Sam Farré Baltà (1605576)

Disseny d'Algorismes Paral·lels

(1) ***Paral·lelització de la infini-norma d'una matriu.***

> 1. Es calculen paral·lelament els valors absoluts de cada element de la matriu, sent n^2 processos paral·lels, cada un d'ordre $O(1)$.
> 2. Per cada fila, paral·lelament, es calcula el sumatori dels seus elements. Podem aplicar un algorisme de reducció, resultant en n processos d'ordre $O(\log n)$.
> 3. Quan ja han acabat els sumatoris, podem aplicar un algorisme de reducció per calcular paral·lelament el màxim dels resultats anteriors, resultant en una tasca d'ordre $O(\log n)$.
>
> Deduïm que, paral·lelament, la infini-norma d'una matriu es pot resoldre amb $O(1 + 2\log n) = O(\log n^2)$.
>
> En sèrie, l'ordre de la tasca seria $O(n^2 + n^2 + n) = O(n^2)$.
>
> Finalment, el speed-up és:
>
> $$S(n^2) = \frac{O(\log n^2)}{O(n^2)}$$

Índexs de Qualitat

(2) ***Llei d'Amdahl.***

> En aquest cas, el speed-up és:
>
> $$S(n) = \frac{T(1)}{T(n)} = \frac{T(1)}{0.001 \cdot T(1) + 0.999 \cdot \frac{T(1)}{n}} = \frac{1}{0.001 + \frac{0.999}{n}}$$
>
> Per tant, per cada n, tenim els speed-ups totals:
>
> $$\begin{aligned} S(30) &= \frac{1}{0.001 + \frac{0.999}{30}} &= 29.16 \\ S(30000) &= \frac{1}{0.001 + \frac{0.999}{30000}} &= 967.77 \\ S(3000000) &= \frac{1}{0.001 + \frac{0.999}{3000000}} &= 999.68 \end{aligned}$$

(3) ***Llei de Gustafson: per què tants cores?***

> Perquè és especialment útil per problemes més grans i complexos. A mesura que la complexitat del problema augmenta, també ho sol fer la fracció del programa que es pot paral·lelitzar.

(4) ***Fracció sèrie màxima a partir del speed-up.***

> Segons la llei d'Amdahl:
>
> $$S(n) = \frac{1}{f + \frac{1-f}{n}} \implies 20 = \frac{1}{f + \frac{1-f}{32}} \implies f = 0.019$$
>
> Segons la llei de Gustafson:
>
> $$S(n) = f + (1-f) \cdot n \implies 20 = f + (1-f) \cdot 32 \implies f = 0.387$$

(5) ***Speed-up i eficiència.***

> Siguin n la mida del problema i c el nombre de processadors, el temps mínim d'execució del codi és:
>
> $$T(n,c) = 100 + n + \frac{n^2}{2c}$$
>
> Per tant, el speed-up és:
>
> $$S(n,c) = \frac{T(n,1)}{T(n,c)} = \frac{100 + n + \frac{n^2}{2}}{100 + n + \frac{n^2}{2c}}$$
>
> La màxima acceleració possible és doncs:
>
> $$\lim_{c \to \infty} S(1000, c) = \lim_{c \to \infty} \frac{100 + 1000 + \frac{1000^2}{2}}{100 + 1000 + \frac{1000^2}{2c}} = \frac{501100}{1100} = 455.54$$
>
> I l'eficiència per diferents nombres de processadors és:
>
> $$\begin{aligned} E(1000, 50) &= \frac{S(1000,50)}{50} = \frac{100 + 1000 + \frac{1000^2}{2}}{50\left(100 + 1000 + \frac{1000^2}{100}\right)} &= 0.903 \\ E(500, 25) &= \frac{S(500,25)}{25} = \frac{100 + 500 + \frac{500^2}{2}}{25\left(100 + 500 + \frac{500^2}{50}\right)} &= 0.897 \end{aligned}$$

(6) ***Escalabilitat.***

> Veiem que quan hem escalat la mida del problema i el nombre de processadors en un factor de 2, la seva eficiència s'ha mantingut molt propera a 0.9, indicant una escalabilitat dèbil.