Pokroky matematiky, fyziky a astronomie

Juraj Wiedermann

Medze efektivnosti paralelných výpočtových systémov

Pokroky matematiky, fyziky a astronomie, Vol. 33 (1988), No. 2, 81--94

Persistent URL: `http://dml.cz/dmlcz/137707`

Medze efektívnosti paralelných výpočtových systémov

Juraj Wiedermann, Bratislava

1. Úvod

1.1. Výpočtová zložitosť

Pre každú fázu histórie vývoja automatizovaných výpočtových prostriedkov sú príznačné isté praktické obmedzenia: vždy existovali problémy, na riešenie ktorých ani ten najvýkonnejší výpočtový systém nestačil, a tak to bude stále. Ak odhliadneme od zrejmého prípadu neriešiteľných problémov, t. j. takých, pre riešenie ktorých neexistuje algoritmus, býva dôvodom praktickej neriešiteľnosti niektorých ostatných problémov ich veľká výpočtová zložitosť.

Časovú výpočtovú zložitosť problémov meriame obyčajne počtom elementárnych operácií, ktoré musíme vykonať pri riešení problému, t. j. pri realizácii príslušného algoritmu, v závislosti od rozsahu problému. Rozsah problému je pri tom daný dĺžkou popisu (veľkosťou reprezentácie) vstupných dát pre daný problém. Pojem elementárnej operácie sa vždy vzťahuje k určitému modelu (alebo triede modelov) tzv. univerzálneho výpočtového systému, ktorý je schopný realizovať ľubovoľnú elementárnu operáciu v jednotkovom čase. Univerzálnosť sa v tomto prípade chápe v tom zmysle, že príslušný výpočtový systém musí byť potenciálne schopný vypočítať, riešiť ľubovoľný rozhodnuteľný problém; nejedná sa teda o špecializované (napr. analógové) výpočtové systémy, zamerané na riešenie len istých problémov.

1.2. Zvyšovanie rýchlosti procesorov

Bariéru vysokej výpočtovej zložitosti niektorých problémov je možné do istej miery — nie ľubovoľne — prekonať zvyšovaním výpočtovej rýchlosti výpočtových systémov. Zvyšovanie rýchlosti jednotlivých elementárnych operácií je technologická záležitosť: pre každú technológiu realizácie týchto operácií však existuje konštanta, pod ktorú rýchlosť príslušných procesorov nemôže klesnúť. A že existuje ultimatívna hranica, vyplýva z konečnej prenosovej rýchlosti signálu vo fyzikálnych médiách: v ľubovoľnom reálnom výpočtovom elemente potrebujeme nenulový čas na prenos signálu medzi dvoma jeho bodami (napr. medzi jeho vstupom a výstupom, medzi dvoma pamäťovými miestami a pod.), a to v závislosti od vzdialenosti týchto bodov. Z tejto jednoduchej úvahy o. i. vyplýva, že čím rýchlejší počítač, tým menší musí byť — trend, ktorý je zreteľne badať „voľným okom" v doterajšom vývoji elektronických počítačov.

1.3. Paralelizmus

Existuje však aj iná možnosť, ako zvyšovať rýchlosť výpočtových systémov — dokonca aj za predpokladu fixnej operačnej rýchlosti jednotlivých jeho komponent. Túto možnosť nám dáva paralelizmus.

Paralelizmus vo výpočtových systémoch v tomto prípade spočíva jednoducho vo vykonaní viacerých akcií, operácií súčasne v závislosti od rozsahu problému a zrejme neexistuje teoretický limit, ohraničujúci počet týchto súčasných akcií. Intuitívne sa zdá, že čím viac operácií vykonáme naraz — t. j. čím väčšia je miera paralelizmu — tým väčšie urýchlenie oproti sekvenčnému prípadu môžeme dosiahnuť.

A práve skúmanie pravdivosti predchádzajúceho tvrdenia bude cieľom tejto práce.

1.4. Obsah a forma článku

V príspevku podáme v 2. kapitole teoretickú charakterizáciu „klasických" sekvenčných počítačov a v 3. kapitole charakterizáciu súčasných (resp. nastupujúcich) „moderných" paralelných počítačov (superpočítačov, počítačov 5. generácie) z hľadiska teórie zložitosti. Vzájomný vzťah paralelných a sekvenčných počítačov popíšeme pomocou tzv. tézy paralelných výpočtov, ktorá tvrdí, že priestorovo ohraničené výpočty sekvenčných počítačov sú ekvivalentné časovo ohraničeným výpočtom paralelných počítačov. Na základe tejto tézy budeme v 4. kapitole skúmať medze efektívnosti paralelných počítačov v porovnaní so sekvenčnými. Ako východisko pre porovnanie bude slúžiť nasledovný „praktický" analóg Churchovej tézy: „prakticky vypočítateľné", efektívne riešiteľné sú len problémy polynomickej sekvenčnej časovej zložitosti — tzv. *zvládnuteľné problémy*. Uvedieme teoretické i praktické dôvody, pre ktoré nie je možné ani na (reálnych) paralelných počítačoch efektívne riešiť iné než zvládnuteľné problémy, a prečo nebude pravdepodobne možné ani v budúcnosti zostrojiť ešte efektívnejšie počítače, ktoré by umožňovali efektívne riešiť ešte širšiu triedu problémov. Špeciálne pre oblasť konštrukcie a využitia paralelných počítačov a pre umelú inteligenciu má tento výsledok závažné praktické dôsledky — prevažne negatívneho charakteru, ktoré si širšia odborná verejnosť nie vždy plne uvedomuje, resp. nie vždy doceňuje ich dosah. Tieto dôsledky, ktoré sa už prejavujú aj v zameraní základného výskumu v tejto oblasti, budú diskutované v záverečnej 5. kapitole.

Príspevok istým spôsobom nadväzuje na prehľadový článok S. Cooka o teórii výpočtovej zložitosti [19]. Populárnou formou oboznamuje s problematikou, metódami, výsledkami a trendami strojovo orientovanej modernej teórie výpočtovej zložitosti, čím ilustruje jej kľúčovú úlohu v teoretickom zázemí nastupujúcej počítačovej revolúcie.

2. Sekvenčné výpočty

2.1. Modely sekvenčných počítačov

Ak chceme oceniť prínos paralelizmu alebo ho dokonca presne kvantifikovať, je rozumné porovnávať efektívnosť paralelných výpočtov s efektívnosťou sekvenčných výpočtov, pretože práve so sekvenčnými výpočtami máme najväčšie skúsenosti.

Ako sme spomenuli už v úvode, efektívnosť — resp. presnejšie: časovú výpočtovú zložitosť — výpočtov meriame počtom elementárnych operácií potrebných na vyriešenie daného problému na nejakom modeli univerzálneho počítača. V teórii výpočtovej zložitosti poznáme niekoľko modelov sekvenčných počítačov. Najprominentnejší z nich je Turingov stroj a počítač s priamym prístupom.

2.2. Turingov stroj

Turingov stroj je primitívny model sekvenčných výpočtov. Pôvodný návrh pochádza z r. 1936 od anglického matematika A. M. Turinga, ktorý navrhol tento stroj ako prostriedok pre formalizáciu pojmu efektívne (t. j. algoritmicky, mechanicky) vykonateľnej procedúry a dokázal pomocou neho existenciu algoritmicky neriešiteľných problémov [12].

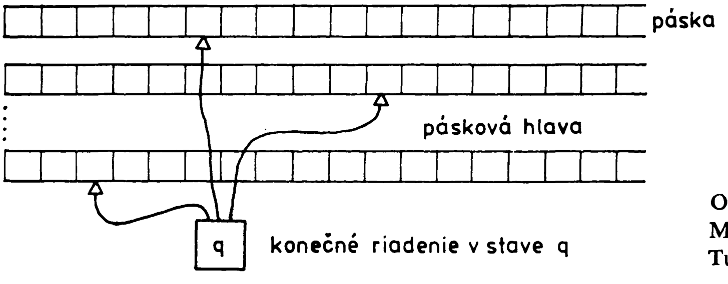
Turingov stroj je zariadenie, ktoré sa skladá z jednej alebo viacerých pások a z konečného riadenia. *Pásky* sú potenciálne nekonečné smerom doprava; sú rozdelené na *políčka* a v každom z nich môže byť zapísaný jeden symbol z konečného počtu znakov. Symboly na páskach sú zapisované a čítané pomocou tzv. *páskových hláv*. Na každej páske je jedna hlava, ktorá v každom výpočtovom kroku skúma práve jedno políčko pásky. Hlavy sú ovládané tzv. konečným riadením. *Konečné riadenie* je v každom okamihu v jednom z konečného počtu *stavov*, v závislosti od tohoto stavu a symbolov čítaných jednotlivými hlavami; Turingov stroj v jednom *výpočtovom kroku* vykoná jednu alebo niekoľko z týchto akcií:

- prepíše symboly čítané zvolenými hlavami inými zvolenými symbolmi,
- posunie vybranými hlavami o jedno políčko doľava, doprava, prípadne ich nechá na mieste,
- zmení stav konečného riadenia.

Tieto akcie sa formálne zapisujú pomocou tzv. *prechodovej funkcie*, v ktorej je zapísané, čo má stroj v danej situácii (popísanej stavom konečného riadenia, a práve čítanými symbolmi jednotlivými hlavami) robiť (t. j. ako prepísať symboly, kam presunúť hlavy, ako zmeniť stav). Prechodová funkcia teda hrá úlohu programu, ktorý Turingov stroj realizuje.

Turingov stroj je schematicky znázornený na obr. 1.

Stroj začína svoj výpočet v zvláštnom, tzv. *počiatočnom stave*. *Vstup* je vtedy zapísaný na prvej páske (v každom políčku jeden symbol) zarovnané doľava. Všetky ostatné políčka (aj na ostatných páskach) neobsahujúce vstup sú prázdne. Hlavy skúmajú prvé políčka



Obr. 1. Mnohopáskový Turingov stroj

na každej páske. Výpočet sa končí, ak stroj dospeje do zvláštneho, tzv. *koncového stavu.* Výsledkom výpočtu je napr. reťazec symbolov zapísaný na zvolenej páske.

Časová zložitosť $T(n)$ výpočtu Turingovho stroja M je daná maximálnym počtom výpočtových krokov stroja M pre všetky možné vstupy dĺžky n.

Pamäťová zložitosť $S(n)$ Turingovho stroja M je daná maximálnym počtom zapísaných políčok pre všetky možné vstupy dĺžky n a všetky pásky stroja M.

Z praktického hľadiska si môžeme Turingov stroj predstaviť ako súčasný počítač „strednej" generácie vybavený páskovými jednotkami, ktorý počíta s týmito obmedzeniami: bez ohľadu na rozsah riešeného problému využíva len niekoľko málo bitov operačnej pamäti (rádove desiatky) a inak ako pamäť využíva páskové jednotky. Tieto jednotky sú rozdelené na bloky, z ktorých každý obsahuje len 1 znak, a dajú sa „prevíjať" maximálne o jeden blok (v ľubovoľnom smere). Je zrejmé, že aj na takomto počítači sa dá „počítať", aj keď veľmi neefektívne. Výhodou Turingovho stroja je jeho koncepčná jednoduchosť (má v podstate jednu jedinú komplexnejšiu inštrukciu), v prospech ktorej bola obetovaná jeho efektivita.

2.3. Churchova téza

Turingov stroj je v dnešnej dobe považovaný za „správny" model výpočtov. Tvrdenie že každý algoritmus (v rozumnom intuitívnom zmysle tohoto slova) je možné realizovať na Turingovom stroji, je známe pod názvom *Churchova téza.* Je zrejmé, že táto téza sa nedá dokázať, pretože dáva do súvisu intuitívny pojem algoritmu s formálnym pojmom Turingovho stroja; dá sa však vyvrátiť: keby niekto prišiel s algoritmom, ktorý sa nedá realizovať na Turingovom stroji.

Nájsť formalizmus pre popis výpočtov, pomocou ktorého by sa dalo vypočítať „viac" než na Turingovom stroji, sa pokúšal celý rad matematikov (medzi inými aj americký matematik A. Church), ale bezúspešne: vždy sa ukázalo, že nový formalizmus popisuje rovnakú triedu výpočtov ako Turingov stroj. Tieto fakty viedli k utvrdzovaniu viery v Churchovu tézu a v dnešnej dobe sa úvahy na tému platnosti tejto tézy prakticky nevyskytujú — skôr naopak, robia sa pokusy o jej „dôkaz", vychádzajúc zo všeobecne platných fyzikálnych zákonov [20].

Turingov stroj môžeme súčasne považovať aj za (správny) *model sekvenčných výpočtov,* pretože v jednom výpočtovom kroku robí len ohraničený počet akcií, a to bez ohľadu na rozsah riešeného problému.

2.4. Počítač RAM

Ako vidno z predchádzajúceho popisu Turingovho stroja, Turingov stroj sa podobá na dnešné počítače len málo. (Mimochodom: Turingov stroj vznikol abstrakciou človeka-počtára, ktorý pre svoje počty používa okrem svojej hlavy (s ohraničenou kapacitou pamäti) len gumu, ceruzku a papier; počítače v r. 1936 ešte neexistovali!)

Abstrakciou súčasných počítačov vznikol *model počítača s priamym prístupom* (RAM — *R*andom *A*ccess *M*achine), (autori Cook a Reckhow, v r. 1972 [2]), ktorý je považovaný za realistický model. Je to vlastne model počítača s jedným *sumátorom* a s potenciálne neohraničenou *pamäťou* s priamym prístupom. Pamäť je rozdelená na *slová*; v každom z nich môže byť reprezentované jedno celé číslo. Počítač RAM má *inštrukčný repertoár*, ktorý mu umožňuje presúvať obsah zvoleného slova do sumátora alebo opačne, realizovať bežné aritmetické operácie, prípadne operácie podmienečného alebo nepodmienečného skoku.

Pre účely „normálneho" programovania sa za vykonanie jednej inštrukcie účtuje jedna časová jednotka. To je realistické do tých čias, pokiaľ číslo vstupujúce do operácie realizovanej danou inštrukciou je možné reprezentovať v jednom slove reálneho počítača. Ak tomu tak nie je — číslo je príliš veľké — účtuje sa za manipuláciu s ním (resp. za jeho reprezentáciu) cena úmerná dĺžke jeho binárneho zápisu — tzv. *logaritmická cena*: číslu veľkosti n sa účtuje cena $[\log_2 n]$ (podrobnosti pozri napr. v [16]).

2.5. 1. počítačová trieda

V teórii výpočtovej zložitosti sa dokazuje, že za realistického predpokladu logaritmickej ceny sa časová efektivita počítača RAM a Turingovho stroja „príliš" nelíši. Turingov stroj dokáže simulovať počítač RAM logaritmickej časovej zložitosti $T(n)$ v čase $c \,.\, T^2(n)$ pre vhodnú konštantu $c > 0$. Pretože platí aj opačné tvrdenie, hovoríme, že počítače *RAM* a Turingov stroj sú *polynomicky časovo ekvivalentné*. Dá sa dokonca dokázať, že predchádzajúce dva počítače sú *lineárne priestorovo ekvivalentné* (to znamená, že počítaču RAM stačí na vykonanie určitého výpočtu asymptoticky rovnako veľká pamäť ako Turingovmu stroju, a naopak) [16].

Ukazuje sa, že poznáme celý rad ďalších modelov výpočtov, ktoré sú polynomicky časovo a lineárne priestorovo ekvivalentné Turingovmu stroju (sú to rozličné modifikácie Turingovho stroja, napr. s viacrozmernými páskami, s viacerými hlavami na jednej páske a pod.) a počítačom RAM, a — napodiv — aj isté jednoduché verzie paralelných počítačov s obmedzeným paralelizmom (napr. jednopáskový paralelný Turingov stroj [14]).

Slot a van Emde Boas v r. 1984 [17] využil túto skutočnosť pre charakterizáciu triedy počítačov, ktorá intuitívne zodpovedá triede sekvenčných počítačov: *1. počítačovú triedu* C_1 tvoria počítače, ktoré sú polynomicky časovo a lineárne priestorovo ekvivalentné Turingovmu stroju (obidve podmienky nemusia platiť súčasne). Bohatosť 1. počítačovej triedy opäť nepriamo potvrdzuje platnosť Churchovej tézy, ešte v silnejšom zmysle: nielen že všetky modely 1. triedy sú ekvivalentné v tom zmysle, že dokážu vypočítať „to isté" (a síce rekurzívne funkcie), ale dokážu tak urobiť približne v rovnakom čase a priestore, ak „rovnaký" chápeme ako polynomickú ekvivalenciu.

2.6. Praktický analóg Churchovej tézy

Z definície 1. počítačovej triedy vyplýva, že v rámci tejto triedy je vlastnosť problémov „byť polynomickej (sekvenčnej) zložitosti" invariantná (či už sa jedná o časovú, alebo pamäťovú zložitosť). To ale súčasne znamená, že táto vlastnosť je do istej dosť širokej miery strojovo nezávislá, pretože nezávisí od konkrétneho modelu, ale len od triedy modelov, ktorá obsahuje celý rad koncepčne odlišných modelov.

Trieda polynomicky časovo zložitých problémov (ktorú označujeme ako *PTIME* — *P*olynomial *T*ime alebo len *P*) má ešte jednu význačnú vlastnosť: z praktického hľadiska je to trieda „maximálne zložitých" problémov, ktoré sa ešte dajú prakticky riešiť.

Toto tvrdenie sa považuje za akýsi *praktický analóg Churchovej tézy*; samozrejme, dá sa diskutovať o tom, či algoritmus zložitosti n^{127} je ešte praktický alebo nie. Skúsenosť však ukazuje, že väčšina problémov polynomickej zložitosti, s ktorými sa v praxi stretávame, je zložitosti lineárnej (napr. vyhľadávanie podreťazcov, výpočet maxima, mediánu, vyhľadávanie, zlučovanie, triedenie prvkov z konečného univerza, niektoré problémy syntaktickej analýzy, sčítanie n-bitových čísiel), kvadratickej (algoritmy na grafoch, triedenie, násobenie dvoch n-bitových čísiel a pod.) alebo kubickej (násobenie matíc, rozoznávanie bezkontextových jazykov). Prakticky motivovaný problém vyššej polynomickej zložitosti ani nepoznáme.

2.7. Zvládnuteľné a nezvládnuteľné problémy

Trieda *PTIME* sa nazýva aj triedou *zvládnuteľných problémov*. Jej protikladom je trieda tzv. *nezvládnuteľných problémov* — t. j. problémov, ktorých riešenie na počítačoch 1. triedy dokázateľne vyžaduje exponenciálny čas.

V súčasnosti je jedným z najväčších otvorených problémov teórie výpočtovej zložitosti problém, kde se začína ríša nezvládnuteľných problémov. Existuje totiž trieda problémov nazývaná *NPTIME* alebo len *NP* (*N*ondeterministic *P*olynomial Time), ktorú tvoria problémy, u ktorých sa dá správnosť daného riešenia v rámci 1. počítačovej triedy overiť v polynomickom čase (resp. riešenie ktorých sa dá nájsť na nedeterministických verziách počítačov 1. triedy v polynomickom čase — pre detaily pozri napr. [16]). Táto trieda obsahuje celý rad dôležitých optimalizačných a kombinatorických problémov. Príkladom je problém hamiltonovej cesty: existuje v danom grafe kružnica, ktorá prechádza každým vrcholom grafu práve raz? Je zrejmé, že overiť, či je daná kružnica hamiltonovská, sa pre daný graf dá jednoducho v polynomickom čase. Intuitívne sa však zdá, že nájsť túto kružnicu je oveľa ťažšie.

Je zrejmé, že $PTIME \subseteq NPTIME$, avšak doteraz nik nenašiel pre niektoré problémy z *NPTIME* (napr. pre problém hamiltonovej cesty) efektívnejší algoritmus než algoritmus exponenciálnej zložitosti a súčasne ale tiež nik nedokázal, že tieto problémy exponenciálny čas naozaj vyžadujú. To znamená, že sa doteraz nevie, napriek skutečne mimoriadnemu úsiliu výskumu, či $PTIME = NPTIME$ (to je známy *P—NP problém*), či teda trieda *NPTIME* je zvládnuteľná, alebo nie. Všeobecne sa však predpokladá, že $PTIME \neq NPTIME$, t. j. že pre problémy z *NPTIME* vo všeobecnosti neexistuje algoritmus polynomickej časovej zložitosti.

3. Paralelné výpočty

3.1. Téza paralelných výpočtov

Z praktického hľadiska má zavádzanie paralelizmu do výpočtov cenu len vtedy, ak podstatným spôsobom, t. j. asymptoticky, rádovo, urýchli sekvenčné výpočty. Ideálne by samozrejme bolo, keby paralelizmus urýchlil výpočty až do tej miery, že by sa dali riešiť aj nezvládnuteľné problémy v rozumnom čase. Preto je dôležité študovať výpočtovú silu (modelov) paralelných počítačov. V dnešnej dobe sa v odbornej literatúre cituje a prezentuje oveľa viac modelov paralelných počítačov než sekvenčných počítačov (pozri napr. prehľadovú štúdiu [15]). Ukazuje sa, že výpočtovú silu drvivej väčšiny týchto počítačov je možné charakterizovať pomocou tzv. tézy paralelných výpočtov, ktorú po prvýkrát formuloval L. M. Goldschlager v r. 1982. Táto téza dáva do súvisu výpočtovú silu sekvenčných a paralelných počítačov (tých, ktoré máme na mysli).

Označme ako $DSPACE\,(T(n))$ triedu problémov, ktoré sú riešiteľné v priestore $T(n)$ na (deterministickom) Turingovom stroji, a ako $//DTIME\,(T(n))$ triedu problémov, riešiteľných v paralelnom čase $T(n)$ na vhodnom, zatiaľ bližšie nešpecifikovanom modeli (deterministického) paralelného počítača.

Téza paralelných výpočtov tvrdí, že polynomicky priestorovo ohraničené výpočty Turingovho stroja (resp. ľubovoľného stroja 1. počítačovej triedy) sú ekvivalentné polynomicky časovo ohraničeným výpočtom paralelných počítačov [5].

Formálne, pre ľubovoľnú funkciu zložitosti $T(n) \geqq \log n$ platí

$$\bigcup_{k=1} DSPACE\,(T^k(n)) = \bigcup_{k=1} //DTIME\,(T^k(n))\,.$$

3.2. 2. počítačová trieda

Tézu paralelných výpočtov môžeme chápať ako definíciu istej „rozumnej" triedy paralelných počítačov. To využil Peter van Emde Boas [18] a definoval tzv. *2. počítačovú triedu C_2* ako triedu počítačov, ktoré spĺňujú tézu paralelných výpočtov. Z tejto tézy paralelných výpočtov (možno) nevidno na prvý pohľad veľkú výpočtovú silu počítačov 2. triedy. Ak sa ale obmedzíme len na výpočty polynomickej zložitosti, t. j. ak položíme $T(n) = n$ vo formálnom vyjadrení tézy, redukuje sa téza na tvar

$$PSPACE = //PTIME\,,$$

kde *PSPACE* označuje triedu problémov riešiteľných v rámci 1. počítačovej triedy v polynomickom priestore a *//PTIME* označuje triedu problémov riešiteľných na počítačoch 2. triedy v polynomickom (paralelnom) čase.

Pozíciu zložitostnej triedy *PSPACE* v hierarchii ostatných tried „zdola" charakterizuje inklúzia $NPTIME \subseteqq PSPACE$, ktorá vyplýva z toho, že ani v nedeterministickom polynomickom čase nemôžeme pri výpočte využiť viac než polynomické množstvo pamäti.

Či je *PTIME* alebo *NPTIME* vlastnou podmnožinou *PSPACE*, nevieme. Isté však je, že *PSPACE* obsahuje celý rad dôležitých problémov, ktoré nevieme riešiť v polynomickom čase v rámci 1. počítačovej triedy (pretože $PSPACE \supseteqq NPTIME$, a pozri diskusiu o triede $NPTIME$ v časti 2.7.), a z tézy paralelných výpočtov vyplýva, že hocijaký počítač z 2. triedy by mal tieto problémy riešiť v polynomickom (paralelnom) čase.

3.3. Počítače 2. triedy

Hoci sme už spomenuli, že existuje relatívne veľa modelov paralelných počítačov, doteraz sme neuviedli žiadny model počítača 2. triedy, takže ani nevieme, či 2. počítačová trieda je neprázdna. Prísne vzaté, pretože nevieme, či $PTIME \neq PSPACE$, môže sa stať, že $C_1 \subseteqq C_2$ – totiž v prípade (nepravdepodobnom), že by platilo $PTIME = PSPACE$. Z toho by už vyplývalo, že $PTIME = ||PTIME$, a teda 2. počítačová trieda by bola iste neprázdna.

V ďalšom ukážeme zaujímavejší prípad, že aj za predpokladu $PTIME \neq PSPACE$ je 2. počítačová trieda neprázdna. Uvedieme dva význačné modely paralelných počítačov a niektoré ich variácie, o ktorých je známe, že patria do 2. počítačovej triedy. Obidva sú zovšeobecnením počítača RAM a možno ich považovať za viac-menej realistické modely paralelných počítačov. Navyše, každý z modelov paralelných počítačov, ktoré uvedieme v ďalšom, reprezentuje špecifický druh paralelných počítačov. V prvom z nich budú procesory komunikovať cez spoločnú globálnu pamäť a v druhom budú komunikačné cesty „zadrôtované" vo forme fixnej komunikačnej siete.

3.4. Počítač pre spracovanie polí

Model počítača pre spracovanie polí (APM – *A*rray *P*rocessing *M*achine) navrhli v r. 1984 autori van Leeuwen a Wiedermann [13]. Hlavným motívom ich návrhu bola snaha o čo najrealistickejší, ale z formálneho hľadiska čo najjednoduchší, elegantný model paralelného počítača, ktorý by hral v triede C_2 podobnú úlohu ako RAM v triede C_1, t. j. aby sa analýzou algoritmov pre tento model počítača dali získať výsledky, ktoré by zodpovedali zložitosti zodpovedajúcich reálnych programov realizovaných na existujúcich paralelných počítačoch.

Z architektonického hľadiska vyzerá APM podobne ako počítač RAM. Inštrukčný repertoár zahrňuje v sebe okrem bežných inštrukcií počítača RAM (ktorým sa v kontexte APM hovorí *skalárne inštrukcie*) ešte aj tzv. *vektorové inštrukcie.* Každej skalárnej inštrukcii zodpovedá príslušná vektorová inštrukcia. Vektorové inštrukcie umožňujú APM pracovať naraz s celými vektormi alebo poliami, ktoré sú adresované pomocou adresy prvej a poslednej komponenty vektora (poľa). Inštrukcie sa realizujú paralelne prostredníctvom špeciálneho tzv. *vektorového sumátora,* ktorý je v centrálnom procesore, tým spôsobom, že nad každou komponentou vektora sa vykoná zodpovedajúca skalárna inštrukcia.

Špeciálnou črtou APM je použitie maskovania. *Maskovanie* umožňuje pri realizácii vektorových inštrukcií zabrániť realizácii zodpovedajúcej skalárnej inštrukcie nad zvolenými, tzv. maskovanými komponentami. Maskovanie sa riadi pomocou maskovacích vektorov. Sú to booleovské vektory rovnakej dĺžky, ako je vektor, ktorý maskujú. Maskovaným komponentám zodpovedajú v maskovacom vektore jednotky, ostatným nuly. Maskovacie vektory sa generujú a sú uložené v spoločnej globálnej pamäti podobne ako ostatné vektory.

Cena skalárnych a vektorových inštrukcií môže byť buď jednotková, alebo logaritmická. V druhom prípade je princíp výpočtu logaritmickej ceny skalárnych inštrukcií obdobný ako v prípade RAMu a logaritmická cena vektorovej inštrukcie je daná „najdrahšou" cenou spomedzi cien príslušných skalárnych inštrukcií, pomocou ktorých sa vektorová inštrukcia paralelne realizuje.

Z popisu počítača APM je vidno, že tento počítač modeluje súčasne prakticky jediné komerčne vyrábané paralelné počítače, tzv. *superpočítače* [6, 7], takže 2. počítačová trieda je neprázdna aj z praktického hľadiska. Všimnime si ešte, že podľa uznávanej Flynnovej taxonómie paralelných počítačov [3] sa jedná o počítač typu SIMD (*S*ingle *I*nstruction, *M*ultiple *D*ata Stream — približne jedna inštrukcia, mnoho dát), pretože APM realizuje v jednom kroku tú istú skalárnu inštrukciu nad rozličnými dátami.

3.5. Paralelný RAM

Paralelný RAM (PRAM) autorov Savitcha a Stimsona z r. 1979 [10] je zvláštny druh počítača RAM, ktorý je schopný paralelne realizovať až $k \geqq 2$ rekurzívnych volaní. Rekurzívne volanie sa pri tom realizuje ako spustenie ďalšieho RAMu, ktorému sa odovzdá v dohovorených registroch ohraničený počet parametrov. Tento RAM môže podobným spôsobom volať ďalších k RAMov, takže jeden počítač môže v lineárnom čase aktivizovať až exponenciálny počet RAMov, ktoré všetky pracujú paralelne. Keď je rekurzívny výpočet skončený, t. j. keď niektorý počítač na istej úrovni rekurzie skončí svoj výpočet, vráti odpoveď volajúcemu počítaču opäť v dohodnutých registroch. Z popisu činnosti PRAMu je vidno, že rekurzívne volania tvoria k-árnu stromovú štruktúru, ktorá zodpovedá komunikácii (odovzdávaniu parametrov) medzi jednotlivými procesormi. Je zrejmé, že rekurzívne volané počítače môžu paralelne konať rozličné výpočty, ktoré sa riadia pomocou prenášaných parametrov volaní. PRAM je preto počítač typu MIMD — *M*ultiple *I*nstruction, *M*ultiple *D*ata Stream (približne: mnoho inštrukcií, mnoho dát).

Za reálne prototypy počítača PRAM môžeme považovať počítač RIMMS, vyvíjaný na univerzite v Newcastle upon Tyne v Anglicku [4] alebo počítač EC 27 04 konštruktéra Torgašova, vyvinutý v Centre výpočtovej techniky v Moskve [8].

Zaujímavú modifikáciu počítača PRAM skúmal Savitch [9], ktorý obohatil inštrukčný repertoár jednotlivých RAMov, ktoré spolu tvoria výslednú stromovú počítačovú sieť, o lispovské inštrukcie pre manipulácie s reťazcami (rozdeľovanie a spájanie reťazcov). Na výsledný výpočtový systém sa môžeme pozerať ako na špecializovaný paralelný lispovský procesor, veľmi pripomínajúci niektoré predstavy o počítačoch 5. generácie [11]. Autor o ňom dokázal, že tiež patrí do 2. počítačovej triedy.

3.6. Dôkaz príslušnosti k 2. počítačovej triede

O predchádzajúcich dvoch počítačoch a o celom rade ďalších (pozri napr. [15]) je známe, že patria do 2. počítačovej triedy. Preto bolo potrebné dokázať, že každý z týchto modelov splňuje tézu paralelných výpočtov. Dôkaz sa zvyčajne robí v dvoch krokoch. Najprv sa ukáže časovo polynomicky zložitá simulácia polynomicky priestorovo ohraničených výpočtov Turingovho stroja na zvolenom modeli paralelného počítača a potom opačne.

V prvom prípade sa využije vlastne hrubá sila — schopnosť paralelného počítača aktivizovať v polynomickom čase až exponenciálny počet procesorov. To mu umožní ďalej v polynomickom čase vygenerovať všetky možné opísania okamžitého stavu pamäti Turingovho stroja, pretože v danom priestore polynomickej veľkosti existuje len konečný, exponenciálny počet rozličných opísaní pamäte. Potom už len zostáva preveriť, či medzi týmito okamžitými opísaniami pamäte existuje výpočtová cesta začínajúca v počiatočnom a končiaca v koncovom opísaní stavu pamäte. To sa deje postupným paralelným prepájaním nadväzujúcich výpočtových ciest dĺžky 1, 2, 4, 8, ... atď., až sa po polynomickom počte krokov preveria všetky možné výpočtové cesty až exponenciálnej dĺžky. Je teda zrejmé, že celá simulácia sa dá realizovať v polynomickom paralelnom čase, ale za cenu využitia exponenciálneho počtu procesorov.

V prípade opačnej simulácie je ťažkosť v tom, že pre simuláciu paralelného počítača, ktorý v polynomickom čase môže „zaplniť" až exponenciálne veľkú pamäť (napr. ako v predchádzajúcom prípade), máme k dispozícii v Turingovom stroji len priestor polynomickej veľkosti. Táto ťažkosť sa obchádza tak, že Turingov stroj si vôbec priebežne nepamätá údaje vznikajúce v jednotlivých procesoroch paralelného počítača, ale vždy si ich znovu a znovu v prípade potreby vypočíta, naštartujúc výpočet od počiatku a opakujúc len tie čiastočné výpočty, ktoré vedú k výpočtu požadovanej hodnoty.

Detaily dôkazu a prípadné ďalšie spôsoby dokazovania príslušnosti do C_2 pozri napr. v práci [15].

3.7. Význam štúdia modelov paralelných počítačov

Teoretickým dôvodom pre skúmanie veľkého počtu a hľadanie stále nových modelov paralelných počítačov je skutočnosť, že ani pri jednom z nich, videnom nie v kontexte ostatných, si nemôžeme byť istí, či vystihuje správne skúmané vlastnosti (napr. silu paralelizmu); ľahko by sa totiž mohlo stať, že niektorá vlastnosť by bola typická len pre daný model, bola by teda „strojovo závislá". Ak však platí pre celú triedu modelov, koncepčne dosť rozličných, stúpa naša istota, že skúmaná vlastnosť je dobre definovaná, robustná, nezávislá od konkrétneho modelu. V prípade počítačov 2. triedy je táto vlastnosť reprezentovaná tézou paralelných výpočtov.

Samozrejme, že výskum modelov paralelných počítačov má aj dôležité praktické aspekty — stačí si uvedomiť, že obrovský potenciál mikroštruktúrnej technológie sa dá efektívne využiť len, ak nájdeme efektívne paralelné algoritmy a paralelné architektúry pre využitie veľkého množstva malých, ale výkonných procesorov.

Na jednej strane počítače 2. triedy so spoločnou globálnou pamäťou poskytujú vhodný konceptuálny rámec pre návrh a analýzu paralelných algoritmov. Na druhej strane modely 2. počítačovej triedy, tvorené sieťou procesorov, majú koncepčne bližšie k možnostiam súčasnej technológie výroby počítačov a poskytujú preto dôležité vodítko pre návrh architektúry týchto počítačov.

O životaschopnosti a adekvátnosti pojmu 2. počítačovej triedy svedčí aj skutočnosť, že ako sme naznačili aj v predchádzajúcej časti, táto trieda obsahuje aj modely reálnych, existujúcich paralelných počítačov — superpočítačov a počítačov 5. generácie.

4. Medze efektívnosti paralelných výpočtov

4.1. Polynomický paralelizmus

Ak za prakticky realizovateľné, zvládnuteľné problémy pre sekvenčné počítače považujeme len problémy polynomickej časovej zložitosti, musíme z podobných príčin v kontexte paralelných počítačov považovať za zvládnuteľné len tie problémy, ktorých riešenie vyžaduje najviac polynomické množstvo procesorov. Takýto počítač na svoju výrobu spotrebuje „len" polynomické množstvo materiálu a na svoju prevádzku „len" polynomické množstvo energie, a to vzhľadom k rozsahu riešených problémov.

Zdá sa teda oprávnené študovať len také triedy problémov, ktoré sa dajú efektívne riešiť len pomocou tzv. *polynomického paralelizmu* využitím najviac polynomického počtu procesorov a najviac polynomického času. Všimnime si, že z toho už nutne vyplýva, že takéto triedy budú určite podtriedami *PTIME* (prečo?) a takisto, že nie sú reálne nádeje na riešenie nezvládnuteľných problémov pomocou počítačov 2. triedy v polynomickom čase, pretože z tézy paralelných výpočtov, resp. z jej dôkazu pre konkrétne počítače (pozri časť 3.6.) vyplýva, že vtedy tieto počítače môžu zapojiť do výpočtu exponenciálny počet procesorov.

4.2. Zvládnuteľné problémy pre druhú počítačovú triedu

Ak má byť polynomický paralelizmus prínosom voči sekvenčnému spracovaniu, požadujeme ďalej, aby výsledné paralelné riešenie bolo podstatne asymptoticky rýchlejšie než najlepšie možné sekvenčné riešenie.

Ak za dostatočne efektívne paralelné riešenie považujeme riešenie tzv. *polylogaritmickej zložitosti* $O(\log^k n)$, pre nejaké $k \geqq 1$, tak spojením predchádzajúcich dvoch podmienok dostaneme atraktívnu triedu problémov, ktorú nazval S. Cook [1] ako *NC*: sú to problémy riešiteľné na paralelných počítačoch 2. triedy v polylogaritmickom čase a súčasne s polynomickým množstvom procesorov.

Je zrejmé, že $NC \subseteq PTIME$, ale nie je známe, či inklúzia je vlastná alebo nie. Tento problém sa svojou perzistentnosťou voči definitívnemu vyriešeniu a svojím významom podobá na známy $P - NP$ problém (pozri časť 2.7.), pretože dôkaz $NC = PTIME$ by znamenal, že pre každý zvládnuteľný problém by existoval efektívny paralelný algoritmus.

Na jednej strane je známe, že do triedy *NC* patrí mnoho prakticky dôležitých problémov: súčet alebo súčin dvoch n-bitových čísiel, celočíselné alebo booleovské násobenie matíc, triedenie n n-bitových čísiel, celočíselné delenie, hľadanie minimálnej kostry grafu a pod. [1]. Tieto skutočnosti, zdá sa, nasvedčujú rovnosti obidvoch tried.

Na druhej strane však poznáme niektoré dôležité problémy z *PTIME*, ako je napr. problém unifikácie alebo problém prehľadávania grafu do hĺbky, pre ktoré sa nijako nedarí ukázať, že patria do *NC* — zdajú sa byť inherentne sekvenčné, ťažko alebo vôbec nie paralelizovateľné. Tomu nasvedčuje aj iná ich teoretická charakterizácia; na rozdiel od predchádzajúcich problémov obidva predstavujú tzv. *PTIME*-úplné problémy; ide o akési „najťažšie" problémy v triede *PTIME*, na ktoré sa dá každý iný problém z *PTIME* transformovať v logaritmickom priestore. Doteraz však nik neukázal, že tieto problémy nepatria do *NC*, takže základný problém vzťahu medzi oboma triedami zostáva otvorený. Je to jeden z centrálnych problémov súčasnej teórie paralelnej výpočtovej zložitosti.

4.3. Za hranice 2. počítačovej triedy

Hlavným praktickým — aj keď negatívnym — dôsledkom predchádzajúcich úvah je skutočnosť, že v rámci 2. počítačovej triedy nie je možné očakávať efektívne riešenie sekvenčne nezvládnuteľných problémov. Natíska sa však otázka, či nie je mysliteľný ešte nejaký iný druh paralelných počítačov, ktoré by boli z výpočtového hľadiska ešte efektívnejšie než počítače 2. triedy.

Veľký počet známych koncepčne odlišných modelov počítačov v triede C_2 naznačuje, že bez pridania nejakej ďalšej fundamentálnej výpočtovej schopnosti paralelizmus samotný nestačí na prekonanie hranice triedy C_2, nech už akokoľvek mysliteľne meníme architektúru zodpovedajúcich počítačov. A skutočne, ukazuje sa, že za hranice triedy C_2 sa dostaneme, ak k paralelizmu typou MIMD (pozri časť 3.5.) pridáme nedeterminizmus. Nedeterministický PRAM dokáže napr. riešiť (nezvládnuteľné) problémy z triedy *NPTIME* v polylogaritmickom čase a problémy dokázateľne exponenciálnej sekvenčnej zložitosti v polynomickom čase.

Pretože v kontexte počítačov je nedeterminizmus asi rovnako dobre technicky realizovateľný ako jasnovidectvo, zdá sa, že 2. počítačová trieda predstavuje z hľadiska efektívnosti ultimatívnu triedu paralelných počítačov, za hranice ktorej sa nikdy nedostaneme.

5. Záver

Predchádzajúce výsledky je zaujímavé hodnotiť v aktuálnom kontexte vlny nadšenia pre počítačové systémy 5. generácie a umelú inteligenciu.

Potenciál súčasnej technológie výroby počítačov — a tu máme na mysli najmä technológiu VLSI a ULSI — je obrovský. Doteraz však nie je jasné, ako ho využiť, t. j. nie je jasné, aké paralelné počítače je najvýhodnejšie budovať. Existujúca teória však dosť jasne naznačuje, že sa pravdepodobne nijako nebudeme môcť dostať za rámec 2. počítačovej triedy. Pokiaľ by sme sa teda mali rozhodnúť pri realizácii skutočného počítača 2. triedy pre niektorý z existujúcich modelov, treba dať prednosť z technologického hľadiska čo najjednoduchšiemu modelu, pretože z príslušnej teórie vyplýva, že z hľadiska ich výpočtovej sily medzi nimi nie sú podstatné rozdiely.

Ďalej je známe, že mnohé dôležité problémy, ktoré rieši umelá inteligencia, najmä v oblasti dokazovania, odvodzovania faktov, optimalizácie, plánovania a riadenia, patria z výpočtového hľadiska medzi nezvládnuteľné problémy. To s veľkou pravdepodobnosťou znamená, že na sekvenčných počítačoch sú exponenciálnej zložitosti, pokiaľ trváme vždy na ich presnom riešení, a že zostanú nezvládnuteľné aj na hocijako efektívnom reálnom paralelnom počítači. Z príslušných úvah vyplýva, že paralelizmus má praktickú cenu jedine vtedy, ak podstatne urýchli riešenie problémov pomocou polynomického množstva procesorov. To sa ďalej oplatí asi len v tých aplikáciách, kedy ide o opakované riešenie tých istých problémov v reálnom čase, ako sú napr. problémy rozoznávania obrazov, signálov a pod.

Nemožno však očakávať, že by nám paralelizmus umožnil riešenie problémov väčšieho rozsahu, ktoré by sme nedokázali ako-tak efektívne riešiť už na súčasných sekvenčných počítačoch. V týchto prípadoch nevedie cesta cez urýchľovanie počítačov, ale cez návrh efektívnejších algoritmov pre daný problém. Takýmito efektívnejšími algoritmami sú pravdepodobnostné a aproximačné algoritmy, ktorým sa v poslednej dobe venuje v teórii veľká pozornosť. V súčasnej dobe sa zdá, že ak je to vôbec možné, tak jedine tieto algoritmy spolu s paralelizmom môžu spôsobiť dlho očakávaný a už ohlasovaný zvrat v umelej inteligencii.

Literatúra

[1] Cook, S. A.: *A Taxonomy of Problems with Fast Parallel Algorithms.* Preprint, zaslané do Information and Control, 1984.

[2] Cook, S. A. - Reckhow, R. A.: *Time bounded random access machines.* JCSS Vol. *7*, str. 354 až 375, 1973.

[3] Flynn, M. J.: *Some computer organizations and their effectivness.* IEEE. Trans. Comput., *C-21*, 1972, str. 948—960.

[4] Foti, L. et al.: *Reduced Instruction Set Multi-microcomputer System* (*RIMMS*). Computing Laboratory, University of Newcastle upon Tyne, 1983.

[5] Goldschlager, L. M.: *A universal interconnection pattern for parallel computers.* J. ACM *29*, 1982, str. 1073—1086.

[6] Hockney, R. W. - Jesshope, C. R.: *Parallel Computers.* Hilger, Bristol, 1981.

[7] Hwang, K. S. - Su, P. - Ni, L. M.: *Vector computer architecture and processing techniques.* In: M. C. Yovits (Ed.), Advances in Computers, Vol. *20*, Acad. Press, New York, N. Y., 1981, str. 115—197.

[8] Klas, A. - Wiedermann, J. - Ružička, P. - Bartoš, P.: Správa zo služobnej cesty v ZSSR č. 17/85, VUSEI-AR, Bratislava, 1985.

[9] Savitch, W. J.: *Parallel random access machines with powerful instruction sets.* Math. Syst. Theory *15*, 1982, str. 191—210.

[10] Savitch, W. J. - Stimson, M. J.: *Time bounded random access machines with parallel processing.* J. ACM *26*, 1979, str. 103—118.

[11] TRELEAVEN, P. C. - LIMA, I. G.: *Future Computers: Logic, Data Flow, ..., Control Flow?* Computer, March 1984, str. 47—56.

[12] TURING, A. M.: *On computable numbers with an application to the Entscheidungsproblem.* Proc. London Math. Soc., Ser. 2, *42*, str. 230—165, 1936—7.

[13] VAN LEEUWEN, J. - WIEDERMANN, J.: *Array Processing Machines: An Abstract Model.* BIT Vol. *27*, 1987, s. 25—43.

[14] WIEDERMANN, J.: *Parallel Turing Machines.* Techn. Rep. RUU-CS-84-11, Dept. of Comp. Sci., Utrecht, 1984.

[15] WIEDERMANN, J.: *Modely synchrónnych paralelných počitačov.* In: *Distribuované a paralelné systémy*, edícia Aktá, VUSEI-AR Bratislava, 1985.

[16] WIEDERMANN, J.: *Zložitosť (Strojovo orientovaná teória zložitosti sekvenčných a paralelných výpočtov).* In: zborník seminára SOFSEM '85, zotavovňa Magura, Ždiar, Vysoké Tatry, 1985.

[17] SLOT, C. - VAN EMDE BOAS, P.: *On tape versus core: an application of space efficient perfect hash function to the invariance of space.* In: Proceedings of STOC '84, Washington D. C., 1984.

[18] VAN EMDE BOAS, P.: *The second machine class: models of parallelism.* In: J. VAN LEEUWEN, J. K. LENSTRA, and A. H. G. RINNOY KAN (eds.), *Parallel Computers and Computations*, CWI Syll., Centre for Mathematics and Computer Science, Amsterdam, 1985.

[19] COOK, A. S.: *Prehľad teórie výpočtovej zložitosti.* Pokroky matematiky, fyziky a astronomie, ročník *32*, č. 1, 1987, s. 12—29.

[20] GANDY, R.: *Church's thesis and principles for mechanisms.* In: J. BARWISE, H. J. KLEISER, and K. KUNEN (eds), The Kleene Symposium, North-Holland, Amsterdam, 1980, s. 120—148.

diskuse

KOMENTÁŘE

k článku M. Rojka a L. Pekárka „Pokus o novou koncepci vyučování fyzice na střední škole" (PMFA *32* (1987), č. 1, str. 37—46)

● Koncepce fyzikálního vzdělávání založená na strukturní systematice jako hlavní ose, z níž vycházejí jednotlivé fenomenologické oblasti, může mít v očích už erudovaného fyzika velkou přitažlivost. Klasifikační a vysvětlovací potence struktury je mimořádně velká. Její plné využití předpokládá však už značné znalosti a v počátcích fyzikálního vzdělávání není tedy přirozeně možné. Avšak nosnost koncepce vyučování založené přednostně na struktuře může být přesto už i na středoškolském stupni značná, podaří-li se s ní organicky skloubit makroskopickou fenomenologii potřebnou i ve funkci srozumitelných empiricko-experimentálních východisek.

Deklarativní varianta by vůbec neměla být uvažována; jen by posilovala nežádoucí verbalismus a formalismus.

Konkrétní podklad pro kvalifikovaný odhad vyhlídek varianty založené na empiricko-experimentálním přístupu může vzniknout teprve propracováním většího počtu relativně svébytných celků. V článku podané vysvětlení integračně vysoce hodnotného pojmu interference je zdařilou ukázkou. Celkový materiál, který má nakonec vzniknout, je však i každou takovou jednotlivou částí natolik hodnotný, že zasluhuje, aby byl i takto po částech pro svou mnohostrannou použitelnost průběžně publikován. Vyvolá-li nejen pochvalnou