A n-core stochastic resource sharing system

Erik Leonards

February 26, 2023

1 The general setup

The model aims at simulating a system in which packages arriving in a queue are handled by n different cores (processing units). On average, part $1-p$ of the package must be done by the core and for part p of the task the core needs to use shared resources.

Keep in mind that the mathematical model that aims at recreating this situation is just an approximation and the model is made such that mathematical analysis is still doable. This may have the consequence that there does not exist a perfect mapping of the model to an application in the real world. The biggest downside of the model in this section will be that next to the cores, the shared resources also have processing capabilities. This is in reality not possible, because the cores are the only component in a computing system that can execute tasks. However, the dynamic of the model and the impact of certain parameter (e.g. number of cores) might still be applicable and give great insights to a situation in the real world.

1.1 The model description

A n core resource sharing system consists of a queue, n-cores and shared resources. In this model it is assumed that both the cores and shared resources have processing capabilities. This might not be the case in real life, but the model might still be useful. The system can handle arriving packages. We will call packages coming from outside as outer-packages and packages coming from the shared-resources as inner-packages. The inter-arrival times of the outer-packages are $E(\lambda)$ (exponentially with parameter λ) distributed. An outer-package consists of N core-tasks and $N-1$ shared-tasks, where N follows a geometric distribution with parameter f. The service requirement of the core-tasks $C_1, C_2, \ldots, C_N$ are $E(\nu_c)$ distributed and are executed in the core. The service requirement of the shared-tasks $S_1, S_2, \ldots, S_{N-1}$ are $E(\nu_s)$

distributed. The order in which the tasks of an outer-package are executed can be seen in Figure 1.

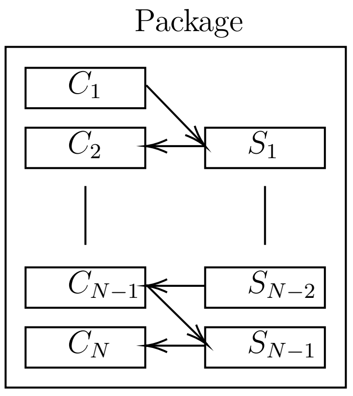

Figure 1: The execution order of the tasks in a package.

Arriving packages are placed in a spot in the queue according to a queueing policy (Figure 2). The queueing policy will be First In First Out (FIFO) unless listed otherwise. The package at the front of the queue is send to the first free core. The first non-executed core-task is executed by the core. Since the number of core-tasks is geometrically distributed, there is a probability of $1 - f$ that the package has another core- and shared-task. The package is send to the shared resources if this is the case. This also means that the package leaves the system with probability f. The package is send back into the queue after the shared-task is executed. Notice that there is no difference between outer- and inner-packages due to the memoryless property of the geometric distribution.

The execution rate of the core-tasks are fixed at 1. This means that the service requirement and service time of core-tasks are equal. This is in contrast to the shared-resources, where the execution rate is dependent on the number of shared-tasks currently being executed. The capacity of the shared resources is equally divided between all shared-tasks currently executed.

1.2 The intuition behind the model

The model tries to model the scenario in which a core receives a task and there is a certain probability (not necessarily equal to $1 - f$) that the task requires the core to access a shared part (for example shared memory). This

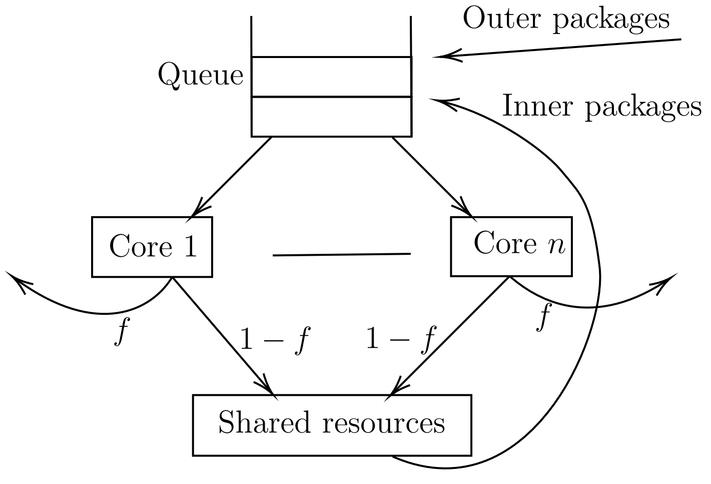

Figure 2: The general setup of a dynamic resource sharing system.

can be seen as the case in which the core wants to process some data saved somewhere in the memory. After the data is retrieved the core wants to do computations with the data and this is the reason why the shared-task always comes paired with a core-task.

1.3 Advantages and disadvantages of the model

There are a few advantages of the model:

- The model is able to capture the fact that not all task require the core to access the shared resources.
- The model incorporates the fact that some processing needs to be done after a task is finished in the shared resources, because there is always a new task in the queue added after one is finished in the shared resources.
- The parameter f allows to precisely set the average number of visits to the shared resources.
- There is a lot of variance in the system. Each core- and shared-task has a different execution requirement and each package arriving in the system makes a different number of visits to the shared resources. Furthermore, the inter-arrival times between the packages is different each time.

The biggest disadvantage is the fact that the shared resources can execute a task without needing the cores.

1.4 Model analysis

The inter-arrival times between the packages are $E(\lambda)$ distributed, which means that the number of newly arrived packages follows the Poisson distribution with parameter λ.

1.4.1 The core-tasks in a package

The total requirement of the core-task of a task is equal to

$$C = \sum_{i=1}^{N} C_i,$$

where $N \sim \text{Geo}(f)$ and $C_i \sim E(\nu_c)$. Since the sum of exponential distributions follows the Gamma distribution, we have that $C \sim \text{Gamma}(n, \nu_c)$ when $N = n$ and for $c \geq 0$ that

$$\mathbb{P}[C \leq c | N = n] = \mathbb{P}\left[\sum_{i=1}^{n} C_i \leq c\right] = \int_{x=0}^{c} \frac{x^{n-1} e^{-x\nu_c} (\nu_c)^n}{(n-1)!} dx.$$

We can use this fact to obtain for $c \geq 0$ that

$$\begin{aligned}
\mathbb{P}[C \leq c] &= \sum_{n=1}^{\infty} \mathbb{P}[C \leq c | N = n] \mathbb{P}[N = n] \\
&= \sum_{n=1}^{\infty} \int_{x=0}^{c} \frac{x^{n-1} e^{-x\nu_c} (\nu_c)^n}{(n-1)!} dx f(1-f)^{n-1} \\
&= \int_{x=0}^{c} f e^{-x\nu_c} \nu_c \sum_{m=0}^{\infty} \frac{(1-f)^m x^m (\nu_c)^m}{m!} dx \qquad (m = n-1) \\
&= \int_{x=0}^{c} f e^{-x\nu_c} \nu_c e^{(1-f)x\nu_c)} dx \\
&= \int_{x=0}^{c} f \nu_c e^{-f\nu_c x} dx \\
&= 1 - e^{-cf\nu_c},
\end{aligned}$$

which means that $C \sim E(f\nu_c)$ and the expected total core-task requirement is $\mathbb{E}[C] = \frac{1}{f\nu_s}$.

1.4.2 The shared-tasks in a package

The analysis of the shared-tasks is very similar to the analysis of the core-tasks. The different however is that the total requirement of the shared-tasks is equal to

$$S = \sum_{i=1}^{N-1} S_i,$$

where $N \sim \text{Geo}(f)$ and $S_i \sim E(\nu_s)$. This means for $N = n \geq 2$ that $S \sim \text{Gamma}(n-1, \nu_s)$ so

$$\mathbb{P}[S \leq s | N = n] = \mathbb{P}\left[\sum_{i=1}^{n-1} S_i \leq s\right] = \int_{x=0}^{s} \frac{x^{n-2} e^{-x\nu_s} \left(\nu_s\right)^{n-1}}{(n-2)!} dx.$$

Notice that for $n = 1$ we have $\mathbb{P}[S \leq s | N = n] = 1$. This can be used to obtain for $s \geq 0$ that

$$\begin{aligned}
\mathbb{P}[S \leq s] &= \sum_{n=1}^{\infty} \mathbb{P}[S \leq s | N = n] \mathbb{P}[N = n] \\
&= f + \sum_{n=2}^{\infty} \int_{x=0}^{s} \frac{x^{n-2} e^{-x\nu_s} \left(\nu_s\right)^{n-1}}{(n-2)!} dx f (1-f)^{n-1} \\
&= f + \int_{x=0}^{s} f(1-f) e^{-x\nu_s} \nu_s \sum_{m=0}^{\infty} \frac{(1-f)^m x^m \left(\nu_s\right)^m}{m!} dx \qquad (m = n-2) \\
&= f + (1-f) \int_{x=0}^{s} f e^{-x\nu_s} \nu_s e^{(1-f)x\nu_s)} dx \\
&= f + (1-f) \int_{x=0}^{s} f \nu_s e^{-f\nu_s x} dx \\
&= f + (1-f)(1 - e^{-sf\nu_s}) \\
&= 1 - (1-f) e^{-sf\nu_s}.
\end{aligned}$$

This gives a probability density function of

$$f_S(s) = \frac{d}{ds} \mathbb{P}[S \leq s] = (1-f) f \nu_s e^{-sf\nu_s}.$$

This can be used to calculate the expected total shared-task requirement of

$$\begin{aligned}\mathbb{E}[S] &= \int_{s=0}^{\infty} s f_S(s) ds \\ &= (1-f) \int_{s=0}^{\infty} s f \nu_s e^{-s f \nu_s} \\ &= (1-f) \frac{1}{f \nu_s} \\ &= \frac{1-f}{f \nu_s},\end{aligned}$$

because the integral is equal to the expectation of a random variable with the $E(f\nu_s)$ distribution.

1.4.3 The inter-arrival times of the core-tasks

The inter-arrival times of the outer-packages are $E(\lambda)$ distributed. This means the number of arrived packages P_t at time t is Poisson distributed with parameter λt.