Paralelné programovanie
Analytické modelovanie

Bc. št. prog. Informatika - 2010/2011

Ing. Michal Čerňanský, PhD.

Fakulta informatiky a
informačných technológií,
STU Bratislava

Prehľad tém

- Réžia v paralelných programoch
- Výkonnostné miery paralelných programov
- Účinok granularity na výkonnosť
- Škálovateľnosť paralelných systémov
- Minimálny čas vykonania a minimálny cenovo optimálny čas vykonania
- Asymptotická analýza paralelných programov
- Iné metriky škálovateľnosti

Analytické modelovanie - úvod

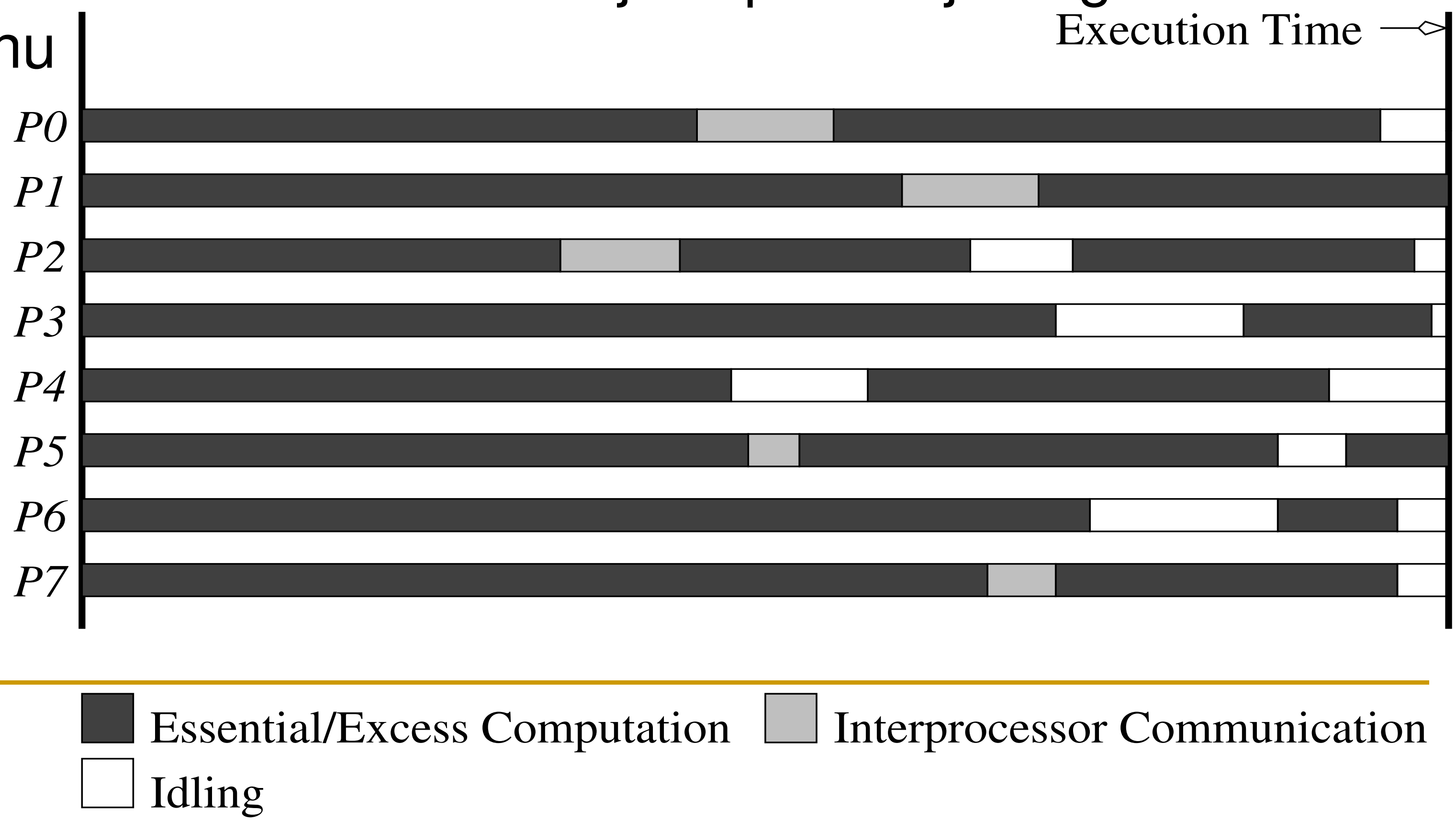
- Sekvenčný algoritmus – ohodnotený na základe času jeho behu (vo všeobecnosti asymptoticky v závislosti na veľkosti vstupu)
- Asymptotický čas behu sekvenčného programu je identický na každej sekvnčnej platforme
- Čas behu paralelného programu závisí od veľkosti vstupu, počtu procesorov a parametrov komunikácie počítačového systému
- Algoritmus musí byť analyzovaný s ohľadom na platformu, nad ktorou bude bežať
- Paralelný systém – kombinácia paralelného algoritmu a platformy

Analytické modelovanie - úvod

- Viaceré výkonnostné metriky sú inutitívne
- Čas behu programu (Wall Clock Time) – čas od spustenia prvého procesora až po čas zastavenia posledného procesora v paralelnom počítačovom systéme
- Ako sa mení, ak sa zmení počet procesorov?
- Ako veľmi je rýchlejšia paralelná verzia?
- Voči ktorej základnej sekvenčnej verzií algoritmu porovnávať?
- Načo FLOPs keď nerieši problém?

Réžia v paralelných programoch

- Dva procesory – nemal by program bežať 2x rýchlejšie?
- Viaceré zdroje réžie – nadbytočné výpočty, komunikácia, nečinnosť a obsadenie zdrojov spôsobujú degradáciu výkonu



Réžia v paralelných programoch

- Interakcie medzi procesmi
  - Netriviálny paralelný problém – potreba komunikácie
- Nečinnosť (Idling)
  - Procesy môžu byť nečinné z dôvody nevyrovnanej záťaže, synchronizácie alebo sekvenčnej časti v probléme
- Nadbytočné výpočty
  - Výpočty, ktoré nie je potrebné realizovať v sekvenčnej verzií algoritmu
  - Sekvenčná verzia algoritmu je ťažko paralelizovateľná
  - Niektoré výpočty sú realizované na viacerých procesoroch, aby sa minimalizovala komunikácia

Výkonnostné miery paralelných programov – čas behu programu

- Čas behu sekvenčného programu – čas uplynutý od spustenia po ukončenie sekvenčného počítač. systému
- Čas behu paralelného programu - čas uplynutý od spustenia vykonávania prvého procesora paralelného počítač. systému po zastavenie vykonávania posledného procesora
- Čas behu sekvenčného programu – T_s
- Čas behu paralelného programu – T_p

Výkonnostné miery paralelných programov – celková paralelná réžia

- Nech T_{all} je celkový čas, ktorý strávili všetky procesory pri riešení úlohy
- T_s je čas behu sekvenčnej verzie problému
- $T_{all} – T_s$ - celkový čas strávený procesormi realizujúc nadbytočnú prácu – celková paralelná réžia
- Celkový čas, ktorý strávili všetky procesory pri riešení úlohy $T_{all} = p\, T_p$ (p – počet procesorov)
- Celková paralelná réžia (T_o) je daná

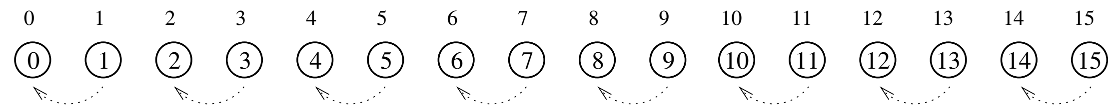
$$T_o = p\, T_p - T_s$$

Výkonnostné miery paralelných programov – zrýchlenie

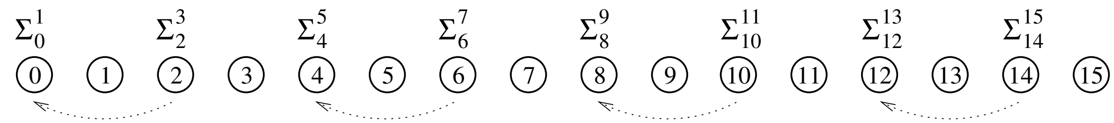
- Aký je úžitok z paralelného riešenia problému?
- Zrýchlenie (***S***) je pomer času riešenia problému na sekvenčnom procesore k času riešenia problému na paralelnom systéme s ***p*** identickými procesormi

Výkonnostné miery paralelných programov – príklad

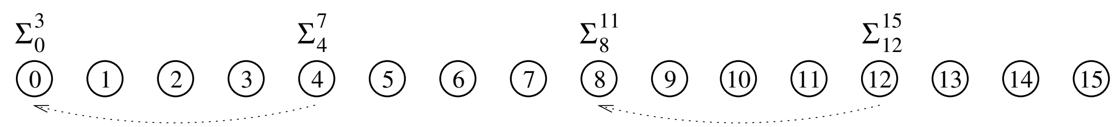
- Problém sčítania n čísel s využitím n procesorov
- Ak ***n*** je mocnina 2, môžeme vykonať operáciu v ***log n*** krokoch propagovaním čiastkových súčtov v logickom binárnom strome procesorov



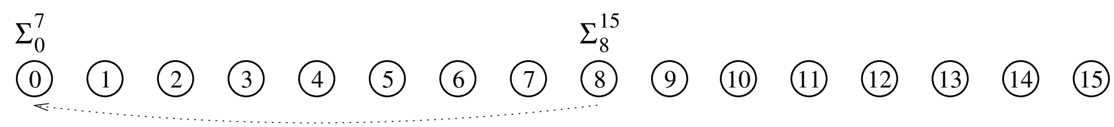
(a) Initial data distribution and the first communication step



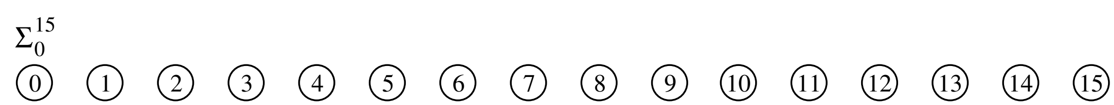
(b) Second communication step



(c) Third communication step



(d) Fourth communication step



(e) Accumulation of the sum at processing element 0 after the final communicatior

Výkonnostné miery paralelných programov – príklad

- Ak operácia súčtu trvá konštantný čas t_c a komunikácia jedného slova trvá $t_s + t_w$, máme paralelný čas vykonania $T_P = \Theta(\log n)$
- Vieme, že $T_S = \Theta(n)$
- Zrýchlenie S je dané $S = \Theta(n / \log n)$

Výkonnostné miery paralelných programov – zrýchlenie

- Pre daný problém môže byť dostupných veľa sekvenčných algoritmov
- Tieto algoritmy sa líšia v asymptotických časoch behu programu
- Pre výpočet zrýchlenia – vždy ten najlepší sekvenčný program

Výkonnostné miery paralelných programov – príklad výpočtu zrýchlenia

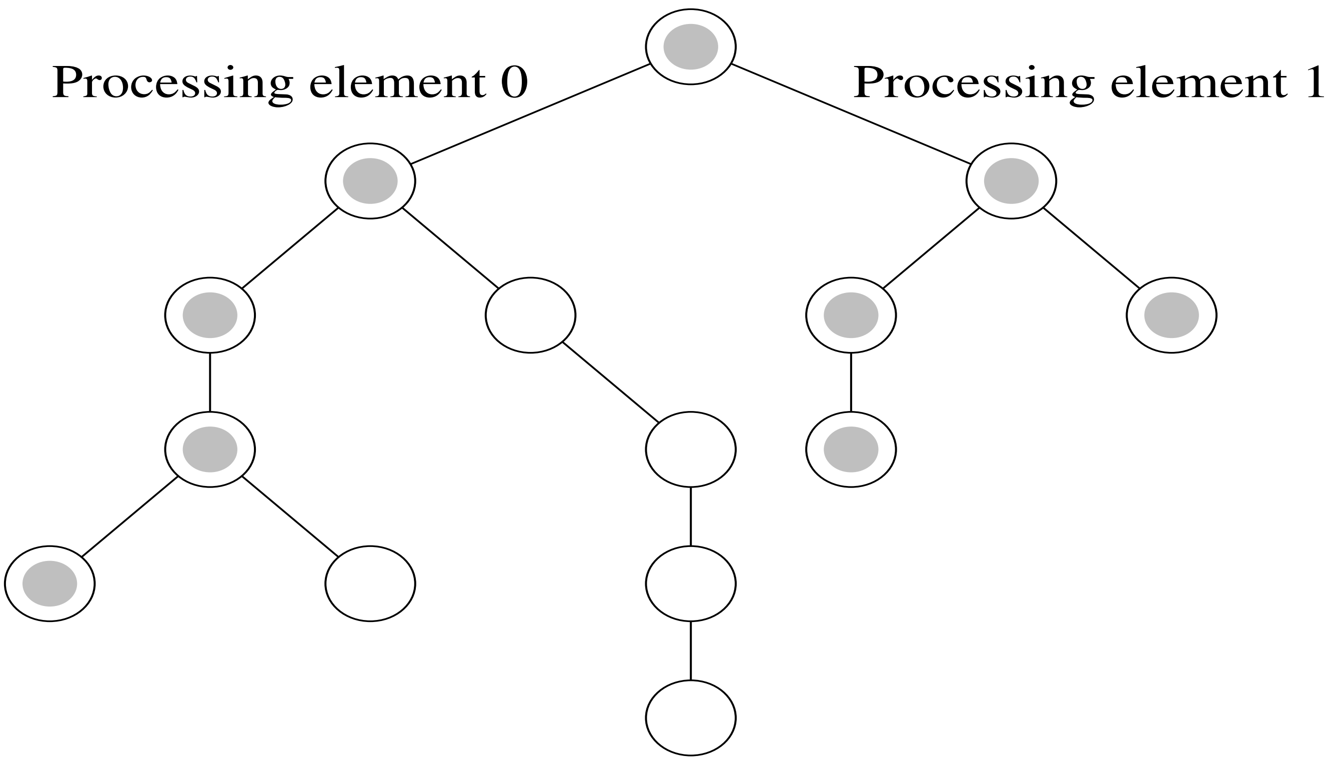
- Paralelný bubble sort
- Čas vykonania bubblesort-u sekvenčnom počítači 150 s
- Čas vykonania odd-even sort-u (efektívna implementácia bubble sort-u) je 40 s
- Zrýchlenie sa javí ako 150/40 = 3.75
- Je to férové posúdenie systému?
- Čo ak by quicksort na sekvenčnom systéme trval iba 30 s ?
- V takom prípade je zrýchlenie 30/40 = 0.75, čo je realistickejšie posúdenie systému

Výkonnostné miery paralelných programov – ohraničenia zrýchlenia

- Zrýchlenie môže byť aj nulové (paralelný program nikdy neskončí)
- Teoreticky môže byť zrýchlenie zhora ohraničené počtom procesorov ***p***, môžeme očakávať ***p***-násobné zrýchlenie ak použijeme ***p*** x viac zdrojov
- Zrýchlenie je možné, iba ak každý procesor strávi výpočtom menej ako T_S / p času pri riešení problému
- V takej situácií ale jediný procesor môže byť využitý (zdieľanie času) na vytvorenie rýchlejšieho sekvenčného programu, čo nie je v súlade s predpokladom, že najrýchlejší sekvenčný alg. bol použitý pre urč.zrýchlenia

Výkonnostné miery paralelných programov – superlineárne zrýchlenia

- Jeden z dôvodov superlineárneho zrýchlenia – paralelná verzia vykoná menej práce ako sekvenčná verzia



Výkonnostné miery paralelných programov – superlineárne zrýchlenia

- Superlineárne zrýchlenie na základe zdrojov
- Vyššia pamäťová priepustnosť paralelného systému - lepšie cache-hit ratio

- Procesor má 64kB vyrovnávacej pamäte – 80% hit ratio
- Ak sú použité 2 procesory, hit ratio je 90% (menší problém na jeden procesor), 10% zvyšných prístupov sa delí na 8% lokálna a 2% vzdialená pamäť
- Čas prístupu do DRAM je 100ns, 2 ns čas prístupu do cache pamäte a čas prístupu do vzdialenej pamäte je 400ns – zrýchlenie 2.43

Výkonnostné miery paralelných programov - efektvita

- Efektivita vyjadruje mieru užitočného využitia procesora

- Matematicky je daná

$$E = \frac{S}{p}.$$

- Podobne ako ohraničenie pri zrýchlení, efektivita môže byť nulová až jednotková

Výkonnostné miery paralelných programov - efektvita

- Zrýchlenie paralelného sčítavania čísel je daná:

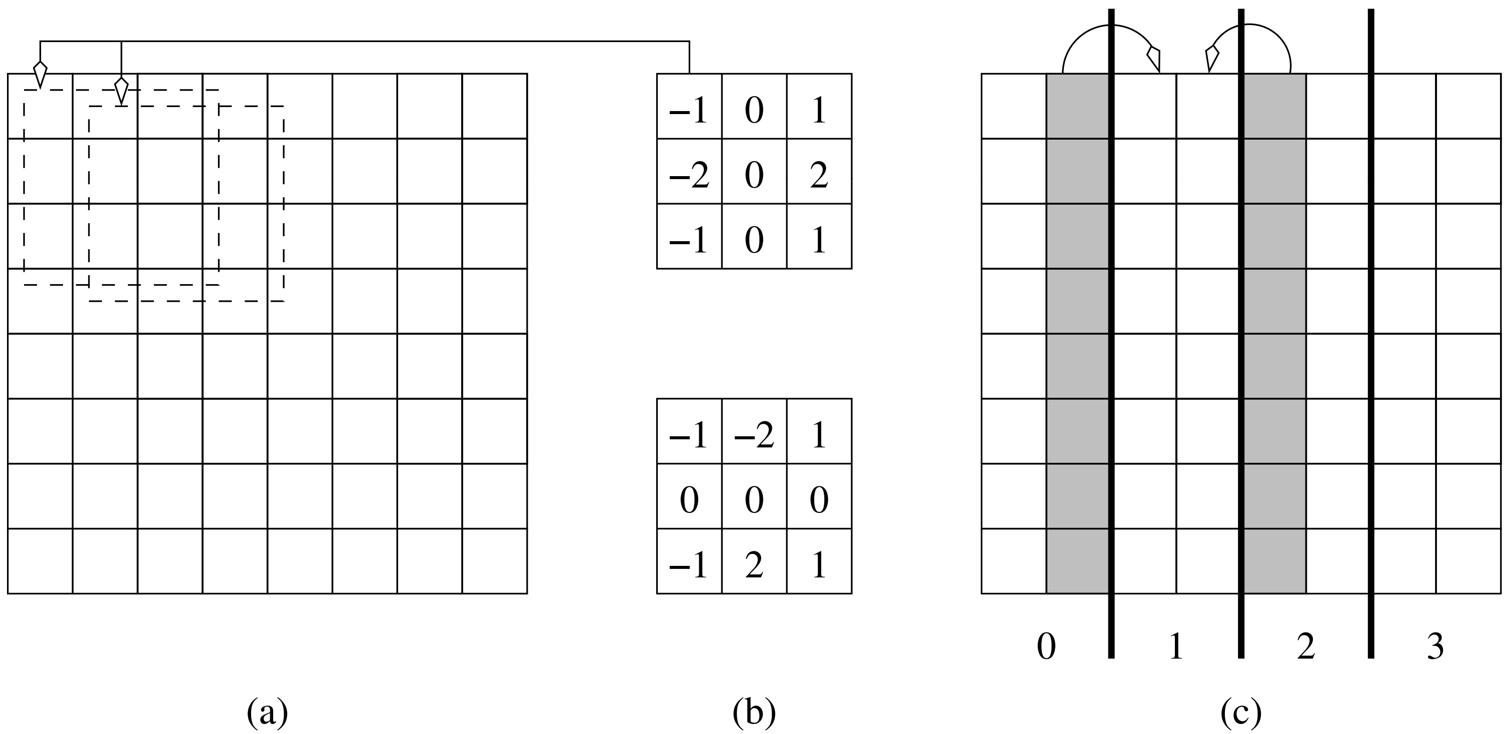
$$S = \frac{n}{\log n}$$

- Efektivita je daná:

$$E = \frac{\Theta\left(\frac{n}{\log n}\right)}{n} = \Theta\left(\frac{1}{\log n}\right)$$

Čas paralelného spracovania, zrýchlenie a efektivita - príklad

- Detekcia hrán, kernel 3x3 na každý pixel.
- Čas sekvenčného sprac. obrazu $\boldsymbol{n}$ x $\boldsymbol{n}$ je $\boldsymbol{T_S = 9\ t_c\ n^2}$



Čas paralelného spracovania, zrýchlenie a efektivita - príklad

- Možnosť paralelnej realizácie – rozdelenie obrazu na rovnako veľké vertikálne segmenty, každý s n^2 / p bodmi

- Okrajové časti každého segmentu majú veľkosť $2n$ bodov, ktoré musia byť komunikované, čo trvá $2(t_s + t_w n)$

- Kernel môže byť potom aplikovaný na všetkých n^2 / p bodov v čase $T_S = 9\, t_c n^2 / p$

Čas paralelného spracovania, zrýchlenie a efektivita - príklad

- Celkový čas behu algoritmu je teda:

$$T_P = 9t_c\frac{n^2}{p} + 2(t_s + t_w n)$$

- Zodpovedajúce hodnoty zrýchlenia a efektivity sú:

$$S = \frac{9t_c n^2}{9t_c\frac{n^2}{p} + 2(t_s + t_w n)}$$

- a

$$E = \frac{1}{1 + \frac{2p(t_s+t_w n)}{9t_c n^2}}.$$

Cena paralelného systému

- Cena je súčin času behu paralelného programu a počtu procesorov ($\boldsymbol{p}$ x $\boldsymbol{T_P}$)
- Cena odzrkadľuje súčet časov, ktoré každý procesor strávil riešením problému
- Paralelný systém je cenovo optimálny, ak cena riešenia problému na paralelnom počítačovom systéme je asymptoticky rovná cene sériového riešenia
- Kedže $\boldsymbol{E} = \boldsymbol{T_S} / \boldsymbol{p}\ \boldsymbol{T_P}$ pre cenovo optimálny systém $\boldsymbol{E} = O(1)$.

Cena paralelného systému - príklad

- Problém sčítavania čísel na paralelnom počítačovom systéme
- Čas behu paralelného programu: $T_P = \log n$ (pre $p = n$)
- Cena systému je daná: $p\, T_P = n \log n$
- Čas behu sekvenčného algoritmu je $\Theta(n)$, paralelná verzia nie je cenovo optimálna

Dôsledok cenovej neoptimálnosti

- Algoritmus usporadúvania, používa ***n*** procesorov na usporiadanie zoznamu v čase $(\mathbf{log}\ n)^2$.
- Čas behu sekvenčného programu je $n\ \mathbf{log}\ n$, zrýchlenie je $n\ /\ \mathbf{log}\ n$ a efektivita algoritmu je $1\ /\ \mathbf{log}\ n$
- Súčin $p\ T_P$ tohto algoritmu je $n\ (\mathbf{log}\ n)^2$.
- Tento algoritmus nie je cenovo optimálny, ale iba malým faktorom ***log n***.
- Ak $p < n$, priradenie ***n*** úloh ***p*** procesorom vedie k $T_P = n\ (\mathbf{log}\ n)^2\ /\ p$.
- Zodpovedajúce zrýchlenie je $p\ /\ \mathbf{log}\ n$.
- Zrýchlenie sa znižuje so zvyšujúcou sa veľkosťou problému pre fixné ***p***

Vlyv granularity na výkonnosť

- Menej procesorov často vedie k lepšej výkonnosti paralelného systemu
- Využite menšieho než maximálneho možného počtu procesorov na vykonanie paralelného programu - škálovanie paralelného systému nadol
- Naivná realizácia škálovania nadol – originálny procesor je virtuálny a viaceré sú namapované na procesor v systéme naškálovaného nadol
- Počet procesorov klesá s mierou ***n*** / ***p*,** množstvo práce pre každý procesor stúpa ***n*** / ***p*** krát
- Komunikácia by nemala stúpať rovnakou mierou – niektoré s virtuálnych procesorov namapované na rovnaký fyzický procesor môžu navzájom komunikovať

Vytváranie granularity - príklad

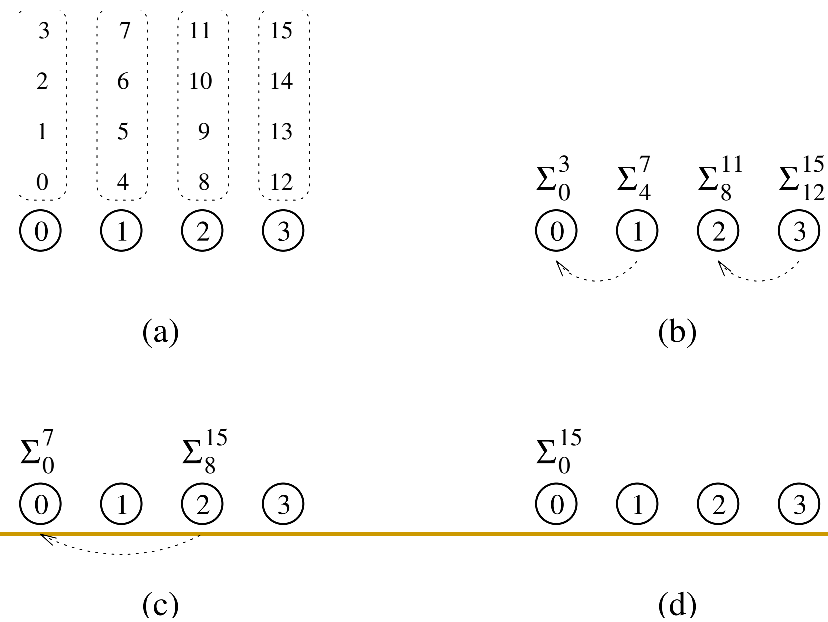
- Uvažujme problém spočítania n čísel p procesormi, pričom platí $p < n$ a n aj p sú mocniny 2
- Použime paralelnú verziu algoritmu s n procesormi, teraz ale virtuálne procesory
- Každému z p procesorov je teraz priradených n / p virtuálnych procesorov
- Prvých $\log p$ z $\log n$ krokov pôvodného algoritmu je teraz simulovaných v $(n / p) \log p$ krokoch p procesorov
- Nasledujúce $\log n - \log p$ krokov nevyžaduje komunikáciu

Vytváranie granularity - príklad

- Celkový čas vykonania paralelného algoritmu na takomto systéme je $\Theta\left((\boldsymbol{n} / \boldsymbol{p}) \mathbf{\log} \boldsymbol{p}\right)$
- Cena je $\Theta(\boldsymbol{n} \mathbf{\log} \boldsymbol{p})$, čo je asymptoticky viac ako cena $\Theta(\boldsymbol{n})$ spočítania ***n*** čísel sekvenčne. Paralelný systém nie je cenovo optimálny

Vytváranie granularity - príklad

- Môžeme vytvoriť granularitu takým spôsobom, že získame cenovo optimálny systém?
- Každý procesor lokálne spočíta ***n*** / ***p*** čase Θ (***n*** / ***p***)
- ***p*** čiastkových súčtov na ***p*** procesoroch môže byť spočítaných v čase ***log p***



Vytváranie granularity - príklad

- Čas behu paralelného programu je

$$T_P = \Theta(n/p + \log p),$$

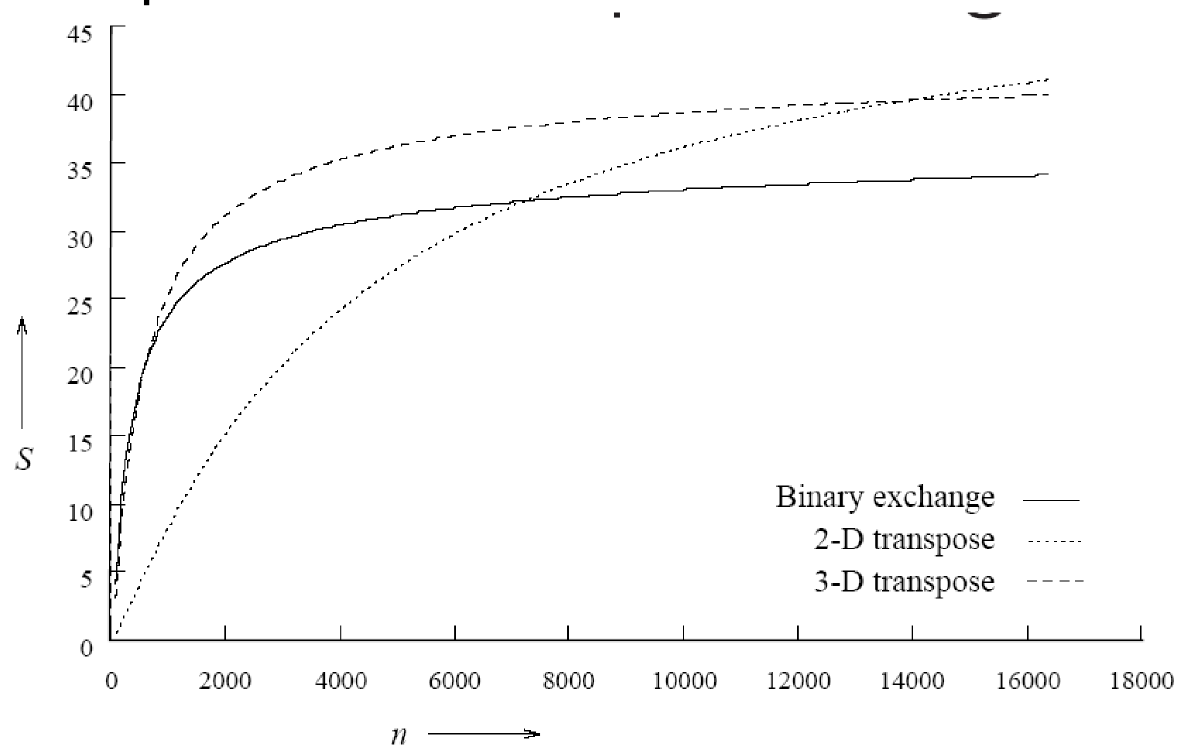
- Cena je $\Theta(n + p \log p)$

- Pokiaľ $n = \Omega(p \log p)$

- Je paralelný systém cenovo optimálny

Škálovateľnosť paralelných systémov

- Ako extrapolovať výkon malých problémov a malých systémov ma väčšie problémy na väčších konfiguráciach?
- FFT na 64 procesoroch



Vlastnosti škálovateľnosti paralelných systémov

- Efektivita paralelného programu môže byť zapísaná ako

$$E = \frac{S}{p} = \frac{T_S}{pT_P}$$

- Alebo

$$E = \frac{1}{1 + \frac{T_o}{T_S}}.$$

- Funkcia celkovej réžie T_o je rastúca s p

Vlastnosti škálovateľnosti paralelných systémov

- Pre danú veľkosť problému (hodnota $\boldsymbol{T_S}$ ostáva konštantná), so zvyšujúcim sa počtom procesorov sa zvyšuje aj celková réžia $\boldsymbol{T_o}$

- Celková efektivita paralelného systému klesá
- V každom paralelnom programe

Vlastnosti škálovateľnosti paralelných systémov - príklad

- Uvažujem problém spočítania ***n*** čísel na ***p*** procesoroch

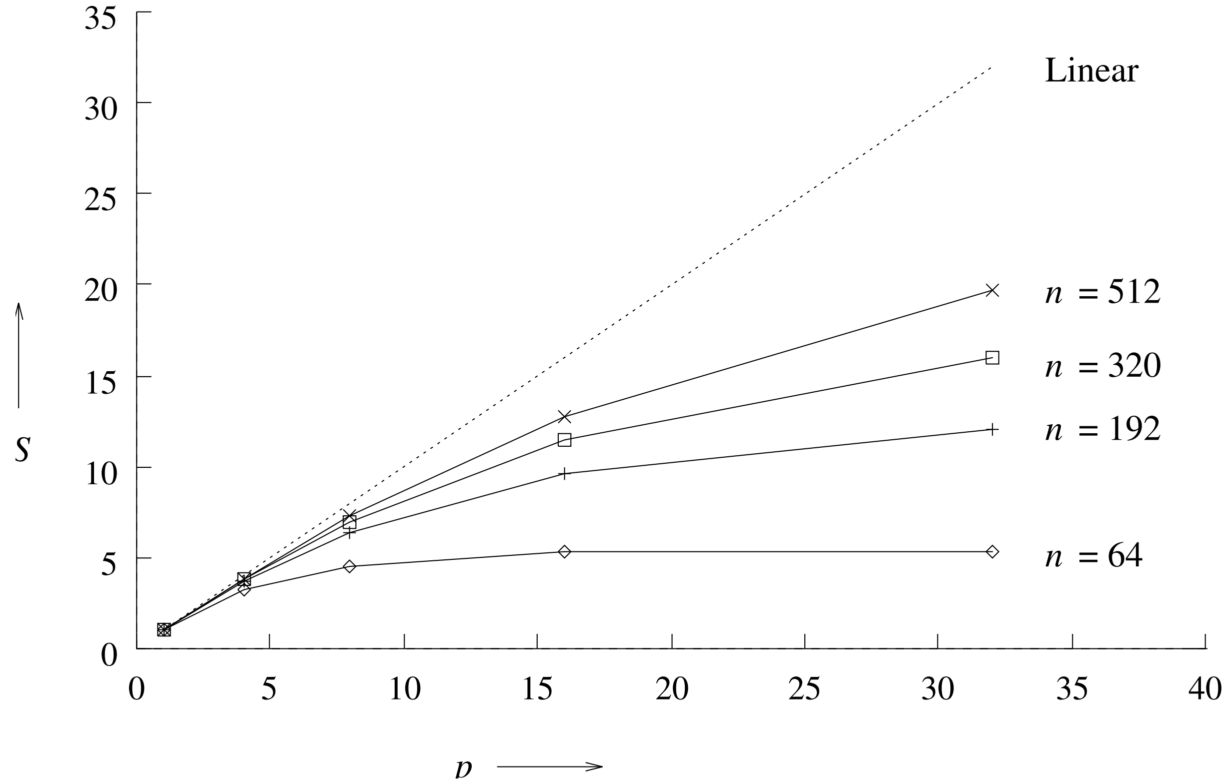
$$T_P = \frac{n}{p} + 2\log p$$

$$S = \frac{n}{\frac{n}{p} + 2\log p}$$

$$E = \frac{1}{1 + \frac{2p\log p}{n}}$$

Vlastnosti škálovateľnosti paralelných systémov - príklad

- Zrýchlenie pre rôzne veľkosti problému – saturácia zrýchlenia a pokles efektivity podľa Amdahlovho zákona



Vlastnosti škálovateľnosti paralelných systémov

- Funkcia celkovej réžie T_o je funkciou aj veľkosti problému T_s aj počtu procesorov p
- V mnohých prípadoch T_o rastie sublineárne vzhľadom na T_s.
- V takýchto prípadoch sa efektivita zvyšuje ak sa zväčšuje veľkosť problému, pričom počet procesorov ostáva konštantný
- Pre takéto systémy môžeme súčasne zväčšiť veľkosť problému a zvýšiť počet procesorov, aby sme zachovali efektivitu konštantnú
- Takéto systémy označujeme ako škálovateľné paralelné systémy

Vlastnosti škálovateľnosti paralelných systémov

- Cenovo optimálne paralelné systémy majú efektivitu $\Theta(1)$.
- Škálovateľnosť a cenová optimálnosť sú previazané
- Škálovateľný systém môže byť vždy cenovo optimálny, ak počet procesorov a veľkosť problému sú vhodne zvolené

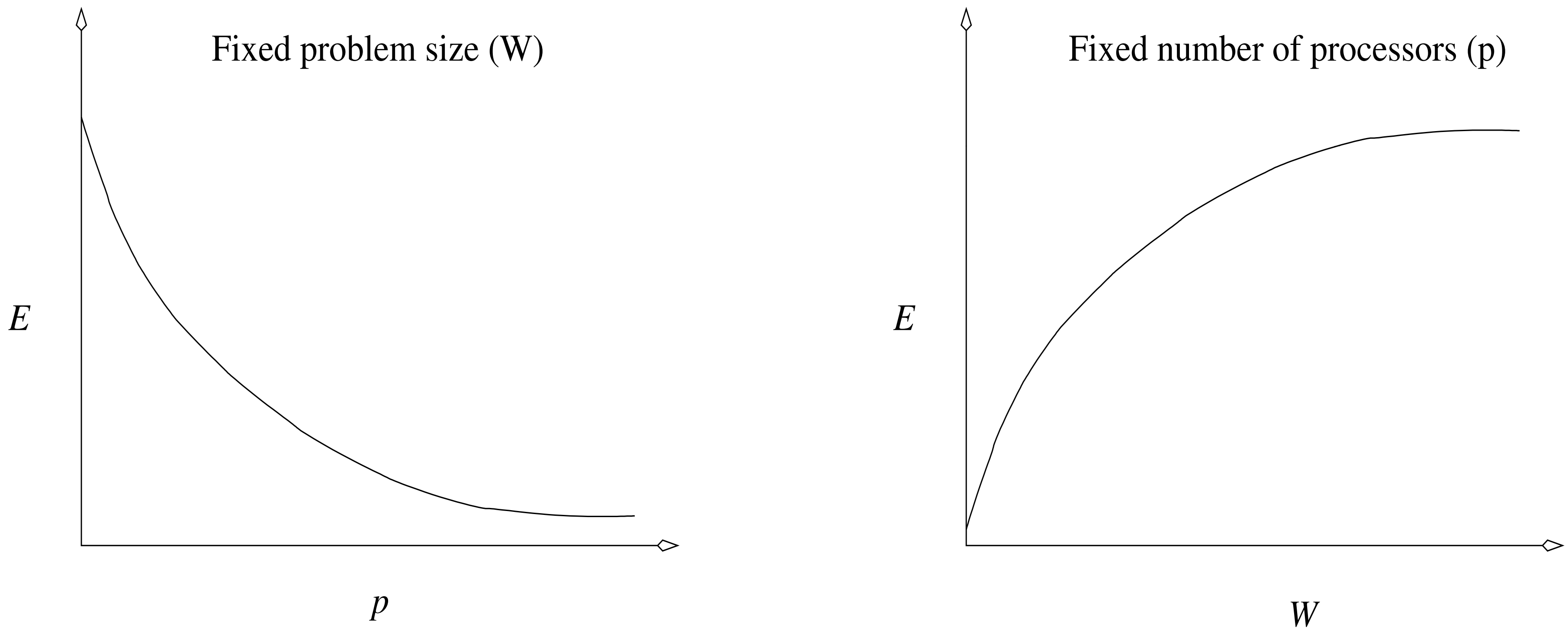
Izoefektivita – metrika škálovateľnosti

- Pre problém danej veľkosti, zvyšovanie počtu procesorov spôsobuje pokles efektivity paralelného systému

- Pre niektoré problémy sa efektivita paralelného systému zvyšuje, ak veľkosť problému rastie a počet procesorov ostáva konštantný

Izoefektivita – metrika škálovateľnosti

- Zmena efektivity vzhľadom na počet procesorov a veľkosť problému



(a) (b)

Izoefektivita – metrika škálovateľnosti

- Aká je miera, s ktorou musí rásť veľkosť problému vzhľadom na počet procesorov, aby zostávala efektivita konštantná?

- Táto miera určuje škálovateľnosť systému
  - Čím menší faktor, tým lepšia škálovateľnosť

- Pred formalizáciou tejto miery definujeme veľkosť problému W ako asymptotický počet operácií v najlepšom sekvenčnom algoritme riešiaceho problém

Izoefektivita – metrika škálovateľnosti

- Čas behu paralelného programu môžeme zapísať ako:

$$T_P = \frac{W + T_o(W,p)}{p}$$

- Výsledný vzťah pre zrýchlenie je:

$$\begin{aligned} S &- \frac{W}{T_P} \\ &= \frac{Wp}{W + T_o(W,p)}. \end{aligned}$$

- Nakoniec môžeme zapísať vzťah pre efektivitu:

$$\begin{aligned} E &= \frac{S}{p} \\ &= \frac{W}{W + T_o(W,p)} \\ &= \frac{1}{1 - T_o(W,p)/W}. \end{aligned}$$

Izoefektivita – metrika škálovateľnosti

- Pre škálovateľné systémy môže byť efektivita udržiavaná na konštantnej hodnote (medzi 0 a 1) ak je podiel T_o / W udržiavaný na konštantnej hodnote
- Koľko práce vzhľadom na želanú úroveň efektivity:

$$
\begin{aligned}
E &= \frac{1}{1 + T_o(W,p)/W}, \\
\frac{T_o(W,p)}{W} &= \frac{1-E}{E}, \\
W &= \frac{E}{1-E} T_o(W,p).
\end{aligned}
$$

- Ak $K = E / (1 - E)$ je konštanta vyjadrujúca efektivitu, ktorú chceme udržať konštantnú, keďže T_o zavisí od W a p, dostávame

$$W = K T_o(W,p).$$

Izoefektivita – metrika škálovateľnosti

- Závislosť veľkosti problému ***W*** môže byť vyjadrený po vykonaní algebraických manipulácií ako funkcia ***p*** ak má byť efektivita konštantná

- Táto funkcia sa označuje ako izoefektivita (isoefficiency function*)*

- Táto funkcia určuje, ako ľahko je možné v paralelnom systéme udržať efektivitu konštantnú a tým dosiahnuť zrýchlenie úmerné počtu procesorov

Izoefektivita – príklad

- Funkcia celkovej réžie pre problém sčítania ***n*** čísel na ***p*** procesoroch je približne ***2p log p***
- Po dosadení ***2p log p*** za T_o získavame

$$W = K2p\log p.$$

- Teda asymptotická funkcia izoefektivity pre tento paralelný systém je $\Theta(p\log p)$
- Ak je počet procesorov zvýšený z ***p*** na ***p'***, veľkosť problému (teraz ***n***) musí byť zvýšená o faktor (***p' log p'***) / (***p log p***) aby efektivita ostala rovnaká ako v systéme s ***p*** procesormi

Izoefektivita – príklad

- Zložitejší príklad: $T_o = p^{3/2} + p^{3/4}W^{3/4}$
- Použijúc iba prvý člen $\boldsymbol{T_o}$ z rovnice

$$W = Kp^{3/2}. \tag{14}$$

- Použijúc iba druhý člen rovnice ***W*** and ***p***:

$$\begin{aligned} W &= Kp^{3/4}W^{3/4} \\ W^{1/4} &= Kp^{3/4} \\ W &= K^4p^3 \end{aligned}$$

- Väčší z týchto asymptotických pomerov určuje izoefektivitu: $\Theta(\boldsymbol{p^3})$

Cenová optimálnosť a izoefektivita

- Paralelný systém je cenovo optimálny vtedy a len vtedy ak:

$$pT_P = \Theta(W).$$

- A teda:

$$\begin{aligned} W + T_o(W,p) &= \Theta(W) \\ T_o(W,p) &= O(W) \\ W &= \Omega(T_o(W,p)) \end{aligned}$$

- Ak máme izoefektivitu $\boldsymbol{f}(\boldsymbol{p})$, vzťah $\boldsymbol{W} = \Omega(\boldsymbol{f}(\boldsymbol{p}))$ musí byť splnený na zabezpčenie cenovej optimálnosti pri škálovaní systému

Dolné ohraničenie izoefektivity

- Pre problém veľkosti ***W*** jednotiek práce nie viac ako ***W*** procesorov môže byť využitých cenovo optimálne

- Veľkosť problému musí rásť aspoň tak rýchlo ako $\Theta(\boldsymbol{p})$ aby bola zabezpečená konštantná efektivita; čize $\Omega(\boldsymbol{p})$ je asymptotické dolné ohraničenie funkcie izoefektivity

Stupeň súbežnosti a izoefektivita

- Maximálny počet úloh, ktoré môžu byť vykonávané súbežne sa nazýva stupeň súbežnosti

- Ak ***C***(***W***) je stupeň súbežnosti paralelného programu, potom pre problém veľkosti ***W***, nie viac ako ***C***(***W***) procesorov môže byť využitých efektívne

Stupeň súbežnosti a izoefektivita: príklad

- Riešenie problému systému ***n*** rovníc o ***n*** neznámy Gaussovou elimináciou
- ***n*** premenných musí byť eliminovaných jedna po druhej a eliminácia jednej premennej trvá $\Theta(\boldsymbol{n}^2)$ výpočtov
- Najviac $\Theta(\boldsymbol{n}^2)$ procesorov môže byť využitých súbežne
- Nakoľko $\boldsymbol{W} = \Theta(\boldsymbol{n}^3)$ pre tento problém, stupeň súbežnosti $\boldsymbol{C}(\boldsymbol{W})$ je $\Theta(\boldsymbol{W}^{2/3})$.
- Ak máme ***p*** procesorov, veľkosť problému musí byť aspoň $\Omega(\boldsymbol{p}^{3/2})$ aby boli všetky využité

Minimálny a cenovo optimálny minimálny čas vykonania

- Minimálny čas vykonania paralelného algoritmu

- Minimálny čas paralelného behu $\boldsymbol{T_P}^{min}$ pre dané $\boldsymbol{W}$ derivovaním $\boldsymbol{T_P}$ podľa $\boldsymbol{p}$ a hľadaním riešenia pre výsledok 0

$$\frac{\mathrm{d}}{\mathrm{d}p} T_P = 0$$

- Ak $\boldsymbol{p}_0$ je hodnota $\boldsymbol{p}$ určená uvedenou rovnicou, $\boldsymbol{T_P}(\boldsymbol{p}_0)$ je minimálny čas vykonania paralelného algoritmu

Minimálny čas vykonania - príklad

- Minimálny čas vykonania pre spočítavanie ***n*** čísel:

$$T_P = \frac{n}{p} + 2\log p.$$

- Po položení derivácie podľa ***p*** rovnej 0 dostaneme ***p = n/ 2***, Zodpovedajúci čas behu je:

$$T_P^{min} = 2\log n.$$

- Nie je cenovo optimálne

Minimálny cenovo optimálny čas paralelného vykonania

- Nech $\boldsymbol{T_P}^{cost\_opt}$ je minimálny cenovo optimálny čas vykonania
- Ak funkcia izoefektivity je $\Theta(\boldsymbol{f(p)})$, potom problém veľkosti $\boldsymbol{W}$ môže byť vyriešený cenovo optimálne vtedy a len vtedy ak $\boldsymbol{W} = \Omega(\boldsymbol{f}(\boldsymbol{p}))$
- Čiže pre cenovú optimálnosť musí platiť $\boldsymbol{p} = O(\boldsymbol{f}^{-1}(\boldsymbol{W}))$
- Pre cenovo optimálne systémy, $\boldsymbol{T_P} = \Theta(\boldsymbol{W/p})$, a teda

$$T_P^{cost\_opt} = \Omega\left(\frac{W}{f^{-1}(W)}\right).$$

Minimálny cenovo optimálny čas paralelného vykonania: príklad

- Uvažujme problém spočítania ***n*** čísel
- Funkcia izoefektivity ***f***(***p***) tohto paralelného systému je Θ(***p* log *p***)
- Z toho vyplýva ***p*** ≈ ***n*** / **log *n***
- Pri tomto počte procesorov, čas behu paralelného programu:

$$\begin{aligned} T_P^{cost\_opt} &= \log n + \log\left(\frac{n}{\log n}\right) \\ &= 2\log n - \log\log n. \end{aligned}$$

- Oba časy T_P^{min} a $T_P^{cost\_opt}$ pre paralelné spočítanie ***n*** čísel sú Θ(**log *n***)

Asymptotická analýza paralelných programov

- Usporadúvanie ***n*** čísel, najrýchlejší sekvenčný algoritmus – čas Θ(***n* log *n***)

Algorithm	A1	A2	A3	A4
p	n^2	$\log n$	n	$\sqrt{n}$
T_P	1	n	$\sqrt{n}$	$\sqrt{n}\log n$
S	$n\log n$	$\log n$	$\sqrt{n}\log n$	$\sqrt{n}$
E	$\frac{\log n}{n}$	1	$\frac{\log n}{\sqrt{n}}$	1
pT_P	n^2	$n\log n$	$n^{1.5}$	$n\log n$

Asymptotická analýza paralelných programov

- Rýchlosť ($\boldsymbol{T_P}$): A1 a potom A3, A4 a A2
- Efektivita: A2 a A4, potom A3 a nakoniec A1
- Cena A2 a A4 sú cenovo optimálne, A1 a A3 nie sú
- Potreba definovať ciele analýzy a podľa toho použiť patričnú mieru

Iné metriky škálovateľnosti

- Mnoho iných metrík, navrhnuté podľa špecifických porieb aplikácií
- Pre aplikácie reálneho času – cieľ je aby systém vykonal úlohu v stanovenom čase
- Ak je obmedzená pamäť, metriky určujú výkonnosť aj so zohľadnením nárastu pamäte

Zdroje

- Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar. Introduction to ParallelComputing, 2nd Edition, Addison-Wesley 2003„Introduction to Parallel Computing“ http://www-users.cs.umn.edu/~karypis/parbook/

- Obrázky prevzaté z:
  - Ananth Grama, Anshul Gupta, George Karypis, Vipin Kumar. Introduction to ParallelComputing, 2nd Edition, Addison-Wesley 2003„Introduction to Parallel Computing“ http://www-users.cs.umn.edu/~karypis/parbook/