Nr.: FIN-002-2019

# MStream: Proof of Concept of an Analytic Cloud Platform for Near-Real-Time Diagnostics using Mass Spectrometry Data

Roman Zoun, Kay Schallert, David Broneske, Sören Falkenberg,
Robert Heyer, Sabine Wehnert, Sven Brehmer,
Dirk Benndorf and Gunter Saake

Arbeitsgruppe DBSE

Fakultät für Informatik
Otto-von-Guericke-Universität Magdeburg

Technical report

Nr.: FIN-002-2019

# MStream: Proof of Concept of an Analytic Cloud Platform for Near-Real-Time Diagnostics using Mass Spectrometry Data

Roman Zoun, Kay Schallert, David Broneske, Sören Falkenberg,
Robert Heyer, Sabine Wehnert, Sven Brehmer,
Dirk Benndorf and Gunter Saake

Arbeitsgruppe DBSE

Technical report (Internet)
Elektronische Zeitschriftenreihe
der Fakultät für Informatik
der Otto-von-Guericke-Universität Magdeburg
ISSN 1869-5078

Fakultät für Informatik
Otto-von-Guericke-Universität Magdeburg

**Impressum** (§ 5 TMG)

*Herausgeber:*
Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Der Dekan

*Verantwortlich für diese Ausgabe:*
Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Roman Zoun
Postfach 4120
39016 Magdeburg
E-Mail: roman.zoun@ovgu.de

http://www.cs.uni-magdeburg.de/Technical_reports.html

Technical report (Internet)
ISSN 1869-5078

*Redaktionsschluss*: 16.08.2019

*Bezug:* Otto-von-Guericke-Universität Magdeburg
Fakultät für Informatik
Dekanat

# MStream: Proof of Concept of an Analytic Cloud Platform for Near-Real-Time Diagnostics using Mass Spectrometry Data

Roman Zoun *
University of Magdeburg
roman.zoun@ovgu.de

Kay Schallert
University of Magdeburg
kay.schallert@ovgu.de

David Broneske
University of Magdeburg
david.broneske@ovgu.de

Sören Falkenberg
University of Magdeburg
soeren.falkenberg@ovgu.de

Robert Heyer
Max Planck Institute for Dynamics of Complex Technical Systems
heyer@mpi-magdeburg.mpg.de

Sabine Wehnert
University of Magdeburg
sabine.wehnert@ovgu.de

Sven Brehmer
Bruker Daltonik GmbH
sven.brehmer@bruker.com

Dirk Benndorf
Max Planck Institute for Dynamics of Complex Technical Systems
benndorf@mpi-magdeburg.mpg.de

Gunter Saake
University of Magdeburg
gunter.saake@ovgu.de

## ABSTRACT

A mass spectrometer is a device to extract biomarkers of biological environments. Using these biomarkers, it is possible to diagnose thousands of diseases with only one mass spectrometer. Unfortunately, the mass spectrometry pipeline is sequential, including hours of waiting time between the workflow steps. Additionally, the data analysis is complex and needs qualified employees and a stable infrastructure, which involves very high costs and effort. Hence, only few hospitals use a mass spectrometer for diagnostics with success.

In our work, we present a proof of concept of an analytical platform for real-time analysis of mass spectrometry experiments. In collaboration with Bruker Daltonik GmbH, we implemented MStream, a cloud-based platform on the SMACK stack (Spark, Mesos, Akka, Cassandra, Kafka) for scalable, streamlined protein identification. Our evaluation shows superior performance in comparison to the state-of-the-art X!Tandem software package. Additionally, we minimize the effort of the hospital by allowing the full analysis pipeline to be outsourced to our cloud platform.

## CCS CONCEPTS

• **Computer systems organization → Embedded systems**; *Redundancy*; Robotics; • **Networks** → Network reliability.

## KEYWORDS

mass spectrometry, SMACK stack, fast data, cloud computing, real-time diagnostic

## 1 INTRODUCTION

Mass spectrometers are increasingly successful devices that digitize real world samples. Mass spectrometry is used in the research fields proteomics, metaproteomics and metabolomics [7] to identify protein biomarkers in biological environments, such as oceans, humans, or microbial communities. These biomarkers are similar to a fingerprint and can be used to identify viruses or bacteria [3]. Thus, mass spectrometers can support the diagnosis of known diseases such as cancer, Alzheimer's disease, and even lupus [16, 20, 21]. However, only few university hospitals use a mass spectrometer for diagnostics with success.

Due to frequent quality upgrades, mass spectrometers produce ever-increasing amounts of data, resulting in terabytes of output data from a single device. This data alone is useless without the analysis and processing to derive the necessary insights. Due to the huge data sizes and the complexity of the algorithms, the current sequential analysis pipeline takes hours to complete: The mass spectrometer needs two hours for the measurement, followed by a conversion step of up to one hour and an additional analysis step that takes several hours to complete [8]. Additionally, qualified staff has to maintain the software infrastructure (servers and special software for the data analysis) in the hospital.

The state-of-the-art software tools for the downstream analysis, such as X!Tandem, Andromeda or Mascot [1, 5, 22], work with file-based input data. Accordingly, the algorithms are specialized to process the complete data in a bulk processing fashion. Hence, state-of-the-art data analysis (protein identification) needs all the experimental data at once. As a consequence, the analysis step, which itself takes hours, is further delayed by another several hours, because it has to wait until all the measurement data is available. While these delays are tolerable in many research applications, in clinical diagnostics they are not. Clinical diagnostics require an analysis in real-time, so that the data can be processed during the measurement. Among other benefits, such a real-time analysis could be stopped if a specific result (specific disease such as cancer) is identified, thus reducing the overall time of the diagnostics.

In our work, we focus on a tandem mass spectrometer from Bruker Daltonik GmbH and present the proof of concept of MStream, an analytic cloud-based platform to process mass spectrometer data in near-real-time. This paper shares the challenges and experiences building the platform. Our MStream prototype is deployed on a SMACK stack, a fast data implementation and works on single data items of the mass spectrometer. MStream provides an asynchronous analysis pipeline, performing the protein identification process as the measurement data arrives from the device. Evaluation

Authors' addresses: Roman Zoun, University of Magdeburg, roman.zoun@ovgu.de; Kay Schallert, University of Magdeburg, kay.schallert@ovgu.de; David Broneske, University of Magdeburg, david.broneske@ovgu.de; Sören Falkenberg, University of Magdeburg, soeren.falkenberg@ovgu.de; Robert Heyer, Max Planck Institute for Dynamics of Complex Technical Systems, heyer@mpi-magdeburg.mpg.de; Sabine Wehnert, University of Magdeburg, sabine.wehnert@ovgu.de; Sven Brehmer, Bruker Daltonik GmbH, sven.brehmer@bruker.com; Dirk Benndorf, Max Planck Institute for Dynamics of Complex Technical Systems, benndorf@mpi-magdeburg.mpg.de; Gunter Saake, University of Magdeburg, gunter.saake@ovgu.de.

results show that MStream compares favorably to the state-of-the-art software X!Tandem in terms of scalability and performance.

This paper is structured as follows: In Section 2, we explain the basics of the mass spectrometry workflow and the architecture of our platform. In Section 3, we describe the challenges and solutions of the four components in our MStream system. In Section 4, we present our implementation. Furthermore, we evaluate the whole pipeline, showing its scalability and comparing it to the state-of-the-art tool X!Tandem. In Section 5, we discuss the results of the evaluation. Finally, Section 6 concludes our paper.

# 2 BACKGROUND

In this Section, we provide some basic knowledge about mass spectrometry data processing. First, we clarify the state-of-the-art of the mass spectrometry workflow. Afterwards we present the fast data architecture, followed by an explanation of the data processing steps for protein identification and validation.

## 2.1 Mass Spectrometry Workflow

As mentioned, the mass spectrometry field deals with analysis of protein biomarkers. The workflow is sequential and the smallest parallelizable unit is the whole experiment itself. Figure 1 shows the experiment pipeline.

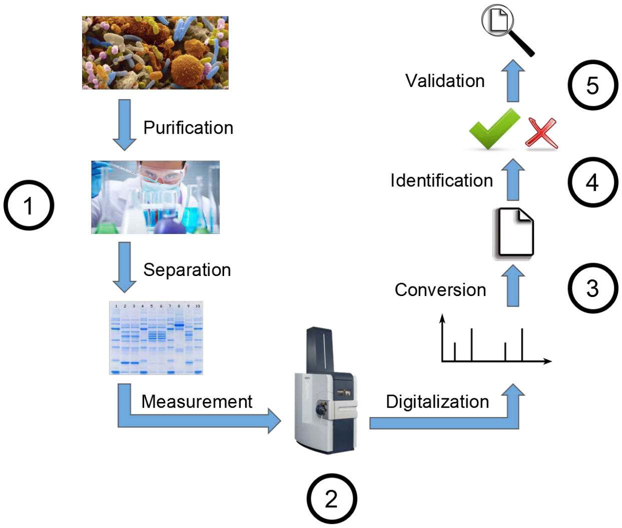

**Figure 1: State-of-the-art workflow of mass spectrometry experiments with the biological preparation (1), the measurement of the mass spectrometer (2) and the data processing pipeline (3-5) [12–14].**

After the sample is collected, the proteins are extracted (Figure 1–1). This preparation is done in a laboratory and is not the focus of this work [6]. For clinical sample preparation these steps are automated. Further the prepared sample is measured in the mass spectrometer (Figure 1–2). The digitized data is collected in a manufacturer specific RAW format. The duration of the measurement is between one and two hours and results in a file of up to 40 GB, containing around 200,000 mass spectrometry datasets, the so called mass spectra [17]. In the next step the RAW data is converted into a standard file format (Figure 1–3). During the conversion, several pre-processing steps, such as noise reduction and pre-filtering are executed in order to increase the quality of the data for the further analysis. This step takes up to 1 hour of the processing time. In the next step (Figure 1–4), the data is compared to a sequence database such as UniProtKB or user defined sequence data. The knowledge base is usually in a fasta file format [19]. The validation of the hits is the last step of the process2 (Figure 1–5). Further analysis such as visualizations, conclusions by physicians or biological researchers is done based on those identification results [6, 7].

The central component of the data processing pipeline is the protein identification step Figure 1–4. In the next Section, we explain the basic algorithm and one state-of-the-art implementation.

## 2.2 Protein Identification

Since the incoming experiment data is a digital signal from real world proteins, it is without any meaning and the data has to be identified compared to a protein knowledge base. Therefore, the protein identification approach is used to identify the measured spectra [18]. First, we explain the general protein identification approach, followed by a state-of-the-art implementation of the method.

*2.2.1 Concept of Protein Identification.* The protein identification compares the real world data, which is represented as measured mass spectra from the mass spectrometer and the theoretical data from the protein sequence database, which contains already known real world proteins [18].

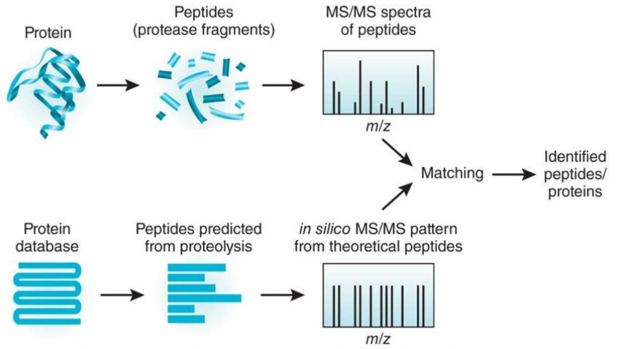

**Figure 2: A general peptide centric protein identification method, which compares the experimental spectra and the theoretical ones [18].**

In Figure 2, on the upper side the biological preparation of the sample is represented. After the purification of the sample from a patient, only proteins are left. Protein data consists of a protein sequence and the meta information. The protein sequence is split (digested by specific enzymes) into peptides. The enzymes cut the protein sequence on specific places in the sequence. The mass spectrometer measures the peptides and generates their digital signal, which represents the experimental spectra. The lower side shows the same procedure for the known proteins. The protein sequences from a protein sequence database get split into peptides, using similar splitting rules and reconstructing each peptide into theoretical spectrum. This ensures the comparability of the data. The matching is then between each peptide (theoretical spectrum) and each experimental spectrum (peptide from the experimental sample), that results in a peptide-spectrum-match (PSM). Due to the unsteady digital signal during the measurement and the noises, the experimental spectrum can never be as good as the theoretical one, so the comparison ends up in low similarity score. The matching is similar to the comparison of a photography and a clip-art. In conclusion, the scoring needs to be validated for true matches or false matches with the idea that wrong hits have even lower score [18].

*2.2.2 Protein Identification in X!Tandem.* Currently, many protein identification software tools exist, such as Mascot or X!Tandem [1, 22]. Mascot is a proprietary software from the company matrix science and X!Tandem is an open source software, which is popular in the omics community and provides very fast processing of the data. Both rely on the sequential data processing pipeline. In this section, we present the processing of the data in X!Tandem, because we could analyze and measure the process in the free available source code.

The protein data comes from the hard disk, while the mass spectrometry data is in the main memory. The software iterates once over the protein database and multiple times over the spectra. The following steps are carried out: firstly the software loads the experimental data at once, filtering out low quality data, based on user-defined parameters. Secondly, the software processes data from files batchwise. Thirdly, X!Tandem splits each protein in smaller peptides and fourthly, the algorithm compares them with the experimental spectra. Lastly, the best peptide-spectrum-matches are stored in the end of the process in an XML-based result file [22]. The identified PSMs need a validation to ensure the quality of the results.

## 2.3 Target Decoy Validation

The validation method processes each peptide-spectrum-match (PSM) and tests whether the PSM is a true positive or false positive match. Since every experimental data is individual, the state-of-the-art validation uses a target-decoy approach [2]. The idea is to allow only small amount of wrong hits in the result, using approximation of the false discovery rate (FDR).

In this method, the identification of experimental spectra against the knowledge base results in target PSMs. Additionally, a created wrong knowledge base is used to identify the experimental data again to produce decoy PSMs. Later, the target and the decoy PSMs get stored in a collection, sorted by the similarity score in descending order. Afterwards, the bottom PSMs are iteratively removed until the desired false discovery rate is reached, using the formula #Decoy PSMs/#Target PSMs for the FDR calculation in every step. All removed target PSMs are false positives, the rest of the target PSMs are valid hits.

A drawback of this approach is that it needs the whole experimental data at once and is not applicable to our prototype, which requires real-time processing.

## 2.4 Fast Data Architecture

To transform the mass spectrometry workflow into a streaming asynchronous pipeline, we adapt it to the fast data architecture. The fast data architecture is a technology pipeline to process possibly infinite data streams in near-real-time, using state-of-the-art big data technologies [23]. A popular fast data implementation is the SMACK stack.

A general fast data architecture is designed as following: incoming data enters directly as messages via HTTP or socket directly from a device or microservice . A distributed cluster manages the incoming data . The data stream is ether directly stored or analyzed by cloud processing engine . The processing components communicate with the persistence layer . The whole system runs on a cloud operating system in order to schedule the components [23].

# 3 MSTREAM: CLOUD-BASED MASS SPECTROMETRY DIAGNOSTIC PLATFORM

The mass spectrometry workflow consists of three main parts - the preparation of the experiment (Step 1 in Figure 1), the measurement (Step 2 in Figure 1) and the data processing (Steps 3, 4 and 5 in Figure 1). In our work, we focus on the data processing pipeline, because it is the bottleneck of the sequential workflow. Our approach starts directly with the data digitization step and parallelizes the measurement and the data processing steps, bringing the fast performance of cloud-based data processing to a mass spectrometer. In this section, we describe the challenges of the new MStream system and how we solved them.

## 3.1 Challenges of MStream

For a better visualization of the challenges, we show in Figure 3 the concept of our system. Our MStream platform needs a stream producer (Figure 3–1), horizontally scalable scorer (Figure 3–2), distributed structured protein knowledge base (Figure 3–3) and a smart validator of peptide-spectrum-matches (Figure 3–4).

The challenge of the stream producer is to stream the mass spectrometry data during the measurement without blocking the current measurement process and converting the RAW data into a readable format. These steps process each spectrum individually before sending it to the cloud-based MStream system. The scorer, in turn, has to scale out horizontally to perform near-real-time analysis even with high throughput or future devices with higher resolution. For performance reasons, it is impossible to traverse all peptides every time a spectrum arrives in the system. Therefore, a smart indexing and aggregation technique is needed to reduce the search area of the peptides. We use the criteria of current protein identification tools to define an index for the sequence database. Furthermore, the smart validation method should allow the MStream system to validate every PSM individually, but without compromising on quality. Finally, the personal data security is very important, especially for a remote cloud system.

The stream producer, the concept of protein data structuring and the validation were already presented in prior work and we will give only an overview of the results. The personal data security is solved by mapping the experimental data into the stream producer to a specific patient without revealing or sending any personal data to the cloud. The contribution of this work is the scalable scorer and the overall evaluation of the MStream analytic platform. In this Section, we will describe our solution for the challenges, beginning with the validation, followed by the stream producer and the persistence layer. Finally, we will describe the identification process and the MStream system.

## 3.2 Streaming Validation of Peptide Spectrum Matches

The validation is needed to trust the results but the state-of-the-art target-decoy method is not applicable to streaming experiment data. For streaming validation method, the goal is to classify a peptide-spectrum-match (PSM) as valid or non-valid directly as they are generated. We solved this classification problem using logistic regression, a machine learning technique. In prior work, we showed that our solution can speed up the identification by a factor of 1.8. At the same time, we reach an accuracy of over 95%, which is enough for the most uses cases [9]. Additionally, the classifier is independent from the experiment; it only has to be trained once for a given device. As a result, our classifier helps us overcome the first challenge towards a stream-based analysis workflow.

## 3.3 MSDataStream: Stream Producer for a Bruker Mass Spectrometer

Since every manufacturer has its own file format to store the measured data, we collaborated with the Bruker Daltonik GmbH to access the data. We adapted the given API to read the data during the measurements in batches without blocking the actual measurement. Furthermore, we added pre-processing steps to increase the quality and reduce the noise of the data. While in the current pipeline, the conversion into a readable format begins after the measurement, we added the conversion into the stream producer to transform each data, which we send to the cloud or write to a file. The application runs locally on the mass spectrometer computer and collects the data continuously. Further adapters for manufacturers are planned for the future work and need additional collaborations with device companies [11].

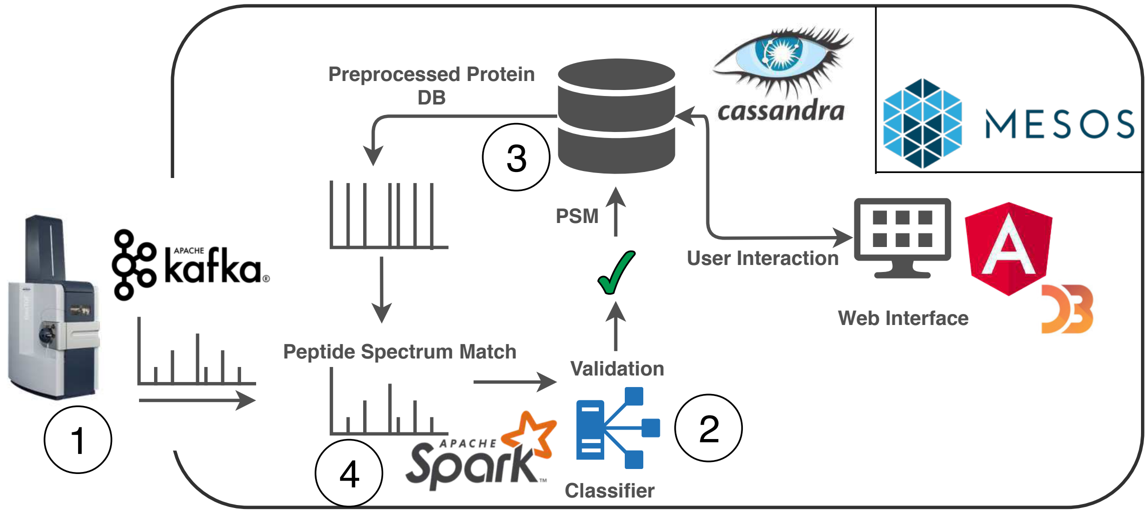

**Figure 3: Architecture of the MStream prototype, the analytic platform for real-time diagnostic of mass spectrometry data**

## 3.4 Preparation of Protein Data

In order to implement the streaming identification process, we analyzed other tools for protein identification. In Section 2.2, we described how local software tools identify the experimental data, and highlighted two limitations. The first requirement is that all the data should be available at once and the second one is that the experimental data should be small and fit into the main memory. In our system, the input data is not completely available and we can access only a few spectra at the same time. Accordingly, in our system, we iterate once over the mass spectrometry data and have to access the protein and peptide sequence data for each spectrum. Due to the data size of the protein knowledge base, we reduce the searching area, using sorted index on the total mass of peptides, based on the work of Juergen Cox et al. [5].

In our system we implement a radix tree as transformation object to transform the sequence protein data into a column-based schema, using features of the data as index family [10]. Our storage schema of the protein data is specialized to reduce the amount of comparisons for the diagnostic use case, selecting only data in a specific mass range. The index structure is column-based, such as the one used by Apache Cassandra or in ELF [4, 15]. The schema of MStream has additional components for user management, data management and results.

## 3.5 Online Processing of Streaming Spectra

The core of the identification process is the scoring function. It is implemented in the Scorer component, which unites all other components of the system. State-of-the-art tools process the experiment data in one bulk during the protein identification step. In our architecture the experimental spectra are not available completely, but arrive periodically in the cloud. In the Scorer, the following components are required: the stream producer (Section 3.3), the validator (Section 3.2) and the storage (Section 3.4).

Since we have to serve multiple clients in parallel, we add user-dependent parameters to each incoming message. In this way, we separate each spectrum and use one message channel for the experimental data, making the spectrum message the smallest parallelizable unit in the system. The parameters define the protein data for the identification and the validation model.

The algorithm that each worker thread executes, is as follows: first, the worker collects a batch of messages. Each message contains the textual representation of the spectrum. Next, each spectrum of the batch is processed individually. For each spectrum, the algorithm selects all peptides that must be considered for a comparison from a specific protein knowledge base. Hence, each peptide in the knowledge base is compared to the spectrum and the match with the best score is kept as a result. Using our logistic regression classifier, the best peptide-spectrum-match is validated. Finally, the PSM and the spectrum are stored in our storage system.

Following the X!Tandem protein engine software, our scoring function is based on the weighted dot product. In the future , other similarity functions can be implemented in our system.

## 3.6 Putting It All Together

The state-of-the-art tools need the experimental data as a file and the protein data as a file, while our system has additional requirements on the technical side and on the infrastructure level. When the data processing starts, a protein knowledge base must already be loaded into memory (see Section 3.4). Consequently, the protein database must be uploaded and prepared for further processing in our system. Furthermore, we need a trained model for the specific device that enables validation of the experimental data (see Section 3.2). Finally, our stream producer must be installed on the mass spectrometer side (see Section 3.3).

The system is deployed on the SMACK stack, an implementation of the fast data architecture (see Figure 3). Additionally, a lightweight web UI displays the results of an experiment during the identification process.

In Figure 4, we show the communication between the components and how they operate in a sequence diagram. The stream producer asks for new data periodically and starts to collect the data as the mass spectrometer begins with the measurement. The producer processes and transforms every experimental spectrum and sends the data to the message service. The MStream worker with the scoring task subscribes to the message service and asks for new data periodically. The subscription time of the scorer defines in which time periods the system asks for a new data batch from the message service. For the protein identification the scorer queries peptides from the protein knowledge base and stores the results.

# 4 EVALUATION

From a biological perspective, our system provides the same results as other protein search engines, since the similarity function and the pre-filtering are based on the state-of-the-art approach. Hence, in our work, we focus on the evaluation of the performance.

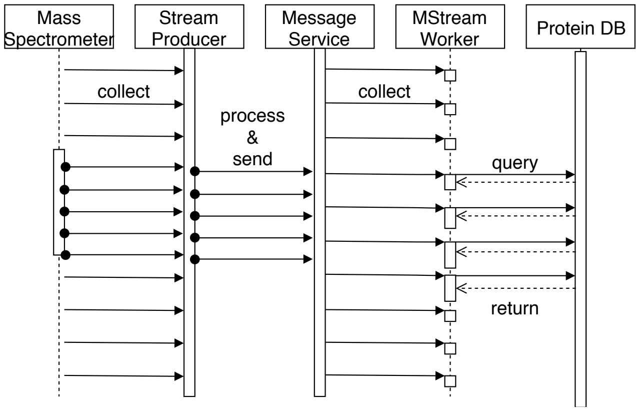

**Figure 4: Sequence Diagram of MStream components.**

This section is structured as follows: firstly, we present the evaluation setup of the prototype. Secondly, we describe the experiments and, finally, we present our results.

## 4.1 Evaluation Setup

In our work, we implemented the MStream system prototype, which we use for the evaluation. We use a de.NBI OpenStack system to run 15 virtual machines, using Harshi Corp Terraform for fast management of the infrastructure. Overall, our OpenStack project offers 160 virtual CPUs and 720GB RAM. We use Apache Mesos as a cloud operating system. On Mesos, we deploy all the services and components of MStream, such as Apache Cassandra for the structured storage, HDFS to store our trained validation models, Apache Spark for processing the data and scoring and Apache Kafka as a message broker. For additional services, we use the Spring framework and SparkJava on a Jetty application server. The stream producer software is a JavaFX implementation and runs on the machine of the mass spectrometer. For the state-of-the-art tool we use a local computer. As a local machine, we use an ASUS UX360 with Intel I7-7500U/BGA with 2 cores and 16GB DDR3-RAM.

For the evaluation data, we use PASEF experiments of ecoli bacteria with 57,758 spectra (5.37GB) to evaluate the scalability and to compare to the state of the art. The data is generated by a PASEF TIMSTOF mass spectrometer from Bruker Daltonic GmbH. As knowledge base of proteins, we adopt the commonly used UniProt SwissProt knowledge base, containing 556,196 proteins, which result in 23,934,321 peptides. The chosen knowledge base contains much more proteins than necessary for diagnostics of specific disease, because we test the performance and the impact of the amount of proteins to the platform. In the next Section, we describe the evaluation of our MStream platform.

## 4.2 Evaluation Experiments

The goal of the evaluation is to show the applicability for near-real-time identification that is needed for medical diagnostics. MStream performs in near-real time if the overall processing rate in amount of spectra per second is the same as the measurement rate of the mass spectrometer in amount of spectra per second. Hence, we first analyze the persistence component (see Section 3.4) regarding the data size that we create, using our index approach. Additionally, we analyze how the indexing reduces the search range and how fast our system queries the data from the persistence layer.

As we propose a central system, we observe if our system can handle multiple devices in the same time, providing real-time analysis. Firstly, we analyze the scoring time by measuring the performance of each scorer, secondly, each step in a scorer and thirdly, the scalability factor of the system.

Finally, we compare our system to the state-of-the-art protein identification software X!Tandem to evaluate the competitiveness of MStream.

*4.2.1 Evaluation 1: Structured Protein Knowledge Base.* In this Experiment, we present the results regarding the knowledge base component (see Section 3.4). After uploading the data, in this case the SwissProt data, the amount of data changes because we remove duplicates on the protein level and on the peptide level. This reduces the amount of peptides from 23,934,321 to 14,579,004 non-redundant peptides. Due to the pre-calculation of the charge property and the modifications for each peptide, the amount of the peptides in the pepmass table rises to 111,183,434. Hence, our system increases a 500MB protein sequence database to 17GB on hard disk.

All the peptide sequences in our storage are precalculated and indexed on their mass and charge properties to reduce the search area and increase the performance of the analysis step. The precalculation results in 111,183,434 peptide sequences stored in 4,814,243 rows, grouped by the charge and by the total mass. The amount of peptides with charge 3 are in the range of 170 and 1300 dalton . The peptides with charge 2 are in the range of 300 and 1950 dalton, while data with charge 1 is between 600 and 3700 dalton. Grouping by the charge attribute distributes the data across the rows. Nevertheless, the amount of peptides is not equal, which results in unbalanced query results. Of course, the amount of data returned by the query for each spectrum has an influence on the runtime. We analyze this influence next.

We test the query time to select the data with different tolerances on their masses. The error tolerance is calculated in parts per million (ppm) and depends on the given device. The range of 100 ppm can be used for older devices, 20 ppm is for current devices, 10 ppm is for current high class devices and 1 ppm is for future devices with even higher precision.

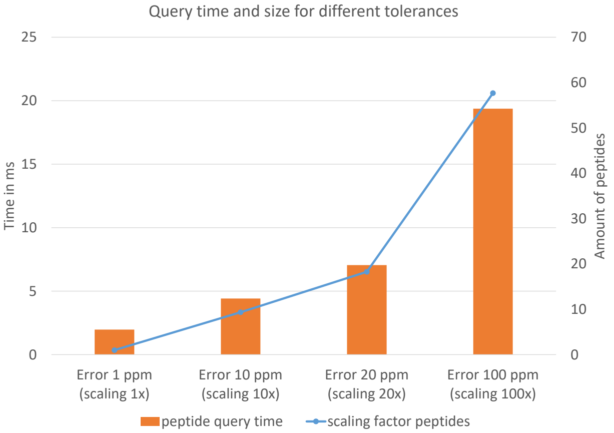

**Figure 5: Query time and the query size regarding different error tolerances.**

In this experiment the data input rate is 189,42 experimental spectra per second, which describes the rate of six to seven mass spectrometers based on our experience with the Bruker mass spectrometer (On average 27 spectra per second). In Figure 5, we show the query time on the left y-axis (orange bars) and the amount of the peptides on the right y-axis (blue line), selected when querying for different tolerances. The evaluation is an average of 50 runs and measures the time for each spectrum. For 1 ppm, the query amount of rows that are selected is 12 on average, the amount of peptides is around 250 and the query is performed in 2 ms. Using 100 ppm, the query time increases to 19 ms, the amount of peptides grows to 14,420 selected out of 818 rows in average. For 10 ppm the query needs 4.5 ms in average and 2,337 peptides are selected, while for 20 ppm the query took 7 ms with 4569 peptides on average. While the scaling factor from 1 ppm to 100 ppm is 100, the factor for the query time for these experiments is only 9.9, and the factor for the peptides is 57.7.

We show the influence of the tolerance to the query time and propose to optimize this parameter for specific real world scenario. Especially for repetitive measurements, such as diagnostics of a disease it is beneficial to use the minimum tolerance, based on experience of the measurements. This will reduce the calculation time and the costs for the analysis.

*4.2.2 Evaluation 2: The Scorer Performance.* In this section, we evaluate the worker (the scorer) scalability and the performance of the MStream system. Our goal is to analyze the possibility to operate with multiple mass spectrometers and provide the results in near- real-time for every user.

To this end, on the one hand we increase the spectra input rate, which will increase the amount of spectra per batch. On the other hand, we increase the error tolerance, which will increase the amount of peptides. This experiment runs with the error tolerance of 20 ppm and a subscription time of 10 seconds.

In the first experiment of the scorer evaluation, we use different spectra input rates to simulate multiple devices and users. For each run, we measure the time of each step of the scorer. The first step is the query of the peptides, the next step is the scoring of the spectrum and the peptides, the third step is the validation of the PSMs, followed by the insert functions of the PSM and the spectrum. All runs achieve real-time processing with the time deviation of the processing time of the last mass spectrometry data, usually a few seconds. In this experiment, we increase the input rate and simulate multiple devices, sending data at the same time. Our goal is to scale out the worker to serve each user's device in real-time. Therefore, we analyze how the system adapts to additional mass spectrometers, which reflects the individual performance of the worker and the overall performance of the system.

In Figure 6, we present the results of this experiment. The measured time is the average value in ms per worker and per spectra. We evaluate our system with simulated input rates 41, 104, 196 and 727 spectra per second on average. Due to the fact that one mass spectrometer sends in average 27 spectra per second, which can be processed by a single worker, the input rates are generated using already measured experiments processing the data with the stream producer (see Section 3.3). We measured the time for each part in the algorithm to analyze the differences between different input rates. Additionally, we measure the minimum number of workers that are needed to perform the analysis in near-real-time.

In the chart in Figure 6, we can observe that the input rate does not influence the single processing steps of the worker, but influences the amount of workers. The individual performance deviation is around 1 ms. We attribute this deviation to the scaled-out workers and the cloud latency. Additionally, we can observe that a worker scales linearly with the input rate. The processing speed of one worker with the parameters of these experiment (using tolerance with 20 ppm) processes around 45 spectra per second. Increasing this rate, it scales linearly without influence on the individual worker performance. To the end, scorer spends most of its time in the pairwise comparison.

The different sections of the algorithm are summed up to show the overall time for one spectrum. The growing input rate increases the amount of spectra in one batch, thus increasing the amount of comparisons of the worker, but not the time of the single spectrum analysis. This bottleneck is handled with out scaling the workers. Hence, the processing time of one batch is increased and leads to longer waiting time of the last batch of the mass spectrometer.

In the next chart, we show the performance of one worker, regarding the error tolerance. This experiment we ran with an input rate of around 200 spectra per second in each run and a subscription time of 10. The input rate simulates high utilization, which usually needs around 5 workers to perform in real-time. In Figure 7, we show the average performance of a single worker as a dependant of the error tolerance. For 1 ppm the scorer is faster, since the amount of peptides is decreased. Additionally, the scorer performs very fast with 335 spectra per second for one worker. However, for 100 ppm one worker processes 15 spectra per second and one worker cannot perform in real-time anymore. Hence, the scorer scales out up to 17 workers to reach near-real-time processing speed.

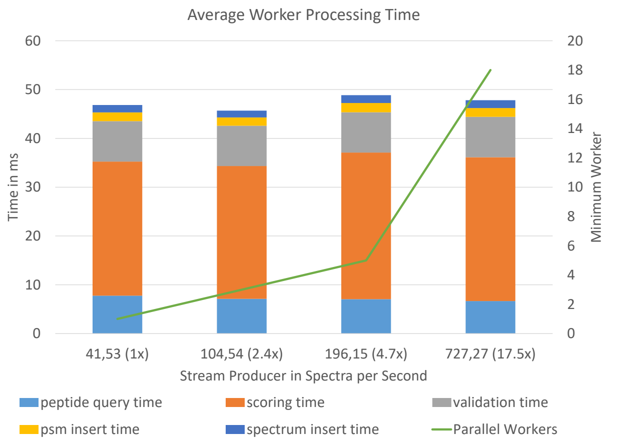

**Figure 6: Average runtime in ms depending on the data input rate.**

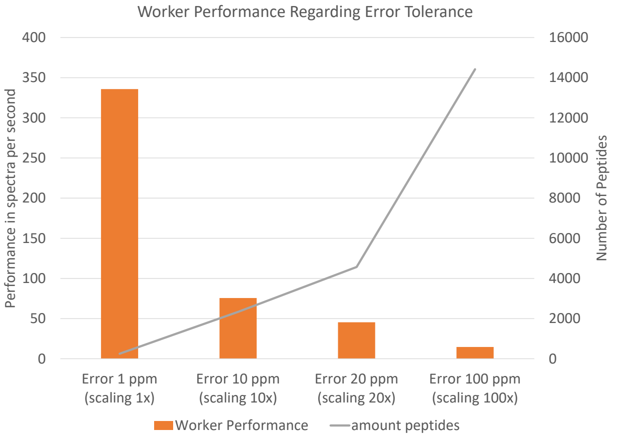

**Figure 7: Average worker performance in spectra per second by increasing the error tolerance for the peptide query in MStream.**

For the next chart in Figure 8, we analyze the influence of the tolerance of the single steps on the scorer. Again, we ran the experiments with different error tolerance, measuring the time and related the scaling factors to the amount of peptides. We can see that, the time for insertion and validation does not change but the tolerance influences the amount of peptides. Accordingly, the time for querying the peptides and scoring increases. The scoring time scaling factor is 39 at 100 ppm. Hence, the scoring scales slowlier than the tolerance regarding the scaling factor. This experiment shows that the influence of the input rate is much higher with the weight of 1 than the error tolerance with weight 0.17, regarding the scaling factors of the worker.

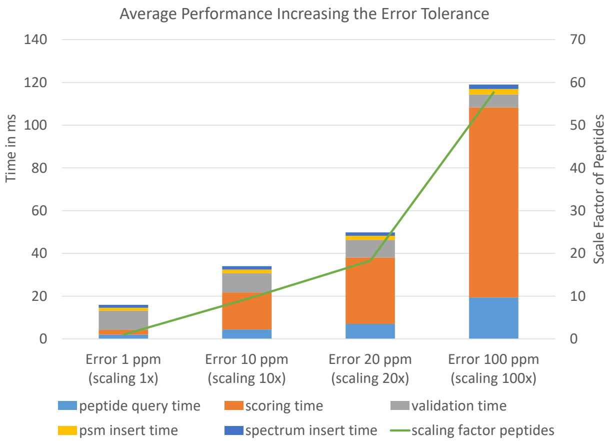

**Figure 8: Average performance in ms by increasing the error tolerance for the peptide query in MStream.**

In the end, we show that increasing the error tolerance or increasing the input rate affects the runtime of the worker and the overall system. Additionally, we see that the scaling factors are different and the amount of mass spectrometry data affects the process more than the amount of peptides. Evaluation of different subscription times is not necessary, since it changes the amount of spectra in a batch, which leads to the same results as the evaluation of the input rate.

*4.2.3 Experiment 3: Comparison to X!Tandem.* After evaluating the MStream platform, we now compare the runtime to the state-of-the-art tool X!Tandem, which runs on a local machine and uses one thread, to the runtime of one worker of our system. X!Tandem needs all mass spectrometry data as input at the beginning, while our worker can start processing a single spectrum. Nevertheless, we compare only the identification time.

In Figure 9, we show the average performance of a mass spectrometer, based on the experience with a Bruker device (On average 27 spectra per second), compared to the physical maximum of the device (100 spectra per second). Those rates reflect the speed at which one device and possible future devices deliver spectrum data.

X!Tandem using one thread processes 18 spectra per second on average and our single worker processes 45 spectra per second. The search parameters were similar for both tools. Thus, our single worker outperformed X!Tandem in single-threaded mode.

The comparison is not entirely fair since X!Tandem runs on a single machine using one thread, and our system runs in a cloud environment with at least 37 virtual CPUs. Based on the scaling factors from this experiment, we can calculate when it is financially reasonable to switch to our system. Our experiments show that on 58 CPUs and 990 spectra per second, our system is more efficient regarding CPU utilization than X!Tandem. In other words, our system becomes more CPU-efficient than X!Tandem if the data comes from 30 or more mass spectrometers.

The results show that MStream is capable of performing near-real-time analyses by efficiently querying the peptide data and scales out if needed. We targeted the three parameters - error tolerance, input rate and subscription time, to optimize the performance. Finally, we examined hardware resource consumption, which is acceptable for a central cloud-based analytic platform that serves thousands of mass spectrometers for clinical diagnostics.

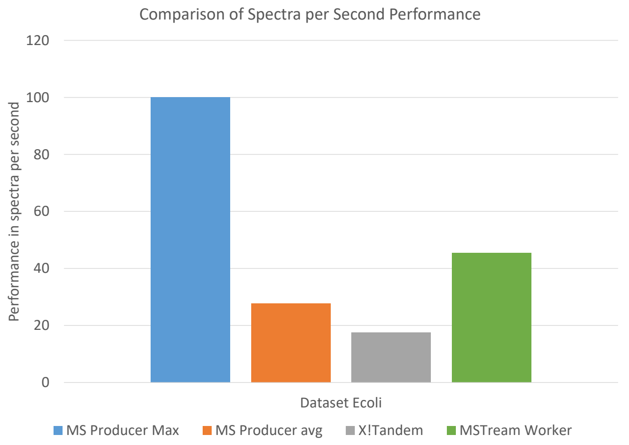

**Figure 9: Comparison of the performance of X!Tandem, MStream and a Mass Spectrometer.**

# 5 DISCUSSION

In this section, we discuss the results of the evaluation, particularly focusing on the goal to implement MStream as a central system for real-time diagnostics.

In Section 4.2.1, we show that our data preparation greatly increases the requirements on hard disk memory. If HDD space is limited, users have to tailor the prepared data to the diseases that they actually intend to diagnose. However, given the low prices for hard disks, we nevertheless argue that the speed-ups enabled by our data preparation step justify the increase in memory requirements.

Next, we examined the query performance and the query size regarding different tolerances (see Figure 5). Our measurements show that MStream scales linearly. The error tolerance property has the biggest influence on the overall performance. Since the error tolerance depends on the precision of the mass spectrometer, this should be considered during experiments using the minimum tolerance.

As described in Section 4.2.2, MStream provides results in near-real-time. In this experiment, we analyze the weighted impact of the number of peptides and spectra. First, we increased the input rate to analyze the out scaling and how it influences a single worker. We examined in Figure 6 that the scaling factor of the workers is linear and each worker delivers the same performance during processing. In Figure 7, we examined the influence of the peptide size on the worker performance. In Figure 8, we show the influence on the individual components in the scorer. Finally, we showed the number of spectra has a performance influence of 1, while the peptide query influences the performance with 0.17. We propose to adjust the pre-filtering methods in the stream producer (Section 3.3) for individual diagnostic scenarios to reduce the calculation effort and costs on the cloud.

In our last experiment in Section 4.2.3, we compared our system to the state-of-the-art software X!Tandem. Specifically, we compared the performance of a single MStream worker to X!Tandem running in single-thread mode. In this case, our system could process on average 45 spectra per second, while X!Tandem processed only 17. Furthermore, MStream outperforms X!Tandem with the input rate of 990 spectra per second (which equals 36 mass spectrometry measurements in parallel) with respect to CPU consumption. Hence, MStream clearly outperforms the state-of-the-art tool X!Tandem as a central platform for real-time diagnostics.

For the clinical use case, a central platform reduces the personnel effort for each clinic compared to local installations, which need qualified support and hardware on site. In MStream, every spectrum is processed one by one, making it possible to calculate the cloud costs of each scoring and map them back to each patient. Patient privacy can be ensured by adding a local de-identification step.

## 6 CONCLUSION

Research in biomarker detection has shown mass spectrometers to be great diagnosticians with growing capabilities. With a single mass spectrometer measurement, it is possible to diagnose thousands of diseases. However, only few hospitals own mass spectrometers because the state-of-the art, sequential data analysis workflow is slow, and maintaining the necessary digital infrastructure requires highly qualified personnel. This motivated us to evolve the data analysis of mass spectrometers to the cloud and break through the standard sequential pipeline. As part of this evolution, we collaborated with Bruker Daltonik GmbH to develop an adapter that allows us process the measurement data as it arrives from the mass spectrometer. We further developed a stream-based validation method based on logistic regression. Additionally, we prepared the protein data and used a column-based index structure to perform fast range queries on the protein data. Finally, we connected the components in the MStream scorer and evaluated the system. The result of these efforts is MStream, a proof of concept for a cloud-based mass spectrometry analytic platform for clinical diagnostics that works in near-real-time. The implementation is open source and available via GIT.

## 7 ACKNOWLEDGMENTS

The authors sincerely thank Niya Zoun, Ivayla Trifonova, Xiao Chen, Gabriel Campero Durand, Marcus Pinnecke, Sebastian Krieter and Andreas Meister for their support and advice. This work is partly funded by the de.NBI Network (031L0103), the European Regional Development Fund (grant no.: 11.000sz00.00.0 17 114347 0), the DFG (grant no.: SA 465/50-1), by the German Federal Ministry of Food and Agriculture (grants no.: 22404015), by Bruker Daltonik GmbH and dedicated to the memory of Mikhail Zoun.

## REFERENCES

[1] John S Cottrell and U London. 1999. Probability-based protein identification by searching sequence databases using mass spectrometry data. *electrophoresis* 20, 18 (1999), 3551–3567.

[2] Joshua Elias and Steven Gygi. 2010. Target-Decoy Search Strategy for Mass Spectrometry-Based Proteomics. *Methods in Molecular Biology* 604 (2010), 55–71.

[3] Bernardo A. P. et al. 2017. Metaproteomics as a Complementary Approach to Gut Microbiota in Health and Disease. *Front Chem* (2017).

[4] David Broneske et al. 2017. Accelerating multi-column selection predicates in main-memory - the Elf approach. In *IEEE International Conference on Data Engineering (ICDE)*. 647 – 658.

[5] Jürgen Cox et al. 2011. Andromeda: A Peptide Search Engine Integrated into the MaxQuant Environment. *Journal of Proteome Research* 10 (2011), 1535–3893.

[6] Pierre-Alain Maron et al. 2007. Metaproteomics: A New Approach for Studying Functional Microbial Ecology. *Microbial Ecology* Volume 53 (2007), 486–âĂŞ493.

[7] Robert Heyer et al. 2015. Metaproteomics of complex microbial communities in biogas plants. *Microbial Technology* 8 (04 2015).

[8] Robert Heyer et al. 2017. Challenges and perspectives of metaproteomic data analysis. *Journal of Biotechnology* 261, Supplement C (2017), 24 – 36. Bioinformatics Solutions for Big Data Analysis in Life Sciences presented by the German Network for Bioinformatics Infrastructure.

[9] Roman Zoun et al. 2018. Streaming FDR Calculation for Protein Identification. In *Advances in Databases and Information Systems*.

[10] Roman Zoun et al. 2019. Efficient Transformation of Protein Sequence Databases to Columnar Index Schema. In *International Workshop on Biological Knowledge Discovery and Data Mining (BIOKDD-DEXA)*. IEEE.

[11] Roman Zoun et al. 2019. MSDataStream - Connecting a Bruker Mass Spectrometer to the Internet. In *Datenbanksysteme für Business, Technologie und Web*.

[12] https://www.murphyandson.co.uk/wp-content/uploads/2015/04/slider3 1-700x400.jpg. feb2019.

[13] http:themocracy.comwp-contentuploads201607Microbial Community.jpg. feb2019.

[14] http://www.komabiotech.co.kr/www/product /pImages/sdspage1.jpg. feb2019.

[15] Avinash Lakshman and Prashant Malik. 2010. Cassandra: A Decentralized Structured Storage System. *SIGOPS Oper. Syst. Rev.* 44, 2 (April 2010), 35–40.

[16] Yahui Liu and et al. 2014. Biomarkers in Alzheimer's disease analysis by mass spectrometry-based proteomics. *International journal of molecular sciences* 15, 5 (may 2014), 7865–82.

[17] Markus Lubeck and et al. 2017. *PASEF on a timsTOF Pro defines new performance standards for shotgun proteomics with dramatic improvements in MS/MS data acquisition rates and sensitivity.* Technical Report. Bruker Daltonik GmbH.

[18] Renato Millioni and et al. 2013. Pros and cons of peptide isolectric focusing in shotgun proteomics. *Journal of chromatography. A* 1293 (June 2013), 1âĂŤ9.

[19] NCBI. 2002. FASTA format documentation. https://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&\DOC_TYPE=BlastHelp

[20] Orthodoxia Nicolaou and et al. 2016. Biomarkers of systemic lupus erythematosus identified using mass spectrometry-based proteomics: a systematic review. *Journal of cellular and molecular medicine* 21, 5 (2016), 993–1012.

[21] Darue A Prieto and et al. 2014. Mass spectrometry in cancer biomarker research: a case for immunodepletion of abundant blood-derived proteins from clinical tissue specimens. *Biomarkers in medicine* 8, 2 (2014), 269–86.

[22] Craig Robertson and C Beavis Ronald. 2003. A method for reducing the time required to match protein sequences with tandem mass spectra. *Rapid Communications in Mass Spectrometry 17* 17 (10 2003), 2310–2316.

[23] Dean Wampler. 2016. *Fast Data Architectures for Streaming Applications* (first ed.). OReilly Media.