Chapter 5

Performance Evaluation

5.1 Simulation Model

In this chapter, we conduct some experimental evaluations of TrustVoting algorithm in terms of the faulty detection rate and the false positive rate. Specifically, we specify an interesting region B to obtain the current readings sensed by the sensors in the region B, where the set of these current readings is denoted as X_B. Assume that Y_B is a set of faulty readings in X_B. After executing algorithm TrustVoting, we can filter out a set of faulty readings denoted as Y'_B, and obtain a subset of current readings $X'_B \subseteq X_B$ without faulty readings. The faulty detection rate is defined as $\frac{|Y_B \cap Y'_B|}{|Y_B|}$ and the false positive rate is defined as $\frac{|(Y_B \cup Y'_B) - (Y_B \cap Y'_B)|}{|X_B|}$. In other words, the faulty detection rate is how many percentage of faulty readings identified and the false positive rate is how many percentage of faulty readings (respectively, normal readings) identified as normal (respectively, faulty).

Our simulation model consists of the environmental model, the faulty model and the query model. The simulated sensor network is deployed in a 500 by 500 sensing field for monitoring a synthetic environment where the reading range is $[-25, 275]$. To model a severe environment, our environmental model is designed to be capable of generating multiple events at any place in the sensing field. Each event runs a predefined finite state machine to make readings

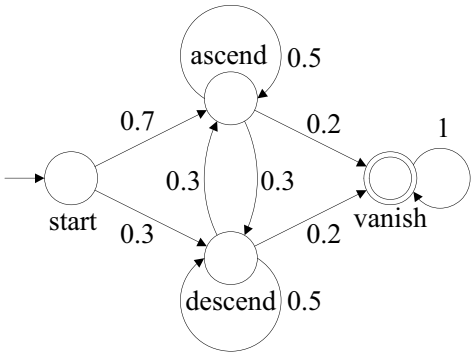

Figure 5.1: The finite state machine of an event.

ascending or descending. The finite state machine is very simple and has only four states: start, ascend, descend and vanish states shown in Figure 5.1. The values on edges are the transition probabilities. When an event occurs, the current state of the event is on the start state initially, and every time the current state will transit to other states in accordance with probabilities given. When the current state is on the ascend state, the reading at the position of the event will ascend with a random amount. Similarly, when the current state is on the descend state, the reading will descend with a random amount. However, if the current state is on the vanish state, the event will be disappear and the environment will become normal gradually. When the reading of an event changes, the surrounding readings will be affected by this event and the affecting degree will be inversely proportional to the distance from the event. Some readings do not follow the environmental model and these reading are thus viewed as faulty readings. Our faulty model is controlled by a parameter *noise_prob*, a probability that a sensor reports noise. If a sensor is faulty, we will set *noise_prob* of the sensor to 1. Consequently, this sensor will always report noise. In our simulation model, assume that a normal sensor can still report faulty readings, and the default *noise_prob* for a normal sensor is set to 0.1. A noise reading (referred to as a faulty reading) is randomly biased towards the real reading. The biased amount is in the range of $[-50, 50]$. Therefore, a faulty sensor will sense random readings and these random readings will not be too large or too small. With the presence of faulty readings, range queries are issued in a random interval of time. A query

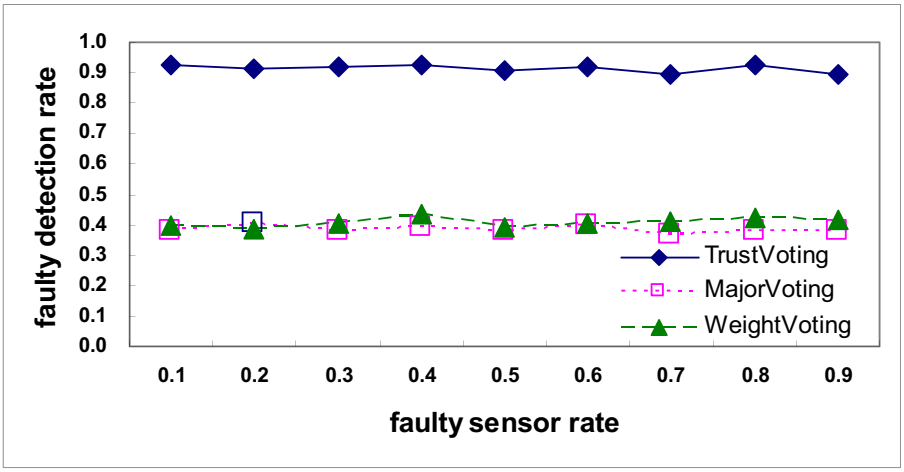

Figure 5.2: The faulty detection rate of the three algorithms.

region is specified as a rectangle and the size of a rectangle ranges from 80 by 80 up to 160 by 160. In each experiment, there are around 100 queries issued. Both the faulty detection rate and the false positive rate of these queries are averaged to represent the result of a experiment. The same experiment is repeated 100 times and the average of all experimental results is the final result.

To show the performance of TrustVoting, we implemented other two algorithms: the classical majority voting (denoted as MajorVoting), and the distance weighted voting (denoted as WeightVoting). MajorVoting and WeightVoting are based on the comparison of single values. In addition, sensitivity analysis for some important parameters, such as the number of iterations of calculating SensorRank, the length of the reading behavior, the number of neighbors, and the similarity threshold is conducted.

5.2 Experimental Results

5.2.1 Performance of TrustVoting

To show the performance of TrustVoting, both MajorVoting and WeightVoting are based on the comparison of single values. When a sensor s_i senses an unusual reading and asks one

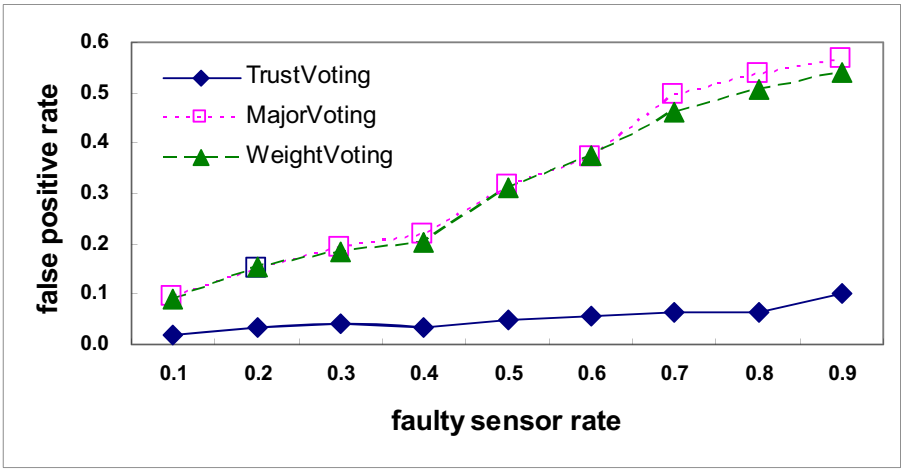

Figure 5.3: The false positive rate of the three algorithms.

of its neighbors s_j, if the difference between their readings exceeds a predefined threshold, s_j considers s_i as a faulty sensor and gives a negative vote to s_i. The threshold value in our simulation is set to 30.

In this experiment, we compare the faulty detection rates and the false positive rates of the three algorithms with various faulty sensor rates. Figure 5.2 shows that TrustVoting can filter out around 90% faulty readings while MajorVoting and WeightVoting can only filter out around 40% faulty readings. Intuitively, the faulty detection rate will be high in a low faulty sensor rate, and will decrease when the faulty sensor rate increases. However, since faulty readings in our faulty model are biased normal readings, it is hard to identified faulty readings for MajorVoting and WeightVoting. Therefore, when the faulty sensor rate is low, the faulty detection rate is around 0.4. When the faulty sensor rate increases, although almost all sensors are faulty, a faulty sensor can still be identified as faulty by its faulty neighbors due to dissimilar readings. Thus, the faulty detection rates are the same with various faulty sensor rates.

Figure 5.3 shows the false positive rates of the three algorithms. The false positive rates of these algorithms increase as the faulty sensor rate increases. It is worth mentioning that TrustVoting can limit the false positive rate under 0.1. Note that the performance of

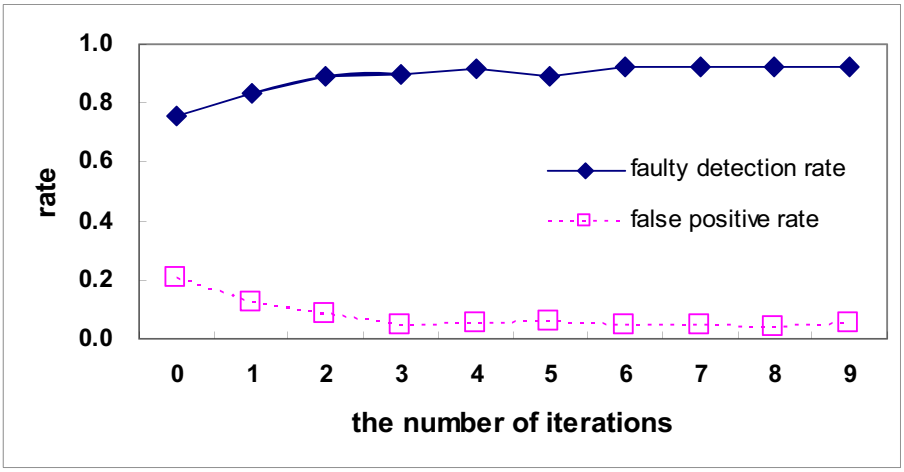

Figure 5.4: The impact of the number of iterations for calculating SensorRank.

WeightVoting is supposed to be better than that of MajorVoting. In fact, their performances are very close. As mentioned before, due to the dominated problem, WeightVoting cannot perform well.

5.2.2 Impact of SensorRank

In this experiment, we examine the impact of the number of iterations of calculating SensorRank (i.e., the parameter δ). The faulty rate is set to 0.5 to show the trend in the faulty detection rate and the false positive rate. In Figure 5.4, when δ increases, the faulty detection rate increases, whereas the false positive rate decreases. The reason is that when the number of iterations for calculating SensorRank is 0, SensorRank of each sensor is 1. In this situation, our TrustVoting is just like a general weighted voting. After three iterations for calculating SensorRank, we can find that SensorRank is converge and thus the faulty detection rate and the false positive rate are also stable.

The convergence rate of SensorRank is further examined. The convergence rate for each sensor s_i is defined as $\frac{\left|rank_i^{(k)} - rank_i^{(k-1)}\right|}{rank_i^{(k-1)}}$ for $k = 1, 2, \ldots 10$. In this simulation, the convergence rates for all sensors are averaged as the experimental result. Figure 5.5 shows that SensorRank will converge fast in a few iterations. Explicitly, the convergence rate drastically decreases after

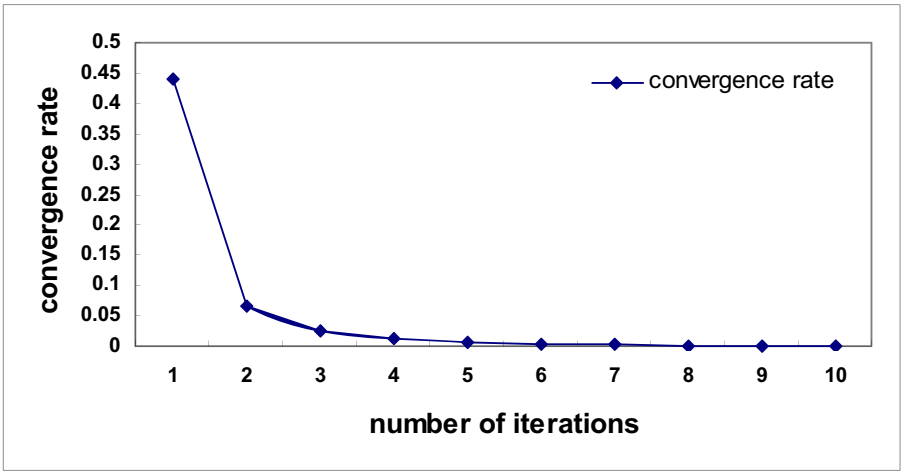

Figure 5.5: The convergence of SensorRank.

three iterations of calculation. Therefore, after three iterations of calculating SensorRank, almost all sensors' SensorRanks are stable.

5.2.3 Impact of Reading Behavior

In this experiment, we examined the impact of the length of the reading behavior. The result is shown in Figure 5.6. When the length of the reading behavior is too long or too short, the performance of filtering faulty reading will not be good. It is intuitive that when Δt is small, wrong filtering will occur easily. When Δt is very large, the reading behavior may contain more noise. In this case, the similarity function generates smaller values and some normal readings are then filtered out. Thus, the false positive rate increases and faulty detection rate is not high.

5.2.4 Impact of Neighbors

Figure 5.7 shows the impact of the number of neighbors. We deployed a large number of sensors to increase the average number of neighbors of each sensor. In order to increase the number of neighbors, we increase the density of the deployment of sensors. Note that the more neighbors a sensor has, the better performance a voting scheme is. Because for a voting-based

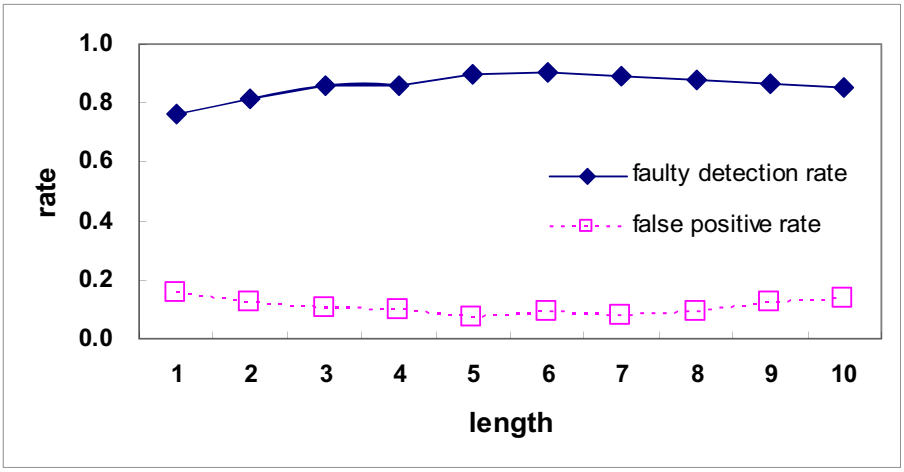

Figure 5.6: The impact of the length of the reading behavior.

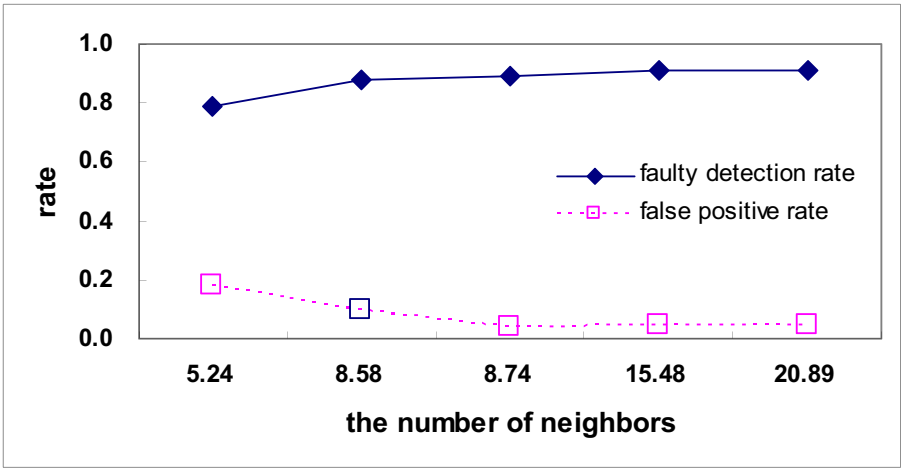

Figure 5.7: The impact of the number of neighbors.

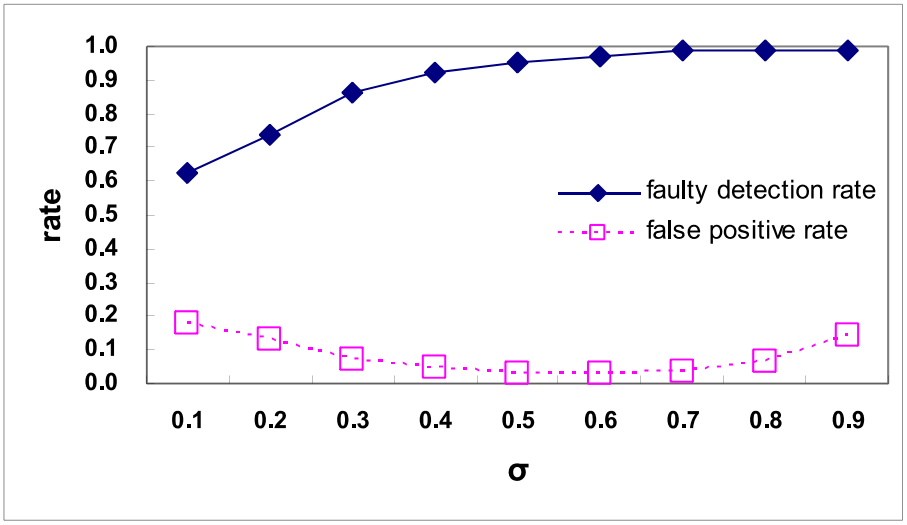

Figure 5.8: The impact of σ.

scheme, if the probability that a sensor makes a wrong decision is low, the probability that the majority of neighbors make wrong decisions is much lower.

5.2.5 Impact of σ

The impact of σ is investigated. Figure 5.8 shows that when σ is small, the faulty detection rate is very small and the false positive rate is high. This is because that any reading behavior can be easily viewed as normal. Therefore, some faulty readings are identified as normal. As can be seen that as the value of σ increases, many sensors are accurately identified as faulty reading. Thus, the faulty detection rate is very high. When σ increases, many normal readings may be identfied as faulty as well. Thus the false positive rate is high when σ is very high.