Universiteit van Amsterdam

Een blik op het *Round-Robin*-algoritme en diens verbeterde varianten

25-05-2018

Tutor:
W.R.J. Vermeulen

Practicumgroep:
A1 - ALGOL

Cursus:
Besturingssystemen

Student:
R.A.J. Wacanno
11741163

1 Introductie

2 De *Round-Robin* met aanpassingen

2.1 Het *Round-Robin*-algoritme

Voordat verbeteringen van het *Round-Robin*-algoritme (*RR*) kunnen worden besproken, dient eerst kennis genomen te worden van de werking van *RR* in zijn simpele vorm. Een uitgebreide beschrijving hiervoor was te vinden in het boek *Operating System Concepts* (Peterson & Silberschatz, 1985).

RR vervult zijn taak als *CPU-schediling*-algoritme door gebruik te maken van een uitvoeringsrij. In deze rij zet RR alle processen die op dat moment uitgevoerd moeten worden. De volgorde van deze processen is in dit geval niet van belang. Vervolgens wordt steeds het proces dat vooraan in de rij staat hieruit gehaald zodat deze voor een bepaalde tijd van de *CPU* gebruik kan maken. Hoeveel tijd een proces per keer krijgt wordt aangeduid als het tijdskwantum; hier soms afgekort als T. Dit tijdskwantum wordt, in deze standaardimplementatie van de *Round-Robin* vooraf bepaald en vastgelegd in de code. Zodra een proces voor de duur van T van de *CPU* gebruik heeft gemaakt wordt deze weer achteraan in de uitvoeringsrij gezet. Een voorbeeld hiervan is met de processen `1` tot en met `5` in figuren 1 en 2 weergegeven.

1-2-3-4-5

Figuur 1: Uitvoeringsrij op tijd 0

2-3-4-5-1

Figuur 2: Uitvoeringsrij op tijd $0+T$

De *Round-Robin* gaat op cyclische wijze door de lijst heen. Mocht een proces binnen een tijdskwantum klaar zijn met uitvoeren, dan zal deze stoppen met gebruikmaken van de *CPU*, maar *Round-Robin* zal tot het einde van het betreffende tijdskwantum wachten tot het een nieuw proces de kans geeft uit te voeren. Met de eerdergenoemde processen `1` tot en met `5`

1	2	3	4	5	1	2	3	4	5	2	3	5	2	5	...

Tabel 1: Voorbeeld van *CPU*-toewijzing per tijdskwantum T gedurende een gegeven periode

2.2 Verbeteringen van de *Round-Robin*

Verbetering bij volgordebepaling
Een allereerste verbeterde implementatie van *Round-Robin* is *SFRR* (*Shortest First Round-Robin*) (Yadav, Mishra, Prakash & Sharma, 2010).

Verbetering bij tijdskwantumbepaling met een bekende procestijd
Veel verbeteringsalgoritmes van de *RR* proberen efficiëntere prestaties te behalen door het tijdskwantum van de *Round-Robin* op een meer dynamische manier te bepalen. Dit betekend dat de grootte van het kwantum af hangt van de processen die in de uitvoeringsrij staan. Voorbeelden van implementaties van dit principe zijn: *RARR* (*Re-Adjusted Round-Robin*) (Behera, Mohanty & Nayak, 2011), *SARR* (*Self-Adjustment Round-Robin*) (Matarneh, 2009) en *SBRR* (*Shortest Burst Round-Robin*) (Hiranwal & Roy, 2011).

In essentie zijn deze varianten gelijkend aan het eerdergenoemde *SFRR*-algoritme. Ook

$$T = \overline{x} = \begin{cases} P_{\frac{n+1}{2}} & \text{als } n \text{ oneven is} \\ P_{\frac{1}{2}}(P_{\frac{n}{2}}) + (Y_{\frac{1+n}{2}}) & \text{als } n \text{ even is} \end{cases}$$

$$T = \begin{cases} \overline{x} & \text{als } \overline{x} \geq 25 \\ 25 & \text{als } \overline{x} < 25 \end{cases}$$

Verbetering bij tijdskwantumbepaling met een onbekende procestijd
(Bakare & Yusuf, 2017)

3 Gevolgen van de verbeteringen

Nu is het natuurlijk de vraag

3.1 Vergelijking met andere algoritmen

(Goel & Garg, 2013)

4 Discussie

4.1 Conclusie

Literatuur

Bakare, O. & Yusuf, N. (2017, jun). Improved dynamic time slice round robin scheduling algorithm with unknown burst time. *International Journal of Computer Applications*, *167*(7), 21–28. doi: 10.5120/ijca2017914305

Behera, H. S., Mohanty, R. & Nayak, D. (2011). A new proposed dynamic quantum with re-adjusted round robin scheduling algorithm and its performance analysis. *arXiv preprint arXiv:1103.3831*. doi: 10.5120/913-1291

Goel, N. & Garg, R. (2013). A comparative study of cpu scheduling algorithms. *arXiv preprint arXiv:1307.4165*.

Hiranwal, S. & Roy, K. (2011). Adaptive round robin scheduling using shortest burst approach based on smart time slice. *International Journal of Computer Science and Communication*, *2*(2), 319–323.

Matarneh, R. J. (2009). Self-adjustment time quantum in round robin algorithm depending on burst time of the now running processes. *American Journal of Applied Sciences*, *6*(10), 1831. doi: 10.3844/ajassp.2009.1831.1837

Peterson, J. L. & Silberschatz, A. (1985). *Operating system concepts*. Addison-Wesley Longman Publishing Co., Inc.

Yadav, R. K., Mishra, A. K., Prakash, N. & Sharma, H. (2010). An improved round robin scheduling algorithm for cpu scheduling. *International Journal on Computer Science and Engineering*, *2*(04), 1064–1066.