a textom. Pretože odtlačková funkcia nie je známa dopredu, je nemožné pre protivníka skonštruovať prípad problému, ktorý by pravdepodobne viedol ku chybným porovnaniam; je možné ukázať, že pravdepodobnosť chybného porovnania bez ohľadu na to, ako sa vyberá vzor a text, je veľmi malá. Napríklad, ak vzor má 250 bitov a text 4000 bitov, môžeme pracovať s 32 bitovými jednoducho vypočítateľnými odtlačkami; a stále je zaručené, že pravdepodobnosť chybného porovnania je menšia ako jedna tisícina v každom možnom prípade. V mnohých systémoch spracovania textu je táto pravdepodobnostná záruka dostatočne dobrá, aby eliminovala potrebu rezervnej procedúry a náš prístup tak výhody znovu získal.

Náhodné algoritmy a pravdepodobnostná analýza sú dva kontrastujúce spôsoby, ako sa odchýliť od analýzy najhoršieho prípadu v deterministických algoritmoch. V prvom prípade je náhodnosť zabudovaná priamo do fungovania algoritmu a v druhom prípade sa predpokladá prítomnosť náhodnosti pri výbere prípadu problému. Prístup založený na náhodných algoritmoch je príťažlivejší z týchto dvoch, pretože sa vyhýba predpokladom o prostredí, v ktorom sa algoritmus bude používať. Avšak náhodné algoritmy ešte nedokázali svoju efektívnosť v boji proti kombinatorickej explózii **NP**-úplných problémov, a tak sa zdá, že sa budú používať oba tieto prístupy.

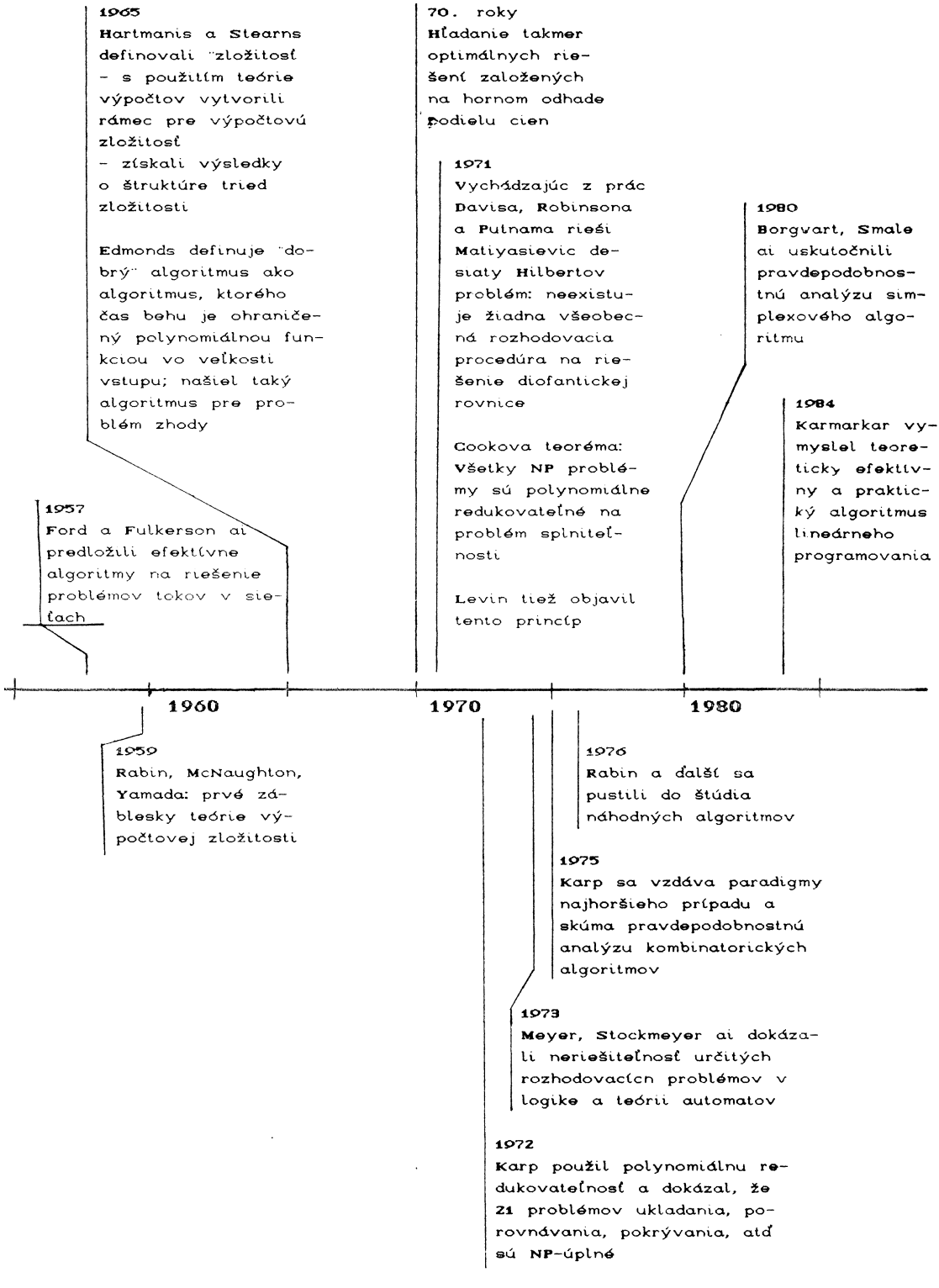
Záver

To ma priviedlo na koniec môjho rozprávania. Rád by som ho uzavrel krátkou poznámkou o tom, čo sa pravdepodobne robí v teoretickej informatike dnes. Vždy, keď sa zúčastňujem pravidelného sympózia ACM Theory of Computing alebo navštevujem Bay Area Theory Seminar alebo vyjdem na vrch za berkeleyským komplexom do Mathematical Sciences Research Institute, kde sa koná ročný kurz vo výpočtovej zložitosti, som unesený významom práce, ktorá sa vykonala v tejto oblasti. Som hrdý na to, že súvisím s oblasťou výskumu, v ktorej sa robí taká vynikajúca práca, a potešený, že som v postavení, kedy z času na čas pomôžem mimoriadne talentovaným mladým odborníkom orientovať sa v tejto oblasti. Ďakujem vám, že som mal možnosť vystúpiť ako zástupca tejto oblasti pri dnešnej príležitosti.

VÝVOJ KOMBINATORICKEJ OPTIMALIZÁCIE



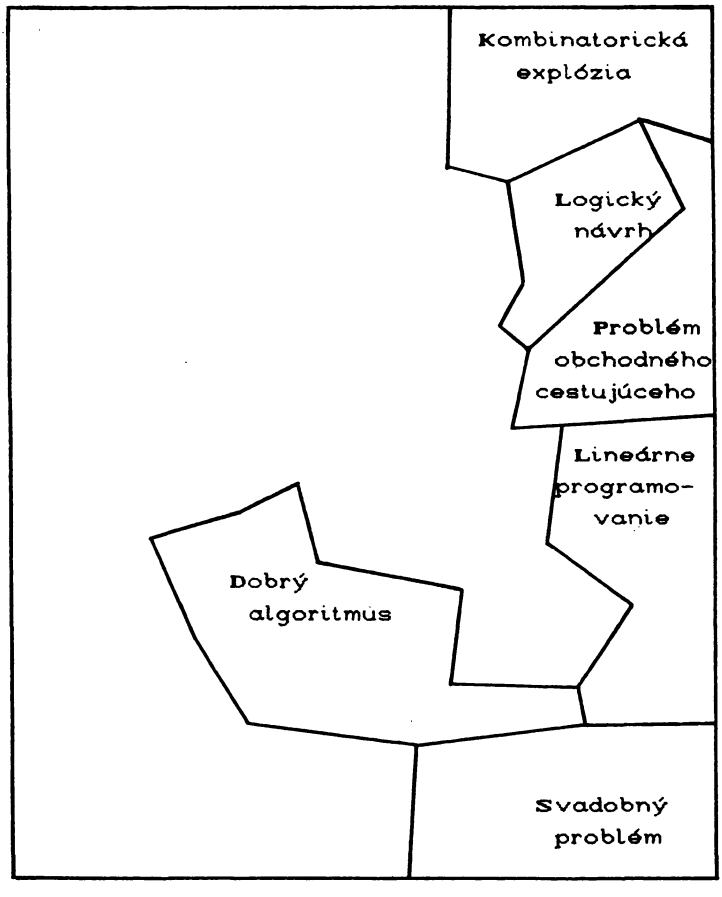
A TEÓRIE VÝPOČTOVEJ ZLOŽITOSTI


1965
Hartmanis a Stearns definovali "zložitosť - s použitím teórie výpočtov vytvorili rámec pre výpočtovú zložitosť - získali výsledky o štruktúre tried zložitosti
Edmonds definuje "dobrý" algoritmus ako algoritmus, ktorého čas behu je ohraničený polynomiálnou funkciou vo veľkosti vstupu; našiel taký algoritmus pre problém zhody
1957
Ford a Fulkerson ai predložili efektívne algoritmy na riešenie problémov tokov v sieťach
70. roky
Hľadanie takmer optimálnych riešení založených na hornom odhade podielu cien
1971
Vychádzajúc z prác Davisa, Robinsona a Putnama rieši Matiyasievic desiaty Hilbertov problém: neexistuje žiadna všeobecná rozhodovacia procedúra na riešenie diofantickej rovnice
Cookova teoréma: Všetky NP problémy sú polynomiálne redukovateľné na problém splniteľnosti
Levin tiež objavil tento princíp
1980
Borgvart, Smale ai uskutočnili pravdepodobnostnú analýzu simplexového algoritmu
1984
Karmarkar vymyslel teoreticky efektívny a praktický algoritmus lineárneho programovania
1960
1970
1980
1959
Rabin, McNaughton, Yamada: prvé záblesky teórie výpočtovej zložitosti
1976
Rabin a ďalší sa pustili do štúdia náhodných algoritmov
1975
Karp sa vzdáva paradigmy najhoršieho prípadu a skúma pravdepodobnostnú analýzu kombinatorických algoritmov
1973
Meyer, Stockmeyer ai dokázali neriešiteľnosť určitých rozhodovacích problémov v logike a teórii automatov
1972
Karp použil polynomiálnu redukovateľnosť a dokázal, že 21 problémov ukladania, porovnávania, pokrývania, atď sú NP-úplné

Skladanie zložitosti

(Doslov od Karen A. Frenkela)

Na ilustráciu „značnej šírky, v akej pôsobí teória zložitosti prostriedkami analogickými ako teória výpočtov" vytvoril Richard Karp túto konceptuálnu mapu alebo skladačku. Na položenie skladačky do roviny použil „algoritmus grafovej planarity". Vzdialenejšie miesta nemusia na prvý pohľad súvisieť, „ale nakoniec ich teória **NP**-úplnosti spojí," hovorí Karp.

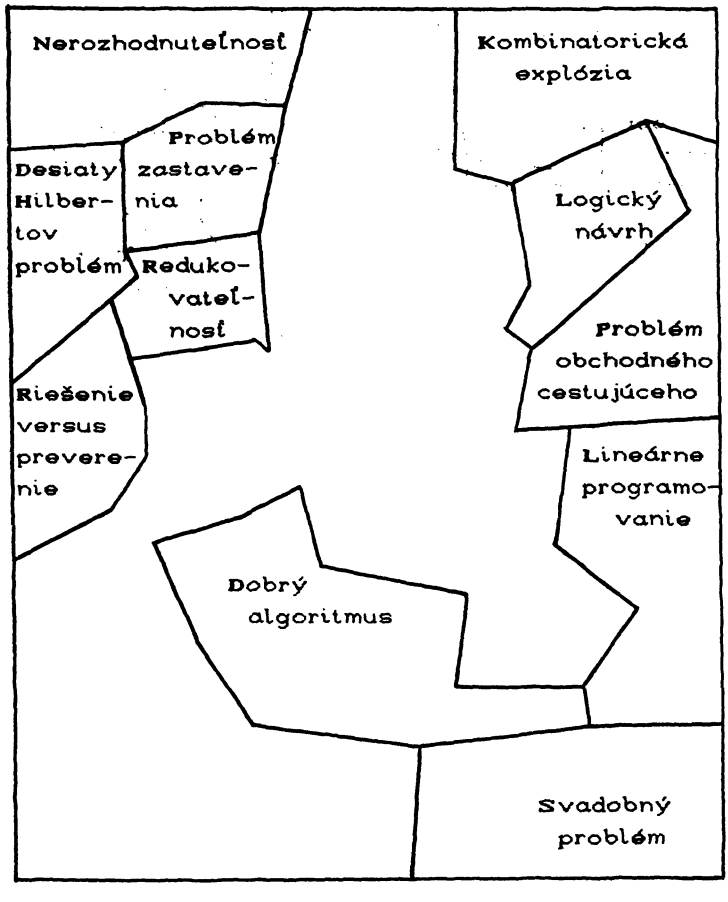


Pravá horná časť skladačky obsahuje pojmy súvisiace s kombinatorickou explóziou a pojmom „dobrého" alebo „efektívneho" algoritmu. Naopak, „zložitosť" spája tieto pojmy s hornou ľavou časťou, ktorá reprezentuje úsilie prvých výpočtových teoretikov.

Problém obchodného cestujúceho je bližšie k pravému hornému rohu, pretože je pravdepodobne prakticky neriešiteľný. Preto je na hranici medzi „**NP**-úplnosťou" a „kombinatorickou explóziou".

Avšak určité hranice sa stierajú. Napríklad „lineárne programovanie" má anomálny status – najširšie používané algoritmy na riešenie tohto problému nie sú dobré z teore-

tického hľadiska a tie, ktoré sú dobré z teoretického hľadiska, často nie sú dobré v praxi. Jedným príkladom je metóda elipsoidov, ktorá priťahovala toľko pozornosti pred šiestimi rokmi. Beží v polynomiálnom čase, ale polynóm je takého vysokého stupňa, že metóda sa ukázala dobrá v technickom zmysle, ale v praxi neefektívna. „Príčinou je to, že náš pojem algoritmu s polynomiálnym časom nezachytáva exaktne pojem intuitívne efektívneho algoritmu", vysvetľuje Karp. „Keď sa dostanete na n^5 alebo n^6, je ťažké pripustiť, že je to skutočne efektívne. Takže Edmondsov pojem dobrého algoritmu nie je dosť ideálny formálny proťajšok dobrého v intuitívnom zmysle. Ďalej, simplexový algoritmus je dobrý

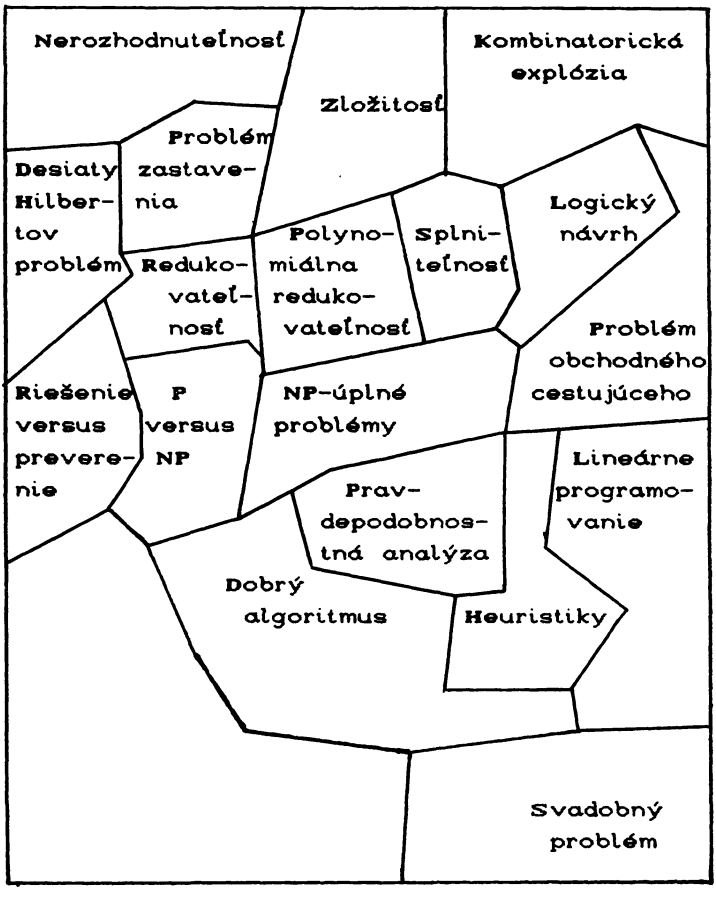


v každom praktickom zmysle," hovorí Karp, „ale nie je dobrý vzhľadom na štandardnú paradigmu teórie zložitosti. Najnovší výsledok v riešení problému lineárneho programovania je algoritmus od Narendru Karmarkara, o ktorom si niektorí myslia, že zdokonaľuje simplexový algoritmus, a ktorý je dobrý v technickom zmysle a vyzerá, že bude efektívny v praxi."

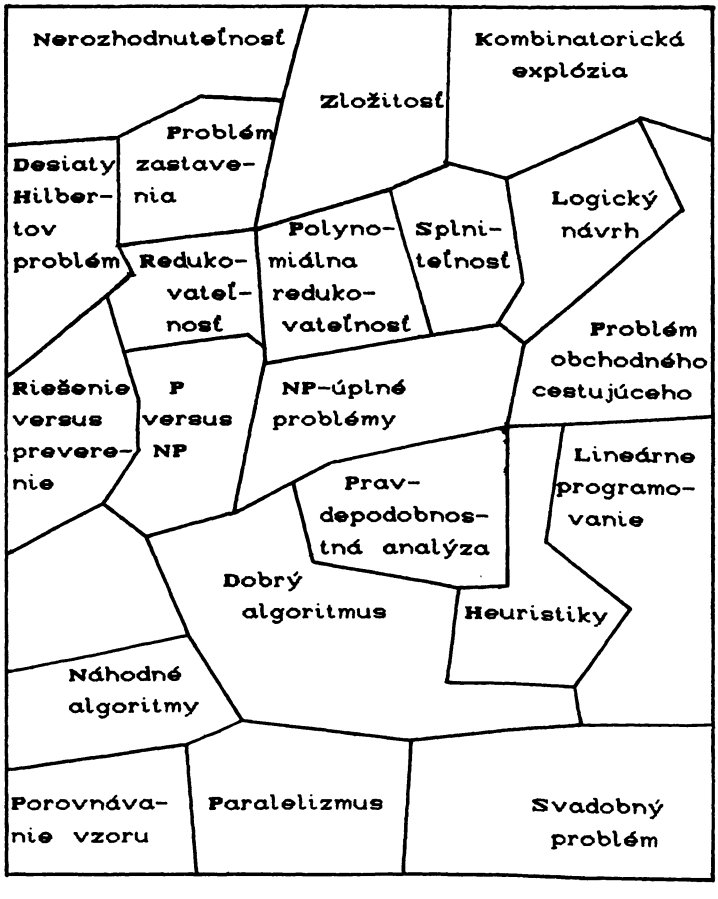
Oblasť „dobrý algoritmus" susedí s „heuristikami", pretože heuristické algoritmy môžu pracovať dobre, ale chýba im teoretické odôvodnenie. Niektoré heuristické algoritmy sú vždy rýchle, ale niekedy nedávajú dobré riešenie. Iné vždy dávajú optimálne

riešenie, ale nie je zaručené, že sú rýchle. Simplexový algoritmus je druhého typu.

„Nerozhodnuteľnosť", „kombinatorická explózia" a „zložitosť" sú v jednej rovine, pretože sú navzájom analogické; nerozhodnuteľnosť obsahuje neohraničené prehľadávanie, kým kombinatorické explózie sú podľa definície veľmi dlhé, ale nie neohraničené hľadania. Teória zložitosti prepája tú medzeru, pretože namiesto pýtania sa, či je problém vôbec riešiteľný, kladie otázky o zdrojoch potrebných na jeho riešenie.



Ľavá dolná oblasť obsahuje oblasti, ktorými sa Karp v súčasnosti najviac zaoberá a ktoré obsahujú otvorené otázky. „Náhodný algoritmus" je napríklad situovaný proti „pravdepodobnostnej analýze", pretože oba sú alternatívami analýzy najhoršieho prípadu pri deterministických algoritmoch. Náhodné algoritmy môžu byť schopné riešiť v polynomiálnom čase také problémy, ktoré deterministické algoritmy nemôžu a mohli by znamenať rozšírenie pojmu dobrého algoritmu. Asi cez návrh programového vybavenia s využitím paralelizmu pre zariadenia nie von neumannovského typu sa algoritmy môžu stať efektívnejšími v praxi. Nakoniec niektoré časti skladačky nie sú ešte definované. „Zodpovedajú neznámej oblasti, ktorú bude treba preskúmať v budúcnosti," hovorí Karp.



Netradiční pohled na Lorentzovu transformaci

Jan Novotný, Brno

1. Úvod

Třebaže je speciální teorie relativity známa a rozvíjena již přes tři čtvrtě století a má stále bohatší a přesnější experimentální podklad [1], přece se její postuláty i některé důsledky stávají čas od času předmětem pochybností a námitek, které někdy proniknou i na stránky fyzikálních časopisů. Jejich hlavním zdrojem je patrně hluboká zakořeněnost klasického (tj. předrelativistického) pohledu, který je v nás stále znovu utvrzován běžnou