Modeling Data Stream Intensity in Distributed Stream Processing System

Marcin Gorawski[1,2], Pawel Marks[1], and Michal Gorawski[3]

[1] Silesian University of Technology, Institute of Computer Science,
Akademicka 16, 44-100 Gliwice, Poland
{Marcin.Gorawski,Pawel.Marks}@polsl.pl
[2] Wroclaw University of Technology, Institute of Computer Science,
Wybrzeze Wyspianskiego 27, 50-370 Wroclaw, Poland
Marcin.Gorawski@pwr.wroc.pl
[3] Institute of Theoretical and Applied Informatics, Polish Academy of Sciences,
Baltycka 5, 44-100 Gliwice, Poland
mgorawski@iitis.pl

Abstract. In recent years energy market has changed. Consumers in many countries are free to buy energy from any of the available providers. This requires continuous reading from a huge number of energy meters to evaluate the amount of energy being bought from a particular provider. In this paper we present a fault-tolerant distributed stream processing system for continuous meter readings. The main goal of the system is to store the readings in a stream data warehouse for further analysis. We focus on modeling of the data stream intensity in order to estimate the size of buffers in a network of components composing the system. We present both the mathematical model of the intensity and the simulation results to prove the correctness of the theoretical analysis.

Keywords: stream processing, modeling, distributed system.

1 Introduction

These days it becomes more common to process continuous data streams [1]. It may have application in many domains of our life such as: computer networks (e.g. intrusion detection), financial services, medical information systems (e.g. patient monitoring), civil engineering (e.g. highway monitoring) and more.

Thousands or even millions of energy meters located in households or factories can be sources of meter-reading streams. Continuous analysis of power consumption may be crucial to efficient electricity production. Unlike other media such as water or gas, electricity is hard to store for further use. That is why a prediction of energy consumption may be very important. Real-time analysis of the media meter readings may help manage the process of energy production in the most efficient way.

There are many systems for processing continuous data streams and they are still being developed [2–6]. Various system processing stream data can also be

A. Kwiecień, P. Gaj, and P. Stera (Eds.): CN 2013, CCIS 370, pp. 372–383, 2013.

found in [7–9]. In [10] the fault tolerant Borealis system is presented. This is a dedicated solution for applications where a low latency criterion is essential. Another system facing infinite data streams is described in [11]. Authors of the work deal with sensors producing data continuously, transferring the measured data asynchronously without pooling. They proposed a *Framework in Java for Operators on Remote Data Streams* (Fjords).

In our research we have focused on processing data originating from a radio-based measurement system [12, 13]. We carried research on efficient recovery of interrupted ETL jobs and proposed a few approaches [14, 15] based on the Design-Resume algorithm [16].

Based on the previous experience, we have focused on fault-tolerance and high availability in a distributed stream processing environment. In [17] we proposed a new set of modules increasing the probability that a failure of one or more modules will not interrupt the processing of endless data streams. Then we prepared a simple model [18–21] of data sources to estimate the amounts of data to be processed, useful in the configuration of the environment. In this paper we want to present a more advanced analysis of the intensity of the stream readings to be able to configure buffers of the network components properly.

In Section 2 we define the problem we want to solve. Sections 2.2, 2.3 and 2.4 contain a detailed description of the analysis we propose with the verification of the proposed model. In the last section we summarize the paper.

2 The Problem

Our research is based on a telemetric network designed for remote and automatic reading of media consumption meters. Meters of energy, water and gas transmit data to collecting nodes. In most cases wireless media is used; however, other experimental approaches are also tested (modulated transmission on AC power lines). Data from collecting nodes is sent to a local telemetric server. The data gathered in the telemetric server can be processed further, e.g. to predict media consumption based on historical data. To make such prediction possible, it is necessary to transfer the data from all of the telemetric servers (also called *data stream sources*) into the stream data warehouse. However, a data source can be also a single collecting node (not only a server). The difference in this case is that a collecting node is too simple device to buffer large amounts of data. A collecting node can be compared to a LAN switch, which only transfers data from one point to another. The data sources (collecting nodes, servers) are distributed geographically, what increases the probability of failures caused by external factors (e.g. local black-outs). Moreover, the data transmission process becomes a continuous ETL process.

The goal of the research is to assure continuity of the ETL process reading data from stream sources with the shortest possible delay between measuring and storing the data in the warehouse. If it is possible, we want to assure successful recovery of the interrupted processing without any stream data loss.

The system presented in Fig. 1 consists of : stream data sources (e.g. telemetric server or collecting node), remote stream buffers (RBF), remote persistent stream integrators (RIF), ETL modules, a module for error detection and stream integration (FTI). Our research is based on the following configuration: 4 independent stream sources, 6 RBF modules, 4 RIF modules, 3 ETL process replicas, an error detection module and a data warehouse server. The sources transmit data to RBF buffering modules, which communicate with RIF integrating modules offering persistent buffering. At this stage a replicated extraction[1] process appears. Outputs of the extraction process are connected to an FTI detecting module. The FTI is responsible for not loading of the improperly processed data (malformed during processing or transmitting). At the end of the modules chain there is a stream data warehouse and the systems using it. There are multiple connections between the modules. They are intended to provide redundant processing of all the data streams.

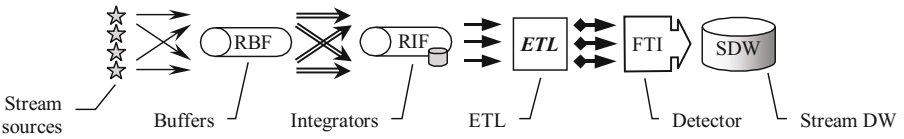

Fig. 1. Layered structure of the distributed system

2.1 Data Source Characteristics

A data source in our system is a single collecting node or optionally a local telemetric server. We assume that the source transmits data from associated media consumption meters and it has no ability to buffer any historical data. It means that there is no possibility to re-read already retrieved tuples from such a source. As a result, any interruption of the transmission from such a source leads to loss of a part of the data stream without any chance of recovering it. We assume that each source stamps the tuples keeping ascending stamp order. Consistency of stamping among all the sources in the system is not required; however, it is desirable for data analysis.

When a data source is a complex module such as a server, it can backup data received from meters on a disk to avoid data loss in case of transmission failures. Unfortunately, such a case is quite rare. To reduce system load simple collecting nodes are used. Then we have to assure that there is always a connection between the source and any receiver module, which is always ready to receive data incoming from the source. In our system RBF modules are used as receiver modules, and their buffers configuration is discussed in the following section.

[1] In this case extraction means a complete ETL process: Extraction + Transformation + Loading.

To configure the target RBF layer properly we have to prepare a mathematical description of such a source.

The analysed data source gathers measurements from particular meters (e.g. meters in blocks of flats, a large factory). The data can be trasmitted in one of the two modes: on-demand or asynchronous. In on-demand mode, the transmission is initiated by the collecting node and then selected meters sent out current values. This mode requires bidirectional communication between meters and a collecting node. It makes the communication easier to handle but unfortunately increases costs of such device. In asynchronous mode transmission goes in only one direction and meters can transmit data at any time. The collecting node (the source we analyse) must always be ready to handle incoming data stream.

2.2 Model Definition

Assume that there are N meters in the distributed system, and each meter transmits a reading approximately every T seconds. In other words it means that in each generation period (Fig. 2) each meter will transmit one reading. The probability that a meter transmits a reading in a particular time slot equals $p = t/T$. The total number of slots is $n_\mathrm{s} = T/t$. The question is: how many readings will be transmitted in any number of time slots of size t?

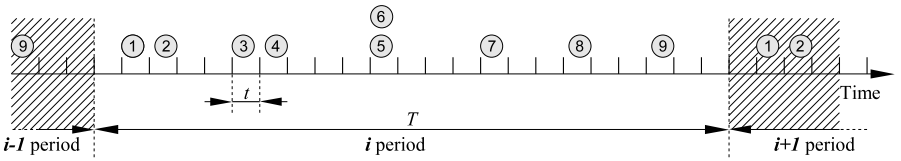

Fig. 2. Distribution of measures in a data source. The period T repeats.

Each meter is independent, so the reading events are independent also. Continuing the analysis presented in [18] the probability of k readings in any single slot t is $\binom{N}{k} p^k (1-p)^{N-k}$. Based on it we can define two recursive functions:

$$P_{=}(N, p, s_\mathrm{s}, k) = \begin{cases} \binom{N}{k} p^k (1-p)^{N-k} & \text{for } s_\mathrm{s} = 1 \\ \sum_{i=0}^{k} P_{=}(N, p, 1, i) \cdot P_{=}(N, p, s_\mathrm{s} - 1, k - i) & \text{for } s_\mathrm{s} > 1 \end{cases} \quad (1)$$

$$P_{\leq}(N,p,s_{\mathrm{s}},k) = \begin{cases} \sum_{i=0}^{k} P_{=}(N,p,1,i) & \text{for } s_{\mathrm{s}} = 1 \\ \sum_{i=0}^{k} P_{=}(N,p,1,i) \cdot P_{\leq}(N,p,s_{\mathrm{s}}-1,k-i) & \text{for } s_{\mathrm{s}} > 1 \end{cases} . \quad (2)$$

Both $P_{=}$ and $P_{\leq}$ functions define the probability that having N meters, in a sequence of s_{s} time slots we will observe exactly ($P_{=}$) or no more than ($P_{\leq}$) k readings. The results obtained for both functions are presented in Figs. 3 and 4.

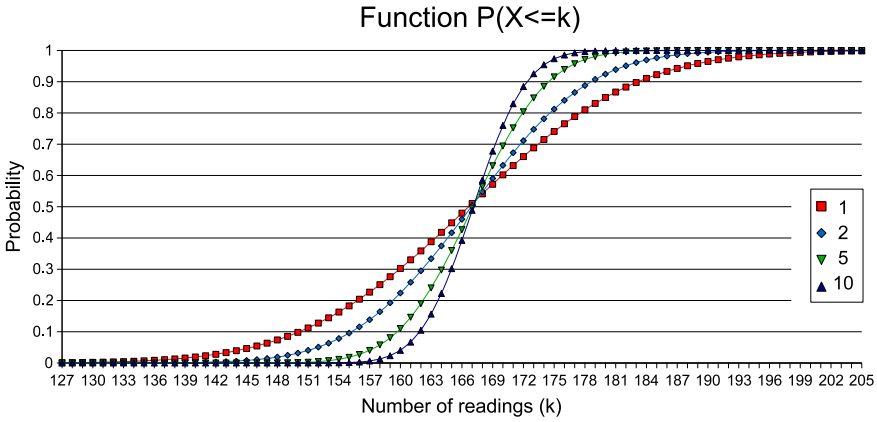

Fig. 3. Function $P_{\leq}$ for $N = 10^5$, $p = \frac{1}{600}$ and $s_{\mathrm{s}} = 1, 2, 5, 10$

Although the definition of functions $P_{=}$ and $P_{\leq}$ is correct, their evaluation in the abovementioned recursive form is extremely complicated. That is why we propose another approach. We can make use of the hypergeometric distribution. The easiest way to understand this distribution is in terms of urn models. Suppose you are to draw n marbles without replacement from an urn containing N marbles in total, m of which are white. The hypergeometric distribution describes the distribution of the number of white marbles drawn from the urn. It is defined as follows:

$$P_H(X=k) = f_H(k;N,m,n) = \frac{\binom{m}{k}\binom{N-m}{n-k}}{\binom{N}{n}} . \quad (3)$$

Our model requires to be modified a little. Unchanged remain: generation period T_{g}, number of time slots n_{s} and the number of meters N. In a single generation period we still can observe up to N readings, but it may happen that all readings

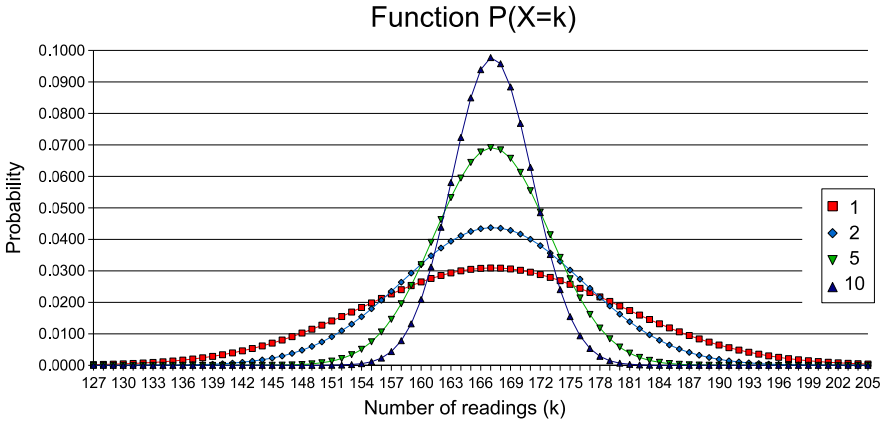

Fig. 4. Function $P_=$ for $N = 10^5$, $p = \frac{1}{600}$ and $s_s = 1, 2, 5, 10$

occur in a single time slot, and in the remaining $n_s - 1$ slots there are no readings at all. Assume that in a single generation period (urn) we have $N \cdot n_s$ events (marbles). We have N reading events (white marbles) and $N \cdot (n_s - 1)$ empty events (black marbles). We always draw $N \cdot s_s$ marbles. The probability of getting exactly k readings in a sequence of s_s time slots equals:

$$P_H(X = k) = f_H(k; N \cdot n_s, N, N \cdot s_s) \tag{4}$$

where $k \leq N$ and $0 < s_s \leq n_s$.

In Figure 5 the results obtained for P_H probability function are compared to the results for recursive $P_=$ function. The figure proves that both functions can be used interchangeably if needed. Moreover to simplify computations the hypergeometric distribution can be approximated using other distributions. Assume that $X \sim H(m, N, n)$[2] and $p = m/N$. Then:

1. If $n = 1$, then X is a Bernoulli distribution with parameter p.
2. If N and m are much greater than n, and p is not too close to 0 or 1, then $P(X \leq x) \approx P(Y \leq x)$, where Y has binomial distribution with parameters n and p.
3. If n is big, N and m are much greater then n, and p is not too close to 0 or 1, then

$$P(X \leq x) = \Phi\left(\frac{x - np}{\sqrt{np(1-p)}}\right) \tag{5}$$

where Φ is a cumulative distribution function of the standard normal distribution with parameters $\mu = np$ and $\sigma^2 = np(1-p)$.

[2] $H(m, N, n)$ defines hypergeometric distribution having probability function defined as $f_H(k; N, m, n)$.

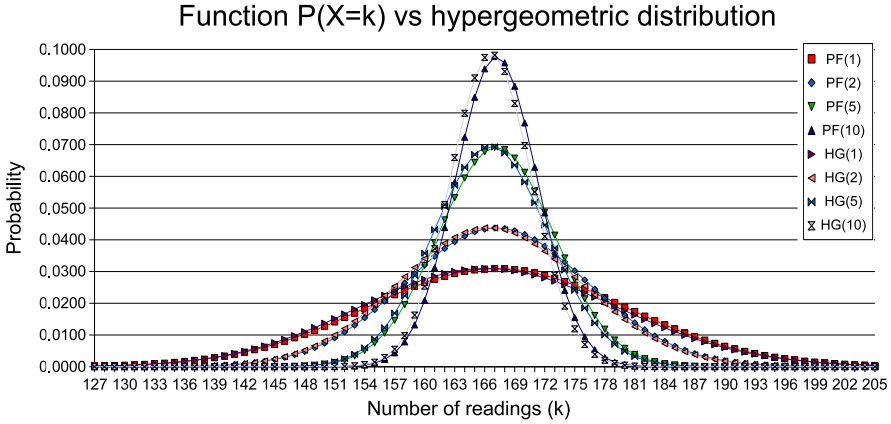

Fig. 5. Comparison of P_H and $P_=$ functions for $N = 10^5$, $p = \frac{1}{600}$ and $s_s = 1, 2, 5, 10$

Presented computations make it possible to evaluate the number of readings we can expect in a particular number of time slots if we have N meters, generation period T_g and a single time slot is t_s long. But how to compute the number of readings in case of various meter types and various generation periods?

To answer the question we need to divide all the meters into groups having the same generation period (Table 1). This way we obtain a set of meter groups Ψ, which elements are pairs $\psi_i = (N_i, T_{gi})$ for each $0 < i \leq |\Psi|$.

Table 1. Example of meter groups

Group	Meter type	Cardinality N	Generation period T_g	Std. dev. T_g
ψ_1	electricity	100 000	10 min	1 s
ψ_2	water	50 000	60 min	2 s
ψ_3	gas	50 000	24 h	5 s

For each ψ_i group we need to evaluate parameters of the distribution as described earlier in this section. This way each group will be described by the distribution of the random variable $X_i \sim N(\mu_i, \sigma_i^2)$. Knowing that the sum of any number of random variables having normal distribution still has a normal distribution we obtain a final distribution for all the meters:

$$X = \sum_{i=1}^{|\Psi|} X_i \sim N\left(\sum_{i=1}^{|\Psi|} \mu_i, \sum_{i=1}^{|\Psi|} \sigma_i^2\right) \quad . \tag{6}$$

2.3 Model Verification

To verify our model we conducted a few experiments. Firstly, we evaluated a theoretical distribution of the the groups of meters mentioned in Sect. 2.2. Secondly, we simulated the behaviour of our system and compared it with the theoretical results.

According to Equation (5) a computation of the distribution for each group of meters goes as follows:

$$\mu_i(s_\mathrm{s}) = np = n \cdot \frac{m}{N_H} = s_\mathrm{s} \cdot N_i \cdot \frac{N_i}{N_i \cdot n_\mathrm{s}} = s_\mathrm{s} \cdot N_i \cdot \frac{N_i}{N_i \cdot \frac{T_{\mathrm{g}i}}{t_\mathrm{s}}} = s_\mathrm{s} \cdot \frac{N_i \cdot t_\mathrm{s}}{T_{\mathrm{g}i}} \quad (7)$$

$$\sigma_i^2(s_\mathrm{s}) = np(1-p) = n \cdot \frac{m}{N_H} \cdot \left(1 - \frac{m}{N_H}\right) = s_\mathrm{s} \cdot \frac{N_i \cdot t_\mathrm{s}}{T_{\mathrm{g}i}} \cdot \left(1 - \frac{t_\mathrm{s}}{T_{\mathrm{g}i}}\right) \quad . \quad (8)$$

After substitution we obtain three groups with the following distribution parameters:

$X_1 \sim N(166.67; 166.39)$ $(\sigma = 12.9)$ for ψ_1 group,
$X_2 \sim N(13.89; 13.89)$ $(\sigma = 3.73)$ for ψ_2 group,
$X_3 \sim N(0.58; 0.58)$ $(\sigma = 0.76)$ for ψ_3 group,

The sum of the X_1, X_2, X_3 random variables gives in a result a random variable $X \sim N(181.13; 180.85)$ with $\sigma = 13.45$. The random variable X expresses the number of readings in a single time slot in a stream being the sum of three streams described in Table 1.

For the meter groups defined above we ran a simulation process. It started in the worst possible case in which all 200 000 meters generated the first measurement in the first second of work. Each meter works with a generation period T_g drawn according to the parameters of the group distribution. This way the moment of generation spreads in time as the simulation runs.

During the simulation we registered a histogram of the number of readings in a single slot. Based on it we prepared a plot of the probability function. In the first stage of the simulation histograms were collected for every 10 000 time slots. After exceeding 1 million of time slots histograms covered 100 000 slots (10 times more). Figure 6 presents the obtained simulation results compared with the theoretical curve. Simulation results where caught in the following points of time: 100 000 time slots, 200 000, 500 000, 1 million, 2 millions, 5 millions. As you can see for the first stage (100 000 slots) the probability function has quite irregular shape with at least three local peaks. At the beginning of the simulation all of the meters transmit the first reading in the first time slot. Further in the simulation the readings spread slowly according to their factory parameters distribution. We did not analyse the cause of the local peaks in details, but it results from the distribution of the factory parameters of the simulated meters. Going further the shape changes and after 1 million of slots it starts to look like a typical bell curve. Since then the simulation results are more and more similar to the shape of theoretical curve. The peak is the same as the computed above random variable $X \sim N(181.13; 180.85)$.

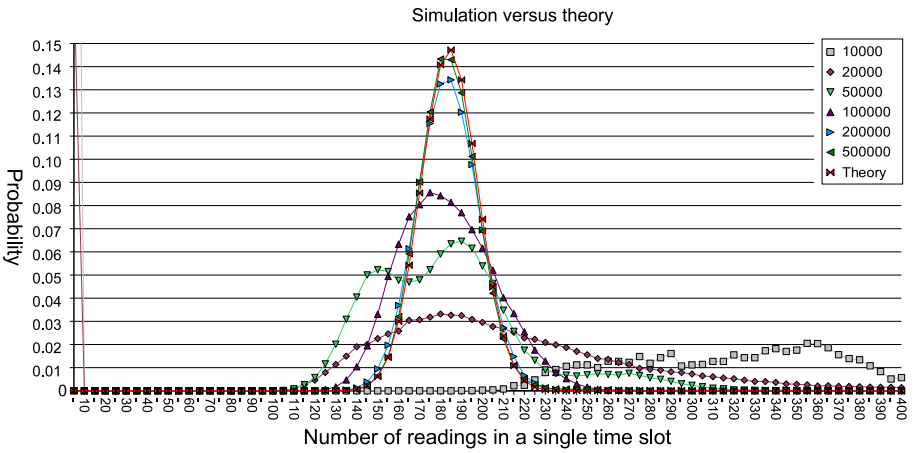

Fig. 6. Comparison of the theoretical readings distribution with the simulation results

We performed a similar calculation not for a single slot, but for a sequence of consecutive 30 times slots ($s_s = 30$). Then we rescaled the results for a single slot and the obtain results we gathered in Fig. 7. As can be seen the results for the sequence of slots are similar. The only difference is that after rescaling to a single slot, the expected number of meters in a single slot is more precise. The expected value remains unchanged and is still $\mu \approx 181.13$. But the standard deviation σ decreased from 13.45 to only 2.46. The final distribution for a sequence of 30 time slots can be computed from Equations (7) and (8). In this case it is: $\mu = 5434.03$, $\sigma^2 = 5425.58$ and $\sigma = 73.66$.

2.4 Edge Condition

Having the verified data distribution model we can try to answer the question stated at the beginning of Sect. 2.2: how many readings will be transmitted in any number of time slots t? We know the parameters of the random variable distribution. We know it is the normal distribution with parameters μ and σ. For normal distribution the density function is:

$$f(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \quad . \tag{9}$$

As the input parametr we take p_k denoting the possibility, that more than k readings will be received in the analysed period of time (number of time slots). We want to compute the smallest value of k for which the probability of receiving more than k readings is less than p_k. It leads to the following inequality:

$$p_k \geq \frac{1}{\sigma\sqrt{2\pi}} \int_{k+1}^{+\infty} e^{-\frac{(x-\mu)^2}{2\sigma^2}} dx \tag{10}$$

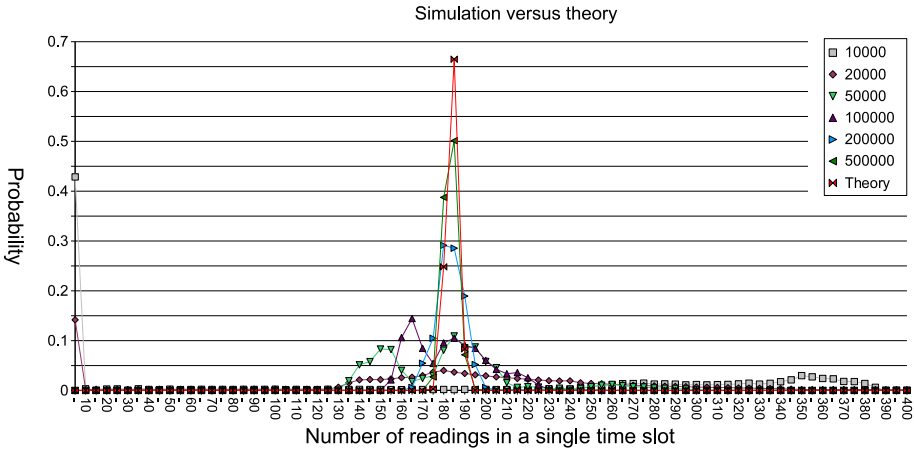

Fig. 7. Comparison of the theoretical readings distribution with the simulation results. Computation for 30 time slots scaled for a single slot.

The computed $k(p_k)$ is a base for setting the size of the buffers in RBF modules. Unfortunately there is no analytical solution for the Equation (10) due to the integration it includes. However, the result can be evaluated using numerical methods.

We can use the cumulative distribution function for the normal distribution based on the error function, after converting the distribution $N(\mu, \sigma^2)$ to the standard normal distribution $N(0, 1)$. Then we get:

$$F(x) = \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{x-\mu}{\sigma\sqrt{2}}\right)\right) \tag{11}$$

where $\operatorname{erf}(x)$ is the error function. The error function can be approximated using Taylor series:

$$\begin{aligned}\operatorname{erf}(x) &= \frac{2}{\sqrt{\pi}}\sum_{n=0}^{\infty}\frac{(-1)^n x^{2n+1}}{(2n+1)n!} = \\ &= \frac{2}{\sqrt{\pi}}\left(x - \frac{x^3}{3} + \frac{x^5}{10} - \frac{x^7}{42} + \frac{x^9}{216} - \cdots\right) \quad .\end{aligned} \tag{12}$$

In this case the solution is such a k, for which the following inequality is satisfied:

$$1 - p_k \geq \frac{1}{2}\left(1 + \operatorname{erf}\left(\frac{k-\mu}{\sigma\sqrt{2}}\right)\right) \quad . \tag{13}$$

3 Summary and Conclusions

In this paper we presented briefly the distributed stream processing system we work on. Our goal was to focus on the problem of modeling the intensity of

streams being processed in order to be able to configure system component properly. Firstly, we described data sources using mathematical rules. Basics of this description are included in [18–21]. Secondly, we tried to prepare a mathematical model which could be used to estimate the distribution of the number readings for a group of meters. Then the model was extended for various groups.

Obtained equations needed verification. We have built a simulation environment in which we conducted experiments. We measured how the number of readings in time periods changes. Based on the gathered data we were able to compare the simulation results with our theory. As described in the paper simulation results are very similar to the theoretical calculations.

Based on this we are able to estimate a safe buffer size, knowing the highest possible number of readings that can be received in a given time. Knowing the parameters of the distribution (σ and μ) and assuming the value of probability p_k small enough, we can find using the Equation (13) the number of readings that with probability p_k will not be exceeded.

This work lets us configure the second layer of the components network (Fig. 1) of our system. Further research needs to be done, to analyse behaviour of the other parts of the network. This is going to be the next stage of our research.

References

1. Wrembel, R.: On Handling the Evolution of External Data Sources in a Data Warehouse Architecture: Integrations of Data Warehousing. In: Taniar, D., Chen, L. (eds.) Integrations of Data Warehousing, Data Mining and Database Technologies, pp. 106–147 (2011)
2. Arasu, A., Babcock, B., Babu, S., Datar, M., Ito, K., Motwani, R., Nishizawa, I., Srivastava, U., Thomas, D., Varma, R., Widom, J.: Stream: The stanford stream data manager. IEEE Data Eng. Bull. 26(1), 19–26 (2003)
3. Gorawski, M., Chrószcz, A.: Synchronization Modeling in Stream Processing. In: Morzy, T., Härder, T., Wrembel, R. (eds.) Advances in Databases and Information Systems. AISC, vol. 186, pp. 91–102. Springer, Heidelberg (2013)
4. Gorawski, M., Chroszcz, A.: Optimization of operator partitions in stream data warehouse. In: Song, L.-Y., Cuzzocrea, A., Davis, K.C. (eds.) DOLAP 2011, pp. 61–66. ACM (2011)
5. Gorawski, M., Malczok, R.: Indexing Spatial Objects in Stream Data Warehouse. In: Nguyen, N.T., Katarzyniak, R., Chen, S.-M. (eds.) Advances in Intelligent Information and Database Systems. SCI, vol. 283, pp. 53–65. Springer, Heidelberg (2010)
6. Gorawski, M., Malczok, R.: Answering Range-Aggregate Queries over Objects Generating Data Streams. In: Kitagawa, H., Ishikawa, Y., Li, Q., Watanabe, C. (eds.) DASFAA 2010. LNCS, vol. 5982, pp. 436–439. Springer, Heidelberg (2010)
7. Gorawski, M.: Multiversion Spatio-temporal Telemetric Data Warehouse. In: Grundspenkis, J., Kirikova, M., Manolopoulos, Y., Novickis, L. (eds.) ADBIS 2009. LNCS, vol. 5968, pp. 63–70. Springer, Heidelberg (2010)
8. Kwiecień, A., Opielka, K.: Industrial Networks in Explosive Atmospheres. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2011. CCIS, vol. 160, pp. 367–378. Springer, Heidelberg (2011)

9. Kwiecień, A., Stój, J.: Genius Network Communication Process Registration and Analysis. In: Kwiecień, A., Gaj, P., Stera, P. (eds.) CN 2011. CCIS, vol. 160, pp. 314–321. Springer, Heidelberg (2011)
10. Balazinska, M., Balakrishnan, H., Madden, S., Stonebraker, M.: Fault-Tolerance in the Borealis Distributed Stream Processing System. In: ACM SIGMOD Conf., Baltimore, MD (2005)
11. Madden, S., Franklin, M.J.: Fjording the stream: An architecture for queries over streaming sensor data: ICDE. In: Proceedings of the 18th International Conference on Data Engineering, pp. 555–566. IEEE Computer Society (2002)
12. Gorawski, M., Malczok, R.: Distributed spatial data warehouse indexed with virtual memory aggregation tree. In: Sander, J., Nascimento, M.A. (eds.) STDBM, pp. 25–32 (2004)
13. Gorawski, M., Bańkowski, S., Gorawski, M.: Selection of Structures with Grid Optimization in Multiagent Data Warehouse. In: Fyfe, C., Tino, P., Charles, D., Garcia-Osorio, C., Yin, H. (eds.) IDEAL 2010. LNCS, vol. 6283, pp. 292–299. Springer, Heidelberg (2010)
14. Gorawski, M., Marks, P.: High efficiency of hybrid resumption in distributed data warehouses. In: DEXA Workshops, pp. 323–327. IEEE Computer Society (2005)
15. Gorawski, M., Marks, P.: Checkpoint-based resumption in data warehouses. In: Socha, K. (ed.) IFIP International Federation for Information Processing. Software Engineering Techniques: Design for Quality, vol. 227, pp. 313–323. Springer, Boston (2006)
16. Labio, W., Wiener, J.L., Garcia-Molina, H., Gorelik, V.: Efficient resumption of interrupted warehouse loads. In: Chen, W., Naughton, J.F., Bernstein, P.A. (eds.) SIGMOD Conference, pp. 46–57. ACM (2000)
17. Gorawski, M., Marks, P.: Fault-tolerant distributed stream processing system. In: DEXA Workshops, pp. 395–399. IEEE Computer Society (2006)
18. Gorawski, M., Marks, P.: Towards reliability and fault-tolerance of distributed stream processing system. In: DepCoS-RELCOMEX, pp. 246–253. IEEE Computer Society (2007)
19. Gorawski, M., Marks, P.: Distributed stream processing analysis in high availability context. In: ARES 2007: Proceedings of the The Second International Conference on Availability, Reliability and Security, pp. 61–68. IEEE Computer Society, Washington, DC (2007)
20. Gorawski, M., Marks, P.: Towards automated analysis of connections network in distributed stream processing system. In: Haritsa, J.R., Kotagiri, R., Pudi, V. (eds.) DASFAA 2008. LNCS, vol. 4947, pp. 670–677. Springer, Heidelberg (2008)
21. Gorawski, M., Marks, P., Gorawski, M.: Collecting data streams from a distributed radio-based measurement system. In: Haritsa, J.R., Kotagiri, R., Pudi, V. (eds.) DASFAA 2008. LNCS, vol. 4947, pp. 702–705. Springer, Heidelberg (2008)