### Block Size

Figure 5‑103 to Figure 5‑108 show the relationship between various performance metrics and the block size . Both simulation results and analytical results are shown for comparison.

Figure 5‑103 illustrates the impact of the block size on the average waiting times in the customer queue for HP, LP, and overall customers. As increases, the decreases. The reduction is more pronounced for , while remain consistently low. This is because larger blocks allow more customers to be served per service cycle, thereby reducing the time, especially for LP customers who tend to experience longer delays when is small. In addition, the is much smaller than . This is because the HP customers have non-preemptive priority over LP customers in the customer queue. Furthermore, with impatience is smaller than that without impatience. This is because the impatience mechanism causes customers to leave the queue once their impatience threshold is reached, which further suppresses queue buildup and contributes to the decline in . Lastly, a strong agreement is observed between the analytical and simulation results.

Figure 5‑104 illustrates the impact of the block size on the average waiting times in the block queue for HP, LP, and overall customers. As increases, the average waiting time in the block queue remains nearly constant for all priority levels. This indicates that the time each block spends in the consensus queue is determined by the consensus rate and system state transition rate, and is independent of block size. In addition, the is smaller than . This is because is larger than . Furthermore, with impatience is the same as that without impatience. This is because once a block is formed, impatience mechanism has no effect on it. Lastly, a strong agreement is observed between the analytical and simulation results.

Figure 5‑105 illustrates the impact of the block size on the average waiting times in the system for HP, LP, and overall customers. As increases, the decreases. The decline is especially significant for LP customers, while the remains relatively constant. This is because larger blocks allow more customers to be served per service cycle, which benefits LP customers who are otherwise delayed by the non-preemptive priority mechanism. In addition, the is much smaller than . This is because the HP customers have non-preemptive priority over LP customers in the customer queue and is larger than . Furthermore, with impatience is smaller than that without impatience. This is because the impatience mechanism causes customers to leave the queue once their impatience threshold is reached, which further suppresses queue buildup and contributes to the decline in . Lastly, a strong agreement is observed between the analytical and simulation results.

Figure 5‑106 illustrates the impact of the block size on the average numbers of customers in the block queue for HP, LP, and overall customers. As increases, the average number of customers in the block queue rises gradually across all priority levels. The increase is most noticeable for and , while the remains relatively low and stable. This indicates that although larger blocks permit more customers per batch, the average block occupancy tends to saturate when the customer arrival rate is equal to the effective service rate of the customer queue. In addition, is smaller than . This is because the HP customers have non-preemptive priority over LP customers in the customer queue and the HP customers have faster consensus rate than LP customers in the block queue, and therefore more LP customers remain waiting in the customer queue before being batched. Additionally, with impatience is smaller than that without impatience. This is because the impatience mechanism causes customers to leave the queue once their impatience threshold is reached, which contributes to a reduction in . Lastly, a strong agreement is observed between the analytical and simulation results.

Figure 5‑107 illustrates the impact of the block size on the blocking probabilities for HP, LP, and overall customers. As increases, the blocking probability generally decreases. However, a slight increase is observed at across all priority levels. This is because, when the block queue is occupied, the effective capacity of the customer queue becomes , which may be too small when is large. With impatience, fewer users remain in the queue, resulting in smaller batches and slower queue clearance, which makes blocking more likely. Without impatience, fuller batches are formed, and the queue is cleared more efficiently before reaching the limit. The decline is more pronounced for , which is initially much higher and drops significantly with increasing . This is because larger blocks allow more customers to be served per block generation cycle, thereby reducing the chance of the customer queue reaching its capacity limit, especially for LP customers who are more likely to be blocked under limited queue capacity. In addition, is smaller than . This is because the capacity limit of the customer queue for the HP customers is no smaller than that for LP customers. Additionally, with impatience is smaller than that without impatience. This is because the impatience mechanism causes customers to leave the queue once their impatience threshold is reached, which contributes to a reduction in . Lastly, a strong agreement is observed between the analytical and simulation results.

Figure 5‑108 illustrates the impact of the block size on the system throughputs for HP, LP, and overall customers. As increases, the system throughput increases across all priority levels and then gradually saturates. Both and increase with , with the growth being more significant for LP customers. This is because larger blocks enable more customers to be processed per consensus cycle. However, the throughput eventually approaches a limit determined by the customer arrival rate and the impatient rate, which is less than the system processing capacity. In addition, is smaller than . This is because the customer arrival rate of the HP customers is smaller than that of LP customers. As a result, with impatience is smaller than that without impatience. Lastly, a strong agreement is observed between the analytical and simulation results.

Figure 5‑109 illustrates the impact of the block size on the impatient probabilities for HP, LP, and overall customers. As increases, the impatient probability across all priority levels. This is because larger blocks allow more customers to be served in each service cycle, which reduces the waiting time in the customer queue. As a result, the probability that customers reach their impatience threshold and leave the system becomes lower. In addition, is higher than , as LP customers are more likely to experience longer waits due to the non-preemptive priority discipline. As described above,  with impatience is larger than without impatience. Lastly, a strong agreement is observed between the analytical and simulation results.

Figure 5‑103: Block size vs. average waiting time in the customer queue

Figure 5‑104: Block size vs. average waiting time in the block queue

Figure 5‑105: Block size vs. average waiting time in the system

Figure 5‑106: Block size vs. average number of customers in block queue

Figure 5‑107: Block size vs. blocking probability

Figure 5‑108: Block size vs. system throughput

Figure 5‑109: Block size vs. the impatient probability

# Conclusion

This research investigates a blockchain-based queuing system that models the transaction process under different combinations of customer priority and impatience, while incorporating ON/OFF operational states that reflect the stochastic availability of block generation and consensus phase. The blockchain queuing system comprises two queues in series: a customer queue, where customers wait to be grouped into a block, and a block queue, where grouped customers await consensus. The system supports a partial batch generation mechanism, allowing up to customers to be grouped at once. The ON/OFF states represent the operational availability of the system; during OFF periods, block generation and consensus are suspended, while arrivals continue. We model four scenarios: (1) single-class customers without impatience, (2) two-class customers without impatience, (3) Single-class customers with impatience, and (4) two-class customers with impatience.

For each scenario, we construct a multi-dimensional Markov chain that captures the state of the system. Balance equations are formulated and solved iteratively until the steady-state probability distribution is obtained. Based on these probabilities, key performance metrics are calculated, including throughput, blocking probability, and average waiting times. To validate our analytical findings, we develop a discrete-event simulation implemented in C++. The simulation strictly adheres to the queuing logic and service discipline of each scenario to ensure consistency with the analytical model.

In conclusion, based on our simulation and analytical results, we have made several interesting observations. First, as increases, initially grows but eventually stabilizes around a constant value. Second, in the priority-based scenarios, remains nearly constant, but increases with but the increase gradually slows down. Third, in scenario 4, decreases with increasing at first, but shows a slight increase when becomes large. Fourth, when increasing in the priority-based scenarios, and remains constant. However, shows a downward trend. Fourth, in the priority-base scenarios, increasing leads to an upward trend in . Lastly, scenarios with impatience leads to improvements in most performance metrics compared to scenarios without impatience. Notably, however, remains unaffected by the impatience rate.

Future research directions include incorporating a more realistic voting mechanism into the consensus phase to better reflect practical blockchain operations. In addition, further extensions may involve optimizing the batch policy dynamically based on the real-time queue state, and enhancing the impatience modeling to capture the total time until departure, including both queueing and consensus delays.