Operativsystemer Øving 3 Rapport

Hvordan eksperimentere for å få bedre Round-Robin

For å få bedre Round-Robin kan man endre verdiene som blir puttet ut. En stor forbedring i programmeringskoden vil være å la CPUen begynne på en oppgave med en gang når den blir avbrutt av I/O, men det har vi ikke lov til ifølge oppgaveteksten.

Utskrift fra kjøringen

Her er en utskrift med de gitte parameterene:

```
Please input system parameters:
Memory size (KB): 2048
Maximum uninterrupted cpu time for a process (ms): 500
Average I/O operation time (ms): 225
Simulation length (ms): 250000
Average time between process arrivals (ms): 5000
Simulating.....done.

Simulation statistics:

Number of completed processes:                                  10
Number of created processes:                                    52
Number of (forced) process switches:                            71
Number of processed I/O operations:                             416
Average throughput (processes per second):                      0.04

Total CPU time spent processing:                                113842 ms
Fraction of CPU time spent processing:                          45.5368%
Total CPU time spent waiting:                                   136158 ms
Fraction of CPU time spent waiting:                             54.4632%

Largest occuring memory queue length:                           36
Average memory queue length:                                    15.42472
Largest occuring cpu queue length:                              7
Average cpu queue length:                                       5.014408
Largest occuring I/O queue length:                              1
Average I/O queue length:                                       0.015072
Average # of times a process has been placed in memory queue:   1
Average # of times a process has been placed in cpu queue:      17.0
Average # of times a process has been placed in I/O queue:      8.9

Average time spent in system per process:                       66935 ms
Average time spent waiting for memory per process:              22051 ms
Average time spent waiting for cpu per process:                 37274 ms
Average time spent processing per process:                      5700 ms
Average time spent waiting for I/O per process:                 73 ms
Average time spent in I/O per process:                          1835 ms
```

Som vi ser, så har vi veldig lite utnyttelse av prosessoren, bare 46%. Dette gir dårlig throughput og ellers dårlige stats. Ved å se på når programmet kjører, kan man tydelig se at dette skjer fordi prosessene trenger for kort tid før de trenger I/O, og dermed står CPUen lenge idle. Dette kan man prøve å bøte på ved å ha kortere tidskvant. Hvis man for eksempel prøver med 200ms, blir det slik:

```
Please input system parameters:
Memory size (KB): 2048
Maximum uninterrupted cpu time for a process (ms): 200
Average I/O operation time (ms): 225
Simulation length (ms): 250000
Average time between process arrivals (ms): 5000
Simulating.....done.

Simulation statistics:

Number of completed processes:                                  26
Number of created processes:                                    55
Number of (forced) process switches:                            651
Number of processed I/O operations:                             570
Average throughput (processes per second):                      0.104

Total CPU time spent processing:                                173901 ms
Fraction of CPU time spent processing:                          69.5604%
Total CPU time spent waiting:                                   76099 ms
Fraction of CPU time spent waiting:                             30.4396%

Largest occuring memory queue length:                           23
Average memory queue length:                                    8.658572
Largest occuring cpu queue length:                              10
Average cpu queue length:                                       4.719156
Largest occuring I/O queue length:                              4
Average I/O queue length:                                       0.210444
Average # of times a process has been placed in memory queue:   1
Average # of times a process has been placed in cpu queue:      35.307693
Average # of times a process has been placed in I/O queue:      11.076923

Average time spent in system per process:                       62884 ms
Average time spent waiting for memory per process:              20616 ms
Average time spent waiting for cpu per process:                 32851 ms
Average time spent processing per process:                      5754 ms
Average time spent waiting for I/O per process:                 899 ms
Average time spent in I/O per process:                          2761 ms
```

Her ser vi at vi fikk drastisk bedre stats, bare av å endre tidskvanten.

Hva skjer i køene når vi endrer ting

Dette er en litt uklar oppgave, men jeg regner med oppgaven mener at jeg skal skrive hva som skjer med lengden på de forskjellige køene (minne, CPU og IO) hvis man endrer de forskjellige parametrene (lagerstørrelse, tidskvant og gjennomsnittlig I/O operasjontid).

Dette er på mange måter et ganske komplisert spørsmål. Grunnen til dette, er at alt henger sammen. Det finnes bare et visst antall prosesser i systemet på en gang, og hvis det er flere i en kø, vil det naturlig også være færre i de andre køene. Samtidig kompliserer det enda mer at hvordan disse parametrene er satt sammen endrer på througputen, og kan dermed endre på hvor mange prosesser vi faktisk har inne i systemet. Men la oss gå grundig igjennom en og en parameter, og se hva som skjer hvis vi endrer de.

La oss begynne med lagerstørrelse. Lagerstørrelse begrenser effektivt hvor mange prosesser som kan være «i spill» samtidig, en lav lagerstørrelse vil gjøre at det er færre prosesser som kan være i I/O og CPU. De prosessene som ikke har plass i lageret, legger seg i minnekøen. De må vente helt til en prosess er ferdig, og det den har brukt blir fjernet fra lageret. Da vil det bli plass til den nye prosessen, og den kan da starte å kjøre.

Lagerstørrelsen kan også påvirke IO-køen og CPU-køen. Det vil naturligvis bli færre elementer her ved lavere lagerstørrelse, og det kan til og med skje at throughputen blir lavere siden både IO og CPU virker bedre når det alltid er en prosess i køen, slik at den alltid har noe å jobbe med. Derfor er det lurt at lagerstørrelsen er så stor som mulig, slik at CPUen og IO får jobbe så fort de klarer.

Gjennomsnittlig I/O operasjonstid har også mye å si i dette samspillet. Siden vi ikke har noen påvirkning til hvor lang tid prosessene bruker på CPUen, er det her vi bestemmer om prosessene er CPU-intensive, eller IO-intensive. Hvis I/O-operasjonstiden er kort, vil prosessene være CPU-intensive og det er CPUen som er flaskehalsen, og vi vil se at lengden på CPUkøen blir lang.

Derimot hvis I/O-operasjonstiden er lang, vil det være I/O som er flaskehalsen, og vi kan si at prosessene er I/O-intensive. Da vil vi se IO-køen bli lang, og prosessor-køen ikke være fullt så lang.

Så I/O-operasjonstiden har altså med forholdet mellom CPU-køen og I/O-køen å gjøre. Den gjør ikke noe med minnekøen. Vi husker at minnekøen har alle prosessene som det ikke er plass til i CPU-køen og I/O-køen.

Den siste parameteren er tidskvanten. Dette er hvor langt et hvert prossessorenhet er. Dvs hvor lenge prosessoren jobber av gangen. Hvis tidskvanten er 500, kan det bare komme en ny prosess hvert fem hundrede sekund. Hvis alle prosesser bare skulle jobbet i prosessoren, så hadde ikke dette gjort så mye, men siden prosessene ofte får avbrudd og må jobbe i IO, så vil prosessoren stå mye idle, uutnyttet.

Dette vil gjøre at systemet går tregere, det vil si lavere throughput (ihvertfall hvis vi snakker om CPU-intensive prosesser). Med andre ord vil da CPU-køen bygge seg opp. Naturligvis vil da også minne-køen bygge seg opp, siden det tar lengre tid å få prosesser ut av systemet, og også minnet. I/O-køen derimot, vil minske, siden prosessoren ikke klarer å sende prosesser så ofte dit.

Hva kan man så lære av dette til den virkelige verdenen. I den virkelige verden så ser ting litt anderledes ut. Man kan endre på minnestørrelse, men det er dyrt. Man kan endre på tidskvant, men å skifte prosess koster sykluser. Hvor lang til I/O går kan man prøve å forbedre ved å for eksempel kjøpe større harddisker. Så her har vi et fint eksempel som gjør at man kan lære litt av hvordan samspillet er i den virkelige verden.