Ondřej Machek; Jan Hejda

Paralelní počítání

Rozhledy matematicko-fyzikální, Vol. 91 (2016), No. 4, 19–29

Persistent URL: http://dml.cz/dmlcz/146687

Paralelní počítání

Ondřej Machek, VŠE v Praze – Jan Hejda, ČVUT v Praze

Abstract. In recent years, scientists have discussed the possibilities of increasing the computing power of computers. One of the possible approaches is the use of parallel computing. In the paper, we deal with the essentials of parallel design of algorithms: basic ideas, architectures, interconnection networks, and we present the parallel reduction algorithm.

Úvod

Současná věda vyžaduje řešení úloh s nároky na enormní výpočetní výkon a množství dat. Experimentální zkoumání některých problémů je obvykle nákladné či dokonce prakticky nerealizovatelné. Mezi takové složité úlohy patří například simulace a modelování v oblasti astronomie a astrofyziky, fyziky plazmatu, proudění, meteorologie (předpověď počasí), vojenského průmyslu (simulace výbuchu atomových zbraní) nebo biotechnologií a genetiky.

Současně se technologický vývoj počítačových procesorů pomalu dostává do stádia, kdy naráží na fyzikální omezení v oblasti miniaturizace a zvyšování hodinového taktu. Tato omezení jsou dána zejména materiálovými limity křemíkových VLSI technologií. Pokud narážíme na fyzikální limity jednotlivých počítačových procesorů, nabízí se jako přirozené řešení využití společné práce více procesorů – místo sekvenčního zpracování problému tedy nechat procesory pracovat na jednom problému společně, čili paralelně. Paralelismus je v současnosti téměř standardně využíván v nových osobních počítačích a postupně se rozšiřuje i do přenosných zařízení, jako jsou například tablety, mobilní telefony a televizory. Paralelismus je využíván i ve službách, které každodenně používáme – např. společnost Google využívá pro běh svých aplikací tisíce standardních, vzájemně propojených počítačů. Příkladem praktické implementace paralelismu jsou rovněž elektronické hry, ať už pro PC či pro konzole, jako jsou Microsoft Xbox nebo Sony Playstation. Paralelní výpočty jsou také nativně podporovány v současných verzích výpočetních systémů typu MATLAB nebo Mathematica.

Jak to vypadá?

Jak vůbec takový paralelní počítač vypadá? V případě osobních počítačů rozdíl mezi jedno- a vícejádrovým procesorem na první pohled prakticky nepoznáme, neboť mají podobný rozměr a jsou zapouzdřeny. V případě vysoce výkonných a drahých počítačů (tzv. superpočítačů, které jsou dnes všechny bez výjimky založeny na paralelismu) si odlišnosti od běžného počítače jistě všimneme, neboť se jedná o poměrně objemná zařízení, která mohou zabírat celé místnosti.

Vývoj superpočítačů se stal prestižní záležitostí jak pro firmy, tak pro akademickou a vědeckou obec. Ve vývoji se tradičně předhání firmy jako Fujitsu, Cray, AMD, Intel a IBM. Seznam 500 nejvýkonnějších paralelních počítačů na světě je umístěn na stránkách [3] a je pravidelně aktualizován.

Jak měřit rychlost počítačů?

Tradičním měřítkem výkonnosti počítačů je jednotka *flops* (FLoating-point Operations Per Second), která představuje počet provedených operací v plovoucí řádové čárce za sekundu. V souvislosti se zmíněnými jednotkami se používají předpony soustavy SI, konkrétně jde v současné době o předponu giga (10^9) u osobních počítačů a peta (10^{15}) u superpočítačů. V tab. 1 je uveden seznam pěti nejvýkonnějších superpočítačů současnosti, včetně počtu výpočetních jednotek – jader (PU) – a maximálního počtu operací za sekundu (měřeno v petaflops).

Pořadí	Název	Specifikace	Výrobce	Země	Počet PU	R_{max} [Pflops]
1	Titan	Cray XK7 Opteron 6274 16C 2.200 GHz	Cray	USA	560 640	17 590 000
2	Sequoia	BlueGene/Q Power BQC 16 1.60 GHz	IBM	USA	1 572 864	16 324 751
3	K computer	K computer SPARC64 VIIIfx 2.0 GHz	Fujitsu	Japonsko	705 024	10 510 000
4	Mira	BlueGene/Q Power BQC 16C 1.60 GHz	IBM	USA	786 432	8 162 376
5	JUQUEEN	BlueGene/Q Power BQC 16C 1.600 GHz	IBM	Německo	393 216	4 141 180

Tab. 1: Seznam pěti nejvýkonnějších superpočítačů (k listopadu 2012)

Všimněme si, že rychlost nejvýkonnějšího počítače je již možné udávat v zettaflopsech, tj. řádově 10^{21} operací za sekundu.

Možnosti urychlení výpočtů

Kdo by si nepamatoval na úlohy o společné práci, kterými nás „trápili" učitelé na základní škole. Typické zadání znělo například tak, že dělník František vykoná práci za 8 hodin, dělník Josef za 10 hodin, a úkolem bylo určit, za jak dlouhou dobu vykonají práci společně. Nejednoho žáka asi napadlo, jak by taková spolupráce v praxi vypadala. Jakým způsobem by si dělníci práci rozvrhli? Nepřekáželi by si navzájem? Je opravdu možné, aby celou dobu pracovali současně? Myšlenka o společné práci přirozeně inspirovala i počítačové architekty a analytiky. Pokud chceme nechat pracovat více procesorů na jednom problému současně, nabízí se otázka, jakého zrychlení je vlastně možné docílit. Předem lze uvést, že ideálním případem je tzv. *lineární zrychlení*, které značí, že pokud na daném problému necháme pracovat N procesorů, pak čas nutný k vykonání práce se zmenší N-krát. Ačkoliv se to může zdát překvapivé, pokud neuvažujeme anomálie, lepšího zrychlení dosáhnout nelze. S lineárním zrychlením se navíc v praxi téměř nesetkáme.

Amdahlův a Gustafsonův zákon

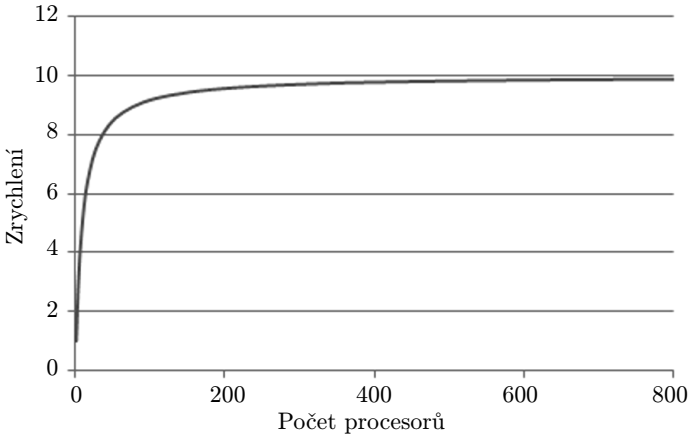
Jak tedy probíhá paralelní práce více procesorů na jednom problému? Můžeme si ji představit jako společnou práci více identických dělníků. Problém je rozdělen na několik nezávislých částí, takže každý procesor může vykonávat práci současně s ostatními. Zrychlení společné práce komplikuje *přirozeně sekvenční část* problému, kterou není možné vykonávat paralelně. Tím se dostáváme k tzv. *Amdahlovu zákonu*, který udává zrychlení počítače po jeho „vylepšení". Ve své nejobecnější formě má tvar

$$S = \frac{\text{doba výpočtu před zlepšením}}{\text{doba výpočtu po zlepšení}},$$

kde S je zrychlení počítače po jeho zlepšení. Nyní předpokládejme, že α označuje část problému, kterou je možné paralelizovat, $(1-\alpha)$ představuje část problému, kterou paralelizovat nelze (čili se jedná o přirozeně sekvenční část problému) a P je počet procesorových jader použitých k výpočtu. Pak maximální zrychlení $S(P)$, kterého je možné dosáhnout, lze určit pomocí varianty Amdahlova zákonu pro paralelní zrychlení:

$$S(P) = \frac{1}{\frac{1-\alpha}{P} + \alpha}$$

Máme-li tedy 8 procesorů a je možné paralelizovat 90 % výpočtu, pak maximální zrychlení je pouze 3,2. Budeme-li zvyšovat počet procesorů „nade všechny meze“, bude se zrychlení limitně blížit desetinásobnému. Průběh zmíněného zrychlení v závislosti na počtu procesorů je znázorněn na obr. 1.

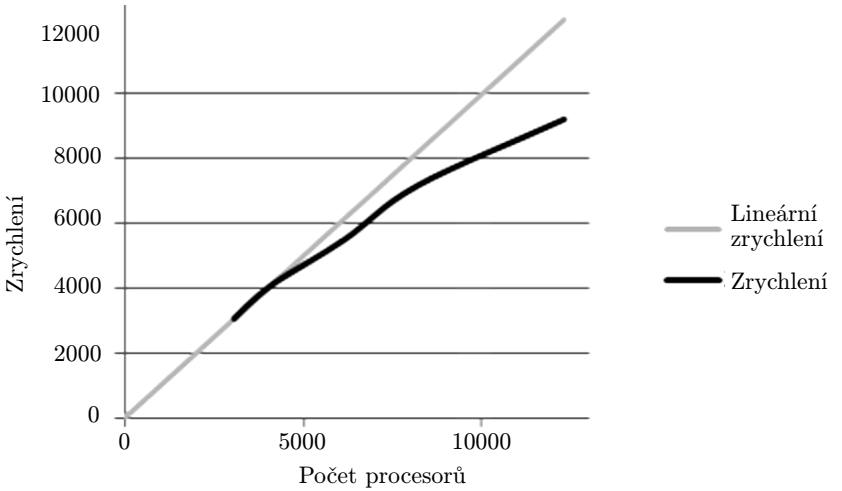


Obr. 1: Amdahlův zákon – omezení dané 10% sekvenční části výpočtu

Popsaný Amdahlův zákon platí pouze za předpokladu, že se se změnou počtu procesorů nemění velikost úlohy (neboli velikost zpracovávaných dat). Pro návrháře paralelních systémů je ale výhodnější, pokud se s rostoucím počtem procesorů provádí výpočty na větším objemu dat. V tom případě ale roste paralelní část problému lineárně s počtem procesorů. Toto rozšíření Amdahlova zákona, známé jako *Gustafsonův zákon*, je možné popsat následovně. Předpokládejme, že P označuje počet procesorů a α představuje část problému, kterou není možné paralelizovat. Pak maximální zrychlení, kterého je možné docílit, lze vypočítat podle vztahu:

$$S(P) = P - \alpha(P - 1)$$

Situace je znázorněna na obr. 2, kde je rovněž vyznačen rekordman v rámci evropského systému předpovědi počasí Integrated Forecast System (IFS), počítač HECToR (Cray XE6).



Obr. 2: Porovnání teoretického a nejlepšího dosaženého paralelního zrychlení na počítači HECToR (Cray XE6) v předpovědi počasí [1]

Zrychlení rovněž komplikuje synchronizace výpočetních jednotek a jejich vzájemná komunikace. V případě, kdy komunikační režie způsobuje časové ztráty, které převyšují čas nutný pro řešení problému, můžeme dokonce hovořit o tzv. paralelním zpomalení. To ostatně platí i pro „lidské dělníky“ – máme-li malý problém a pracuje na něm současně neúměrné množství lidí, jejich práce zřejmě nebude příliš efektivní.

Vybrané typy propojovacích sítí

Propojovací síť paralelního počítače je způsob vzájemného propojení jednotlivých procesorů, který může být formalizován a popsán pomocí prostředků známých z teorie grafů. Graf je, zjednodušeně řečeno, množina bodů – tzv. *uzlů*, které jsou navzájem propojeny spojnicemi – *hranami*.

Procesory mohou být reprezentovány uzly grafu a komunikační linky mezi nimi hranami. Díky tomu pak můžeme použitím grafů konstruovat rozličné typy sítí, omezené pouze naší fantazií.

Nicméně, aby mohla síť efektivně fungovat, je třeba zohlednit řadu mnohdy protichůdných požadavků:

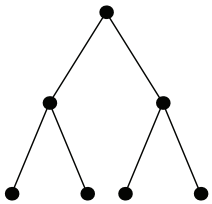
- malý počet sousedů každého uzlu (čím více vzájemných propojení, tím nákladnější bude počítač),

- malý průměr sítě, neboli co nejmenší maximální vzdálenost libovolných dvou uzlů v grafu (to znamená menší počet komunikačních linek, menší počet zpráv a jejich směrování, méně chyb a zahlcení linek),
- co nejmenší počet úzkých míst (neúměrně vytížených linek), aby byla snížena pravděpodobnost zahlcení sítě v případě přerušení důležitých spojů,
- sítě by měly být reprodukovatelné a algebraicky popsatelné.

Uvedeme si zde základní typy sítí, které se používají při návrhu paralelních počítačů. Jsou to úplný binární strom, hyperkrychle a mřížka.

Úplný binární strom

Binární strom je struktura definovaná nad konečným počtem uzlů, která se skládá ze tří částí: kořen, levý podstrom a pravý podstrom. Kořen je tedy uzel, který má pod sebou nejvýše dva následníky (potomky). Tato jednoduchá definice umožňuje rekurzivně strom rozšiřovat – pokud považujeme levý a pravý podstrom za nový strom, dostaneme se na další úroveň „globálního“ stromu a tím rozšiřujeme jeho výšku. Strom je zajímavý tím, že mezi dvěma uzly existuje jen jediná cesta. Úplný binární strom výšky N má přesně 2^{N+1} uzlů a hran. Příklad úplného binárního stromu je znázorněn na obr. 3.

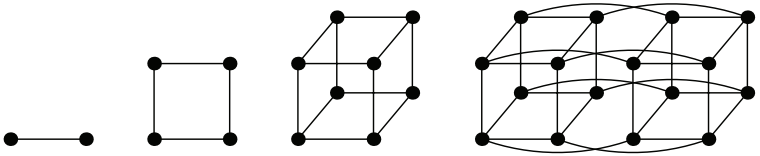


Obr. 3: Úplný binární strom výšky 2

Hyperkrychle

Hyperkrychle (n-cube) patří k nejjednodušším síťovým strukturám, n-rozměrná hyperkrychle se obvykle značí $Q(n)$, kde n je počet jejich dimenzí. Třírozměrnou hyperkrychli, tedy $Q(3)$, zná každý z vlastní zkušenosti například při hře v kostky. Dvourozměrná hyperkrychle je prostý čtverec a jednorozměrná krychle jsou dva spojené uzly – úsečka. Zkusme si nyní představit, jak dospějeme od úsečky ke čtverci a od čtverce ke krychli. Rozšíření hyperkrychle o nový rozměr provedeme tak, že „zkopí-

rujeme“ výchozí graf a hranami spojíme nové uzly se svými původními vzory. Tímto rekurzivním způsobem lze dospět k hyperkrychli o libovolném počtu dimenzí. Na obr. 4 jsou znázorněny hyperkrychle dimenze 1, 2, 3 a 4.

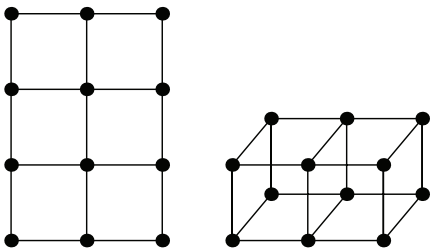


Obr. 4: Hyperkrychle dimenze (a) 1, (b) 2, (c) 3, (d) 4

Hyperkrychli dimenze n je také možné si představit jako binární posloupnost (sekvenci jedniček a nul) délky n. Tato posloupnost – např. 0001 – vyjadřuje souřadnice jednoho konkrétního uzlu. Kolik je celkový počet permutací v binární posloupnosti? Přesně 2^n, tedy tolik uzlů, kolik má hyperkrychle. Mezi dvěma uzly vede hrana právě tehdy, pokud se liší právě v jednom bitu. Každý uzel má tedy n sousedů, protože je možné znegovat n bitů v dané posloupnosti.

Mřížka

Mřížka, podobně jako hyperkrychle, je poměrně snadno představitelnou sítí. Dvojrozměrná mřížka je síť, která je dobře známá z běžného života, třeba z rybolovu. Jednorozměrná mřížka je prostá řada procesorů spojená hranami. Na obr. 5 je znázorněn příklad dvourozměrné a třírozměrné mřížky.



Obr. 5: Příklad dvourozměrné a třírozměrné mřížky

Zatímco u hyperkrychle byla souřadnice uzlu vyjádřena binární posloupností, u n-rozměrné mřížky je možné ji vyjádřit posloupností kladných čísel, přičemž délka posloupnosti je n – příkladem souřadnice jednoho uzlu v třírozměrné mřížce je posloupnost (1, 2, 3). Podobně jako u hyperkrychle vede mezi dvěma uzly hrana právě tehdy, když se v právě jedné souřadnici liší o jedničku. Jak je vidět z obrázku, jednoduchá změna v definici způsobila významnou změnu ve struktuře.

Výčet sítí samozřejmě zdaleka není úplný. Existuje řada různých topologií, jako jsou *toroid*, tzv. *motýlek* (*butterfly*) nebo *graf hvězda* (*star graph*), nicméně jejich popis by vydal na samotnou knihu.

Ze zmíněných příkladů je důležitý zejména poznatek, že paralelní systémy se dají popsat grafem, který splňuje určité vlastnosti a zákonitosti, a od toho se následně odvíjí návrh paralelních algoritmů přizpůsobených konkrétní architektuře. Je tedy jasné, že návrh takových algoritmů je komplikovanější než v případě sekvenčního počítače.

Nyní máme popsán paralelní počítač a zkusíme se podívat, jakým způsobem na něm efektivně provádět výpočty. Situace evidentně není tak jednoduchá, že na problém „pustíme“ více procesorů a problém se sám vyřeší rychleji. Je tedy nutné přistoupit k novému způsobu návrhu algoritmů – takovému, který bere zároveň v úvahu podstatu problému a architekturu paralelního počítače, pro který je určen.

Součet N čísel paralelně aneb paralelní redukce

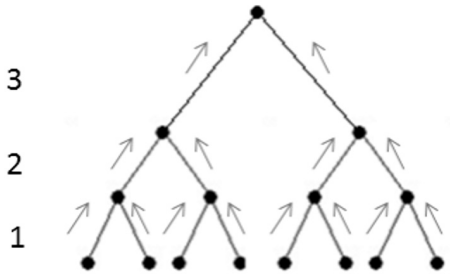
Často používaným způsobem paralelního řešení úloh je tzv. paralelní redukce. Triviální úlohou, na které je tento přístup ilustrován, je součet N čísel uložených v paměti. Předpokládejme, že máme skupinu N procesorů, které mají sečíst N čísel. Pokud bychom postupovali sekvenčně a práci by vykonával pouze jeden procesor, bylo by potřeba všechna čísla, která mají být sečtena, načíst jedno po druhém z paměti (N kroků) a tato čísla jedno po druhém sečíst (dalších N kroků). Celkově bychom tedy museli vykonat $2N$ kroků. Pokud by tento procesor měl na začátku jedno číslo ve své paměti, bylo by o jeden krok méně, což je v globálním měřítku zanedbatelné zlepšení.

Paralelismus ale nabízí rychlejší řešení. Předpokládejme, že v počátečním kroku mají již všechny procesory své číslo ve své paměti, a zbývá tedy pouze všechna čísla sečíst dohromady.

Očíslujme nyní jednotlivé procesory v rostoucím pořadí od jedné do N. V prvním kroku procesory s lichým pořadím čísla od svých sudých sou-

sedů a sečtou tato dvě čísla dohromady. To se provede v jediném paralelním kroku a po jeho provedení obsahuje polovina procesorů součet dvou čísel. Nyní můžeme tento krok opakovat: každý z těch procesorů, které obsahují součet dvou čísel, očíslujeme v rostoucím pořadí od jedné do $N/2$. Procesory s lichým pořadím obdrží od svých sudých sousedů čísla, která jsou součtem dvou čísel z předchozího kroku, a tato čísla sečtou. Tím pádem redukovaná skupina procesorů o velikosti $N/4$ obsahuje součet čtyř čísel. Postup analogickým způsobem opakujeme, dokud jeden procesor nebude obsahovat součet všech čísel.

Můžeme si všimnout, že počet kroků se oproti sekvenčnímu řešení významně snížil. Porovnejme například zrychlení pro $N = 1\,024$ čísel. Při sekvenčním řešení vykonáme 2 048 kroků, zatímco při paralelním řešení pouze 20 kroků! To je přibližně 50násobné zrychlení. Problémem, který je však třeba řešit, je skutečnost, že potřebujeme 1 024 procesorů a většina z nich začne po určitém počtu kroků „zahálet“, ačkoliv by mohly vykonávat jinou užitečnou práci. Tento problém se řeší tzv. vyvažováním zátěže, kterým se však zde zabývat nebudeme.



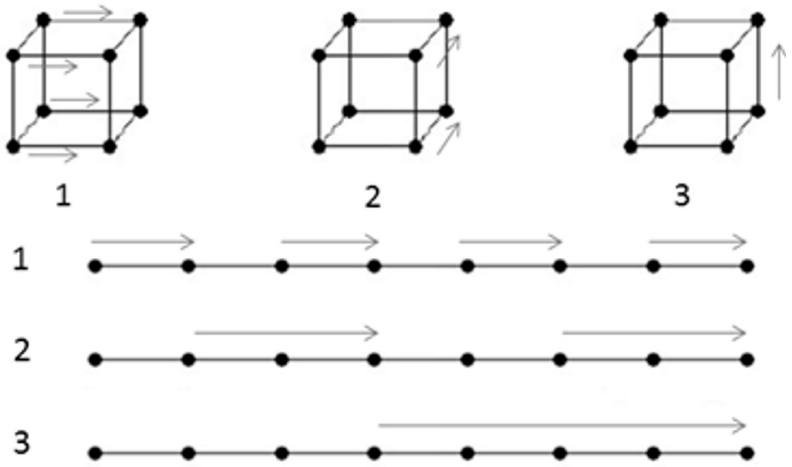
Obr. 6: Paralelní redukce na úplném binárním stromu

Algoritmus součtu N čísel může být elegantně proveden na v článku prezentovaných topologiích. Připomeňme si, že v prvním kroku mají všechny procesory své číslo ve své paměti.

Úplný binární strom je pro tento typ algoritmů ideální strukturou, a to proto, že jak algoritmus, tak strom mají rekurzivní charakter. Jak jsme již zmínili, binární strom je definovaný rekurzivně – je to struktura definovaná nad konečným počtem uzlů, která se skládá ze tří částí: kořen, levý podstrom a pravý podstrom. Oba podstromy jsou pak buď prázdné, nebo tvoří nový strom.

Průběh součtu 8 čísel na úplném binárním stromu výšky 3 je možné ilustrovat pomocí obr. 6. V prvním kroku pošle 8 procesorů v dolní části obrázku svá čísla čtyřem procesorům do vyššího patra, které sečtou dohromady po dvou číslech. V druhém kroku pošlou tyto čtyři procesory své součty dvěma procesorům do ještě vyššího patra, které opět provedou součet dvou čísel. V posledním kroku sečte jediný procesor – kořen stromu – dvě čísla dohromady, čímž dostaneme požadovaný celkový součet. Pro součet 8 čísel jsme tedy potřebovali 6 kroků – rozeslání, součet, rozeslání, součet, rozeslání, součet. V sekvenčním případě bychom vykonali 16 kroků.

Stejný algoritmus je možné provést i na hyperkrychli a mřížce, což je ilustrováno na obr. 7. Princip je úplně stejný.



Obr. 7: Paralelní redukce na hyperkrychli a jednorozměrné mřížce

Pro ukázku funkce algoritmu optimalizovaného pro hyperkrychli si zvolíme hyperkrychli dimenze 3, tedy klasickou třírozměrnou kostku, jelikož je velmi dobře představitelná. Stejně jako v předchozím příkladu budeme sčítat 8 čísel. V prvním kroku pošlou čtyři procesory v levé části krychle svá čísla svým čtyřem sousedům vpravo. Tyto procesory následně sečtou svá čísla s těmi, co právě obdržely. V dalším kroku se situace rekurzivně opakuje v pravé části krychle – dva procesory vepředu pošlou své částečné součty dvěma procesorům vzadu, které však už své částečné součty také mají. Provedou tedy pouze další součet. V posledním kroku

pošle dolní procesor svůj součet hornímu procesoru, který provede poslední součet, čímž dokončí celou operaci. I v tomto případě jsme tedy vykonali pouze 6 kroků – tři rozeslání čísel a tři součty.

Podobně elegantně umí algoritmus řešit i jednorozměrná mřížka, což je v podstatě řada procesorů propojených linkami. V prvním kroku pošlou liché procesory svá čísla sudým sousedům, ty provedou částečný součet. V druhém kroku pošle polovina těchto procesorů své součty sousedům vzdáleným o dvojnásobek předchozího kroku a tak dále. Není překvapením, že i zde bylo nutné vykonat pouze šest paralelních kroků.

Shrnutí

Paralelní počítače jako jeden ze způsobů urychlení vědeckotechnických výpočtů mají do budoucna významný potenciál. V tomto článku jsme představili obecné principy paralelních počítačů a ukázali jednoduchý algoritmus součtu N čísel na N výpočetních jednotkách. Paralelismus v tomto případě poskytuje efektivní zrychlení oproti sekvenčnímu řešení, nicméně ani v tomto jednoduchém případě se nepodařilo dosáhnout lineárního zrychlení, tedy N-násobného zrychlení oproti sekvenčnímu případu. To je jeden z nejdůležitějších poznatků, které by měly plynout z tohoto článku – lineární zrychlení je pro většinu byť jednoduchých úloh těžko dosažitelným ideálem. S vývojem paralelních počítačů je nutné řešit otázky adaptace rostoucího počtu procesorů na řešené problémy, vyvažovat výpočetní zátěž a brát ohled na sekvenční části výpočtu a komunikační režii. Nicméně je téměř jisté, že v budoucnu budeme využívat stále více paralelismu, což bude klást nové nároky na znalosti informatiků a inženýrů.

Literatura

[1] Mozdzynski, G., Hamrud, M., Wedi, N., Doleschal, J., Richardson, H.: A PGAS Implementation by Co-design of the ECMWF Integrated Forecasting System (IFS). In: *SC Companion: High Performance Computing, Networking Storage and Analysis*, 2012, s. 652–661.

[2] Tvrdík, P.: *Paralelní systémy a algoritmy.* Nakladatelství ČVUT, Praha, 2006.

[3] *TOP500.org, 2013*: `http://www.top500.org/`