# **Simulation model**

This chapter provides a comprehensive explanation of simulation models for two situations. The first scenario involves a single node. The second scenario is a network composed of three interconnected nodes. The above are both batch arrivals, and each may consist of one or two packets arriving at a time. To create and simulate these models, we code in C Programming Language.

## Scenario 1

Under this scenario's conditions, we consider a single sensor node with two priorities. Both priorities packets require one energy. The packets in this scenario are in batch arrival, which arrive with one or two packets at a time. The energy required for operation can be supplied by either harvested energy or a regular battery. For clarity, we have divided the simulation into one main program and seven subprograms.

### Main program

The program depicted in Fig. 4 - 1 follows several steps to simulate the system. First, all variables used in the simulation are initialized, including setting the statistics parameters to zero, the server state to idle, the next departure time to infinite. Next, the user is prompted to input various parameters via the keyboard, such as the packet queue capacity , energy queue capacity , mean arrival rate of energy units , mean batch arrival rates of one packet with HP and LP, mean batch arrival rates of two packets with HP and LP, mean service rates, mean impatient rates of HP and LP packets , the probability of HP and LP packets using regular battery , and the number of packets required. The program then generates the next energy arrival time and batch packet arrival times using an exponential variate function based on the input values. In the second step, the program compares the times of the seven possible events and selects the earliest one to execute the corresponding subprogram. Finally, a "while loop" is used to repeat the second step until the required number of packets are served, at which point the simulation ends and the performance results are displayed.

### HP packet arrival with one packet per batch subprogram

Fig. 4 - 2 illustrates the flow chart of the subprogram for HP packet arrival with one packet per batch. Initially, the program updates the current simulation time as the HP packet arrival time and generates the next HP packet arrival time using the exponential variate function with the arrival rate . The program then accumulates the total number of arrived packets, packets in the queue, and packets in the system separately. It checks whether the packet queue is full or not. If the packet queue is not full, the program adds one packet to the queue and inserts the related information of the HP packet at the front of all LP packets. It generates the impatient time using the exponential variate function with . On the other hand, if the queue is full, the program increases the number of blocked packets by one. The program then describes the situations divided into the following cases.

1. The first step is to check the status of the server. If it is not "IDLE", we update the next impatient time by considering the impatient time of packets waiting in the queue. However, if the server is "IDLE", we need to check whether the energy units in the energy queue are sufficient to meet the energy requirement of the packet at the front of the packet queue.
2. The second step involves checking whether there are enough energy units in the energy queue to serve the packet at the head of the packet queue. If there are enough units, the relevant statuses are updated, including the energy and packet queue sizes, server status, and waiting and service times. The departure time is then generated using an exponential variate function, and the packets in the queue are shifted one position. The next impatient time is updated or set to "infinite" based on the status of the packet queue.
3. If the number of energy units in the energy queue is insufficient to serve the packet, a random value between 0 and 1 is generated, which is compared with the probability of regular battery usage . If it is less than or equal to , we update the relevant statuses and accumulate the appropriate time, including the number of energy units consumed from the regular battery, which are not in the energy queue. Next, the departure time is generated using the exponential variate function with , and all the remaining packets in the packet queue are moved forward one place. Finally, depending on whether the packet queue is empty or not, we decide whether to update the next impatient time or set it to infinite.
4. If the number of energy units in the energy queue is insufficient and the random value is greater than , then the next impatient time needs to be updated based on the impatient time of packets in the queue.

### LP packet arrival with one packet per batch subprogram

Fig. 4 - 3 illustrates the flow chart of the subprogram for LP packet arrival with one packet per batch. To begin, the current simulation time is updated as the LP packet arrival time and the next LP packet arrival time is generated using the exponential variate function with . Additionally, the total number of arrived packets, packets in the queue, and packets in the system are accumulated separately. The program checks whether the packet queue is full, and if not, it adds one packet to the queue and stores the related information of the LP packet at the end of the queue. The impatient time is generated using the exponential variate function with . If the queue is full, the number of blocked packets is increased by one. The program then describes the cases that follow.

1. The first step is to check the status of the server. If it is not "IDLE", we update the next impatient time by considering the impatient time of packets waiting in the queue. However, if the server is "IDLE", we need to check whether the energy units in the energy queue are sufficient to meet the energy requirement of the packet at the front of the packet queue.
2. The second step involves checking whether there are enough energy units in the energy queue to serve the packet at the head of the packet queue. If there are enough units, the relevant statuses are updated, including the energy and packet queue sizes, server status, and waiting and service times. The departure time is then generated using an exponential variate function, and the packets in the queue are shifted one position. The next impatient time is updated or set to "infinite" based on the status of the packet queue.
3. If the number of energy units in the energy queue is insufficient to serve the packet, a random value between 0 and 1 is generated, which is compared with the probability of regular battery usage . If it is less than or equal to , we update the relevant statuses and accumulate the appropriate time, including the number of energy units consumed from the regular battery, which are not in the energy queue. Next, the departure time is generated using the exponential variate function with , and all the remaining packets in the packet queue are moved forward one place. Finally, depending on whether the packet queue is empty or not, we decide whether to update the next impatient time or set it to infinite.
4. If the number of energy units in the energy queue is insufficient and the random value is greater than , then the next impatient time needs to be updated based on the impatient time of packets in the queue.

### HP batch arrival with two packets per batch subprogram

Fig. 4 - 4 illustrates the flow chart of the subprogram for HP batch arrival with two packets per batch. Initially, the program updates the current simulation time as the two HP packets arrival time and generates the next two HP packets arrival time using the exponential variate function with the batch arrival rate . The program then accumulates the total number of arrived packets, packets in the queue, and packets in the system separately. It checks whether the packet queue is full or not. If the packet queue has at least two seat, the program adds two packets to the queue and inserts the related information of two HP packets at the front of all LP packets. It generates the impatient time using the exponential variate function with . If the packet queue has only one seat, the program adds one packet to the queue and inserts the related information of one HP packet at the front of all LP packets, and increases the number of blocked packets by one. If the queue is full, the program increases the number of blocked packets by two. The program then describes the situations divided into the following cases.

1. The first step is to check the status of the server. If it is not "IDLE", we update the next impatient time by considering the impatient time of packets waiting in the queue. However, if the server is "IDLE", we need to check whether the energy units in the energy queue are sufficient to meet the energy requirement of the packet at the front of the packet queue.
2. The second step involves checking whether there are enough energy units in the energy queue to serve the packet at the head of the packet queue. If there are enough units, the relevant statuses are updated, including the energy and packet queue sizes, server status, and waiting and service times. The departure time is then generated using an exponential variate function, and the packets in the queue are shifted one position. The next impatient time is updated or set to "infinite" based on the status of the packet queue.
3. If the number of energy units in the energy queue is insufficient to serve the packet, a random value between 0 and 1 is generated, which is compared with the probability of regular battery usage . If it is less than or equal to , we update the relevant statuses and accumulate the appropriate time, including the number of energy units consumed from the regular battery, which are not in the energy queue. Next, the departure time is generated using the exponential variate function with , and all the remaining packets in the packet queue are moved forward one place. Finally, depending on whether the packet queue is empty or not, we decide whether to update the next impatient time or set it to infinite.
4. If the number of energy units in the energy queue is insufficient and the random value is greater than , then the next impatient time needs to be updated based on the impatient time of packets in the queue.

### LP batch arrival with two packets per batch subprogram

Fig. 4 - 5 illustrates the flow chart of the subprogram for LP batch arrival with two packets per batch. Initially, the program updates the current simulation time as the two LP packets arrival time and generates the next two LP packets arrival time using the exponential variate function with the batch arrival rate . Additionally, the total number of arrived packets, packets in the queue, and packets in the system are accumulated separately. The program checks whether the packet queue is full, and if not and still has at least two seats, it adds two packets to the queue and stores the related information of the two LP packets at the end of the queue. The impatient time is generated using the exponential variate function with . If the packet queue has only one seat, the program adds one packet to the queue at the end of the queue, and increases the number of blocked packets by one. If the queue is full, the number of blocked packets is increased by two. The program then describes the cases that follow.

1. The first step is to check the status of the server. If it is not "IDLE", we update the next impatient time by considering the impatient time of packets waiting in the queue. However, if the server is "IDLE", we need to check whether the energy units in the energy queue are sufficient to meet the energy requirement of the packet at the front of the packet queue.
2. The second step involves checking whether there are enough energy units in the energy queue to serve the packet at the head of the packet queue. If there are enough units, the relevant statuses are updated, including the energy and packet queue sizes, server status, and waiting and service times. The departure time is then generated using an exponential variate function, and the packets in the queue are shifted one position. The next impatient time is updated or set to "infinite" based on the status of the packet queue.
3. If the number of energy units in the energy queue is insufficient to serve the packet, a random value between 0 and 1 is generated, which is compared with the probability of regular battery usage . If it is less than or equal to , we update the relevant statuses and accumulate the appropriate time, including the number of energy units consumed from the regular battery, which are not in the energy queue. Next, the departure time is generated using the exponential variate function with , and all the remaining packets in the packet queue are moved forward one place. Finally, depending on whether the packet queue is empty or not, we decide whether to update the next impatient time or set it to infinite.
4. If the number of energy units in the energy queue is insufficient and the random value is greater than , then the next impatient time needs to be updated based on the impatient time of packets in the queue.

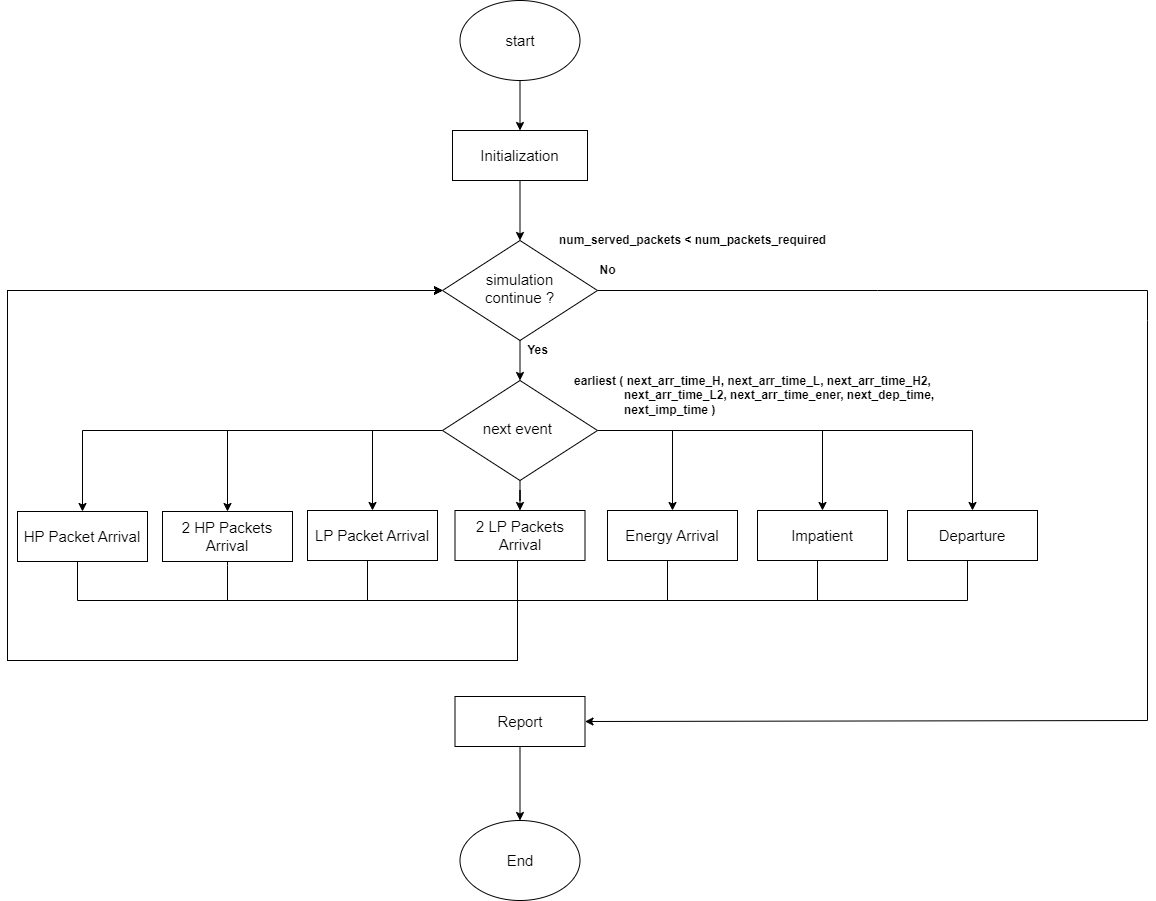


Fig. 4 - 1: Flow chart graphic of main program for scenario 1

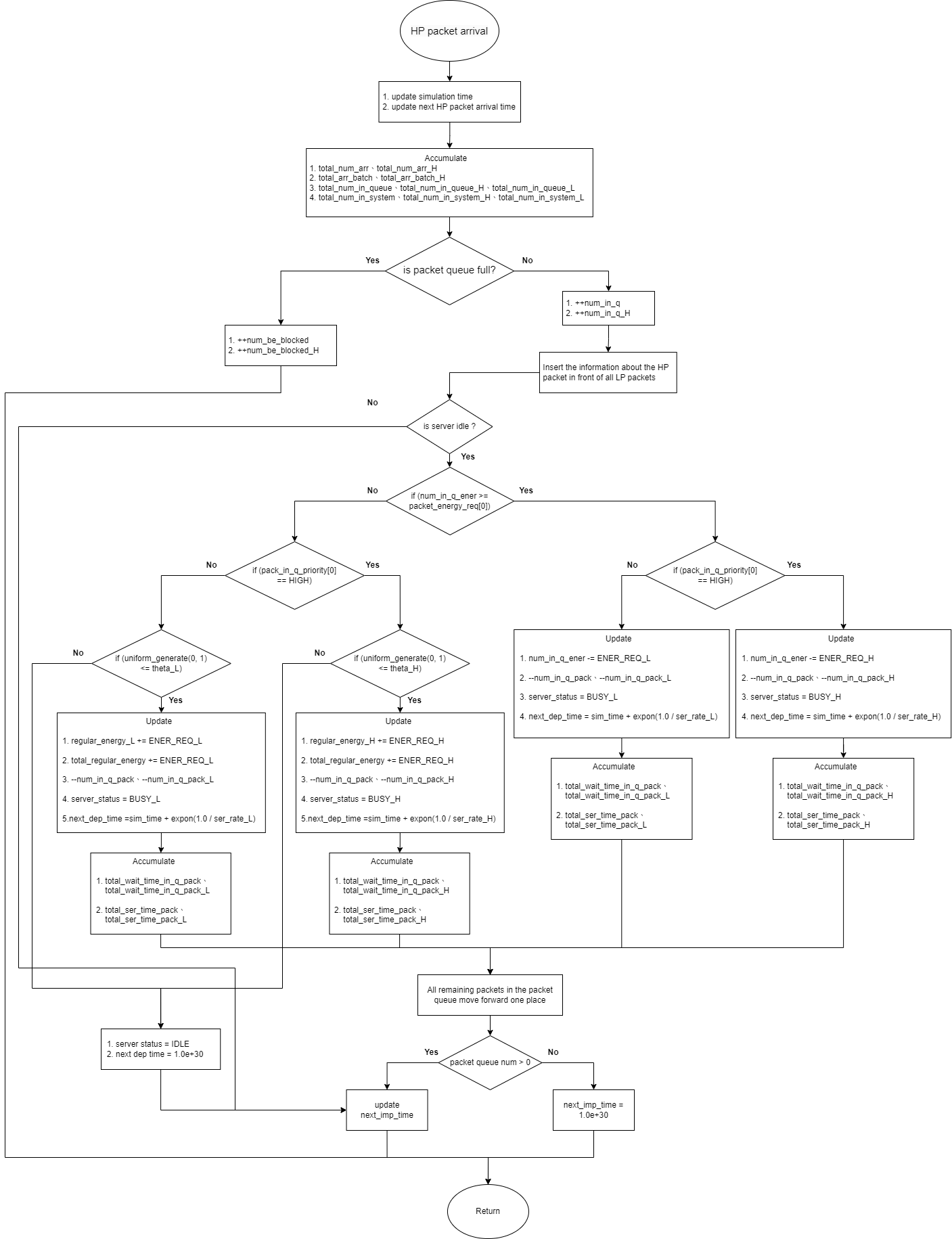


Fig. 4 - 2: Flow chart graphic of HP packet arrival with one packet per batch subprogram for scenario 1

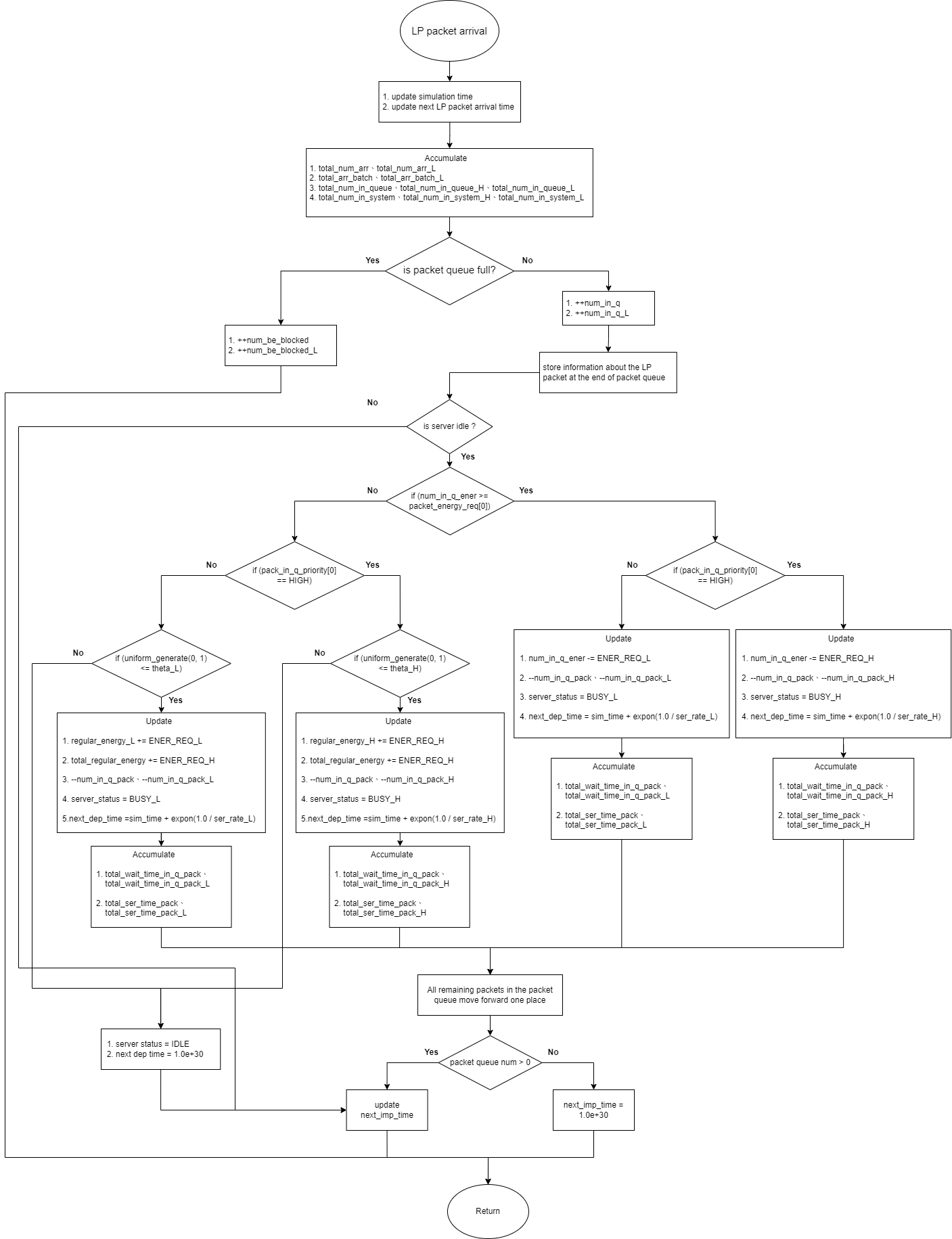


Fig. 4 - 3: Flow chart graphic of LP packet arrival with one packet per batch subprogram for scenario 1

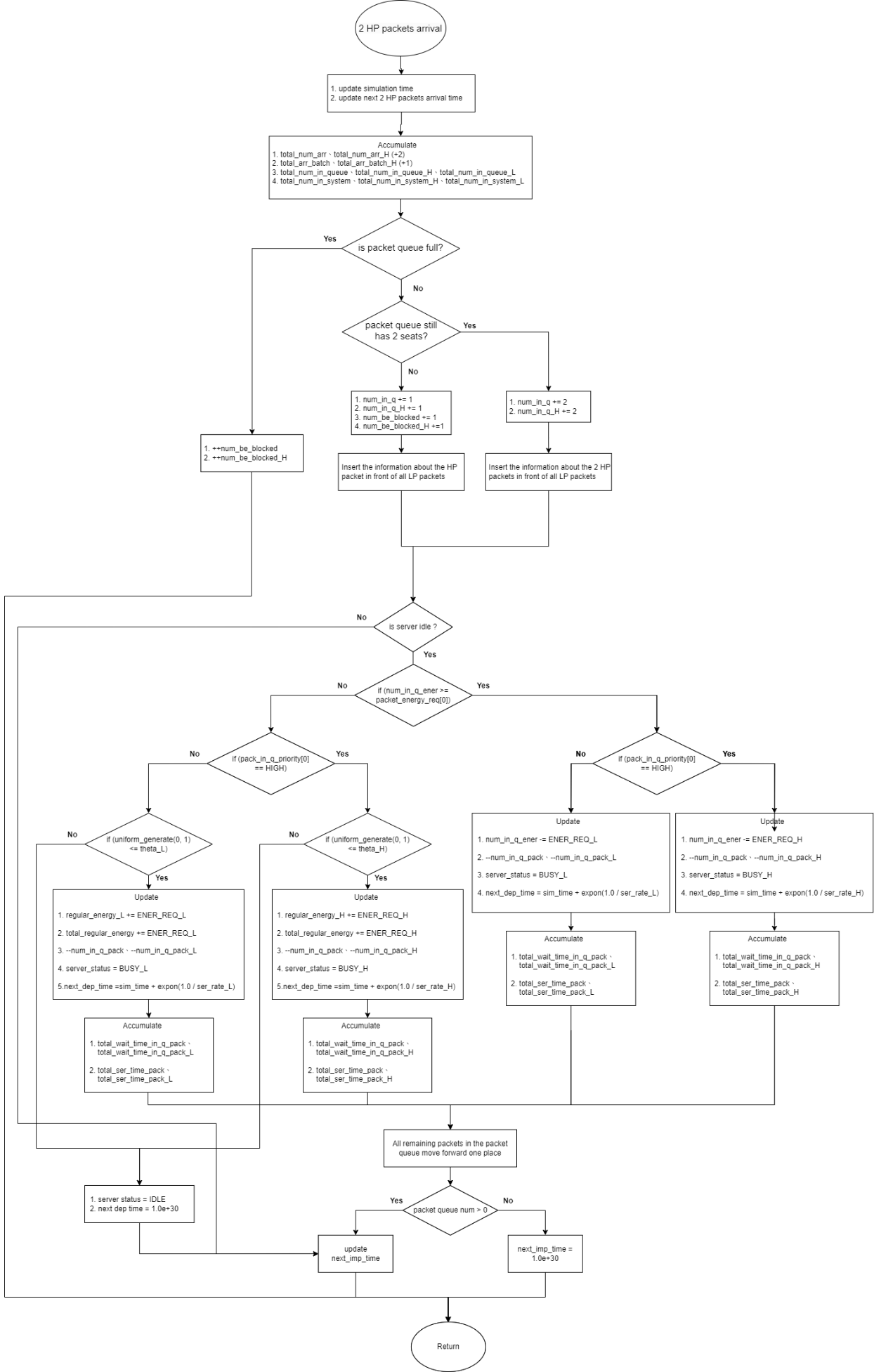


Fig. 4 - 4: Flow chart graphic of HP batch arrival with two packets per batch subprogram for scenario 1

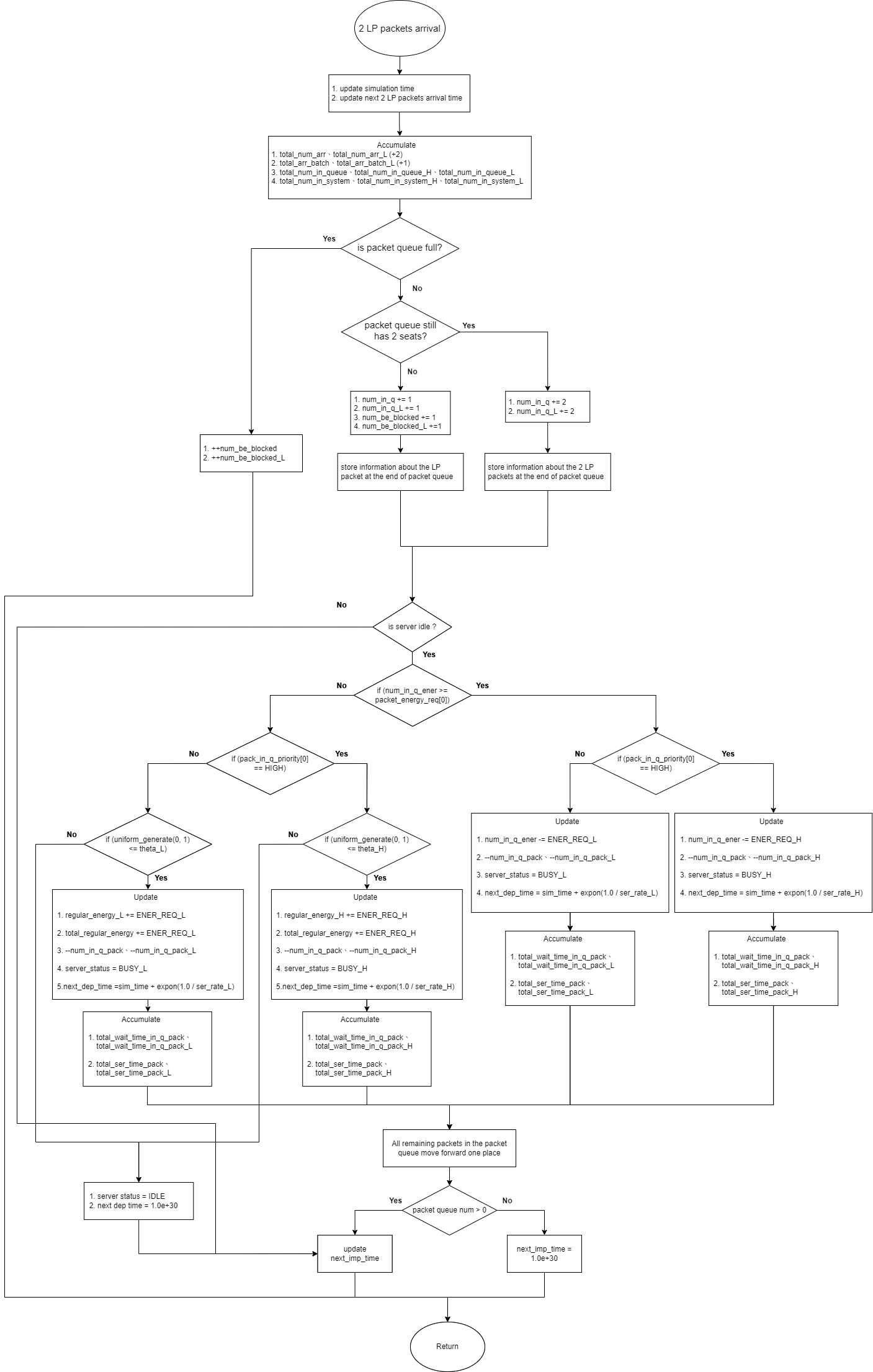


Fig. 4 - 5: Flow chart graphic of LP batch arrival with two packets per batch subprogram for scenario 1