Group Zipf: Second DC Homework

Dilara Isikli *1891292*, ***Alexandru Melnic*** *1692625*, ***Vedat Katirci*** *1908990*, ***Lorenzo Loretucci*** *1903794*

PART 1

β parameter

Given a mean arrival rate of tasks Λ and N servers each with a mean serving rate μ, the utilization coefficient ρ can be written as: $\rho = \frac{\Lambda}{N\mu}$. In our simulation the expected inter-arrival time between tasks is given by: E[T] and the mean processing time of tasks is: E[X] $= \beta\,\Gamma(1+1/\alpha)$. From the previous expressions we get to the following form of β :

$$\beta = \frac{\rho N E[T]}{\Gamma(1+1/\alpha)}$$

With our values of α, β, T_0, $E[T]$ and N we get that: $\beta = 50\,\rho$.

Mean system time

We simulated the 3 policies JSQ, POD-3 and JBT-3 with $T_{update} = 1000$. The simulation was conducted by generating 200000 tasks for different values of ρ. From each simulation, for each ρ, we evaluated the mean system time. Furthermore to build the confidence intervals, or more specifically to calculate the standard error of the sample mean $\hat{se}(\overline{X}_n)$, we performed the simulation 50 times. Results in Fig. 1.

Mean message overhead

The mean number of messages sent per task was calculated as the total number of messages exchanged in a simulation divided by the total amount of tasks. For the JSQ and POD this is a constans for each ρ, but the result is not trivial for the JBT case, where the mean number of messages is approximately constant. Confront between the 3 policies in Fig. 2.

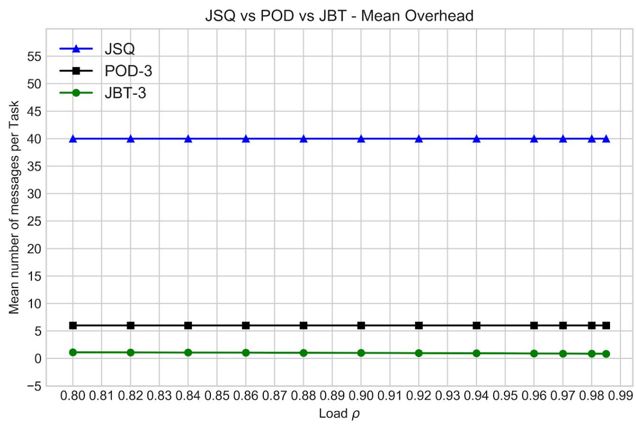

Fig. 2 : Mean message overhead for the JSQ, POD and JBT.

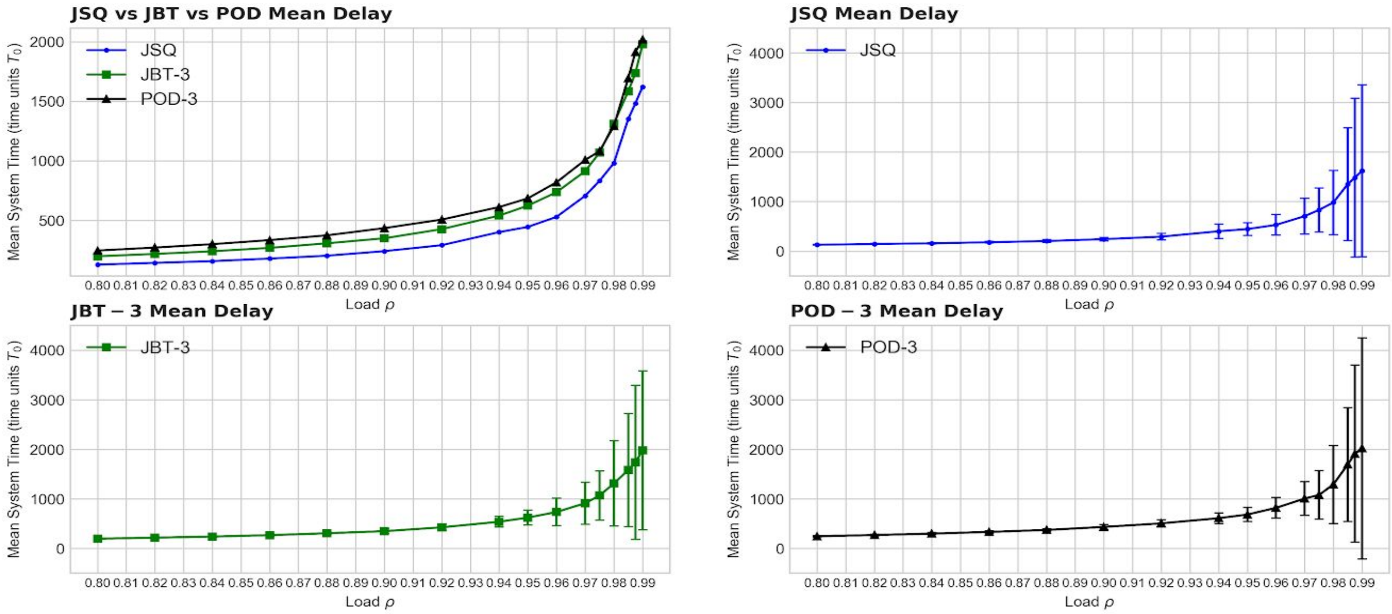

Fig. 1: Mean Delay for JSQ, POD and JBT

Group Zipf: Second DC Homework

Dilara Isikli *1891292*, ***Alexandru Melnic*** *1692625*, ***Vedat Katirci*** *1908990*, ***Lorenzo Loretucci*** *1903794*

PART 2

Invented algorithm

The load balancer has the information about the processing time of each task in advance. Given this information we decided to implement an alternative version of the JSQ with the length of a queue in time and not in number of tasks. More precisely, when a task arrives at the load balancer, it checks the state of each queue and sends the task to the one with the shortest total processing time, keeping into account also the state of that server. So the queue is evaluated as the sum of total processing times of the tasks in the queue plus the current processing time of the server.
The number of messages in the invented algorithm is the same as for the JSQ.

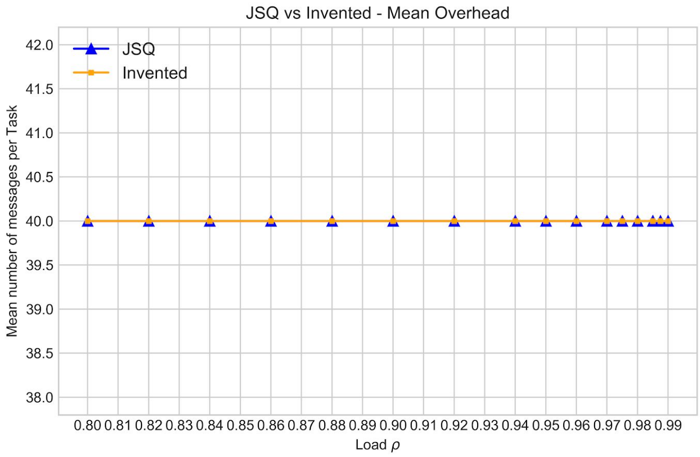

Fig.4: Mean message overhead JSQ vs Invented

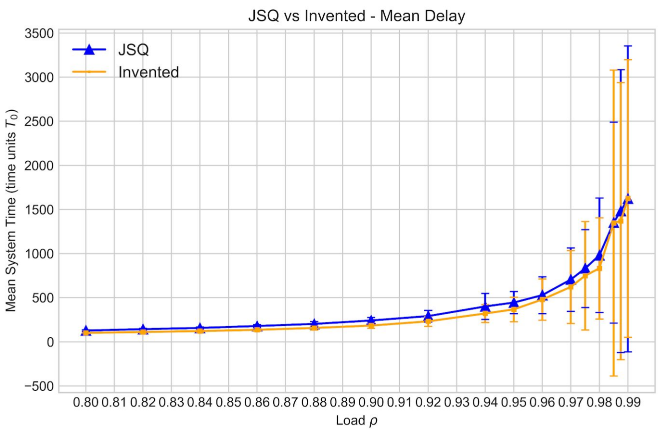

Fig.3: Mean Delay JSQ vs Invented