Real-Time Processing of Continuous Physiological Signals in a Neurocritical Care Unit on a Stream Data Analytics Platform

Yong Bai, Daby Sow, Paul Vespa, and Xiao Hu

Abstract Continuous high-volume and high-frequency brain signals such as intracranial pressure (ICP) and electroencephalographic (EEG) waveforms are commonly collected by bedside monitors in neurocritical care. While such signals often carry early signs of neurological deterioration, detecting these signs in real time with conventional data processing methods mainly designed for retrospective analysis has been extremely challenging. Such methods are not designed to handle the large volumes of waveform data produced by bedside monitors. In this pilot study, we address this challenge by building a prototype system using the IBM InfoSphere Streams platform, a scalable stream computing platform, to detect unstable ICP dynamics in real time. The system continuously receives electrocardiographic and ICP signals and analyzes ICP pulse morphology looking for deviations from a steady state. We also designed a Web interface to display in real time the result of this analysis in a Web browser. With this interface, physicians are able to ubiquitously check on the status of their patients and gain direct insight into and interpretation of the patient's state in real time. The prototype system has been successfully tested prospectively on live hospitalized patients.

Keywords Stream computing • InfoSphere streams • Intracranial pressure (ICP) steady state • Real-time data processing and analysis

Introduction

Brain monitoring is becoming increasingly multimodal and data intensive. The data produced by monitors in neurocritical care meet Gartner's [1] well-known three Vs criteria of "big data." In terms of velocity, continuous brain signals including intracranial pressure (ICP) and electroencephalography (EEG) have a high temporal resolution, up to several thousand samples per second. They reflect fast physiological processes, the interpretation of which is critical for the detection of early signs of neurological deterioration. In terms of volume, the continuous nature of brain monitoring combined with the requirement of multiple-day monitoring for patients in a critical condition results in a large amount of patient data being produced. In terms of variety, brain monitoring tracks the states of hemodynamics, hydraulics, electrophysiology, and metabolism. Unfortunately, the processing capabilities of electronic medical record systems and bedside monitors do not scale to meet the requirements of big data analytical systems.

Traditionally (since the early 1990s [2]), researchers have developed homegrown systems to acquire and analyze data streams for brain monitoring. One excellent product is ICM+ [3]. ICM+ supports off-the-shelf data acquisition hardware to obtain analog waveform data from various brain monitoring devices. These waveform signals can then be analyzed using built-in algorithms, one of which is the pressure reactivity index (PRx). More recently, ICM+

Y. Bai,
Biomedical Engineering Graduate Program, Henry Samueli School of Engineering and Applied Science,
University of California-Los Angeles, Los Angeles, CA, USA

D. Sow
IBM T. J. Watson Research Center, Yorktown, NY, USA

P. Vespa
Department of Neurosurgery, David Geffen School of Medicine, University of California-Los Angeles, Los Angeles, CA, USA

X. Hu (✉)
Departments of Physiological Nursing/Neurosurgery, University of California-San Francisco, San Francisco, CA, USA

Institute for Computational Health Sciences, University of California-San Francisco, San Francisco, CA, USA

Affiliate, UCB/UCSF Graduate Group in Bioengineering, University of California-San Francisco, San Francisco, CA, USA
e-mail: xiao.hu@ucsf.edu

B.-T. Ang (ed.), *Intracranial Pressure and Brain Monitoring XV*, Acta Neurochirurgica Supplement, Vol. 122,
DOI 10.1007/978-3-319-22533-3_15, © Springer International Publishing Switzerland 2016

started to provide plug-in capabilities allowing additional external algorithm modules to be incorporated. This traditional model works well for single institutions, especially as a research tool. However, this model cannot easily meet the requirements of real-time clinical decision support that is well integrated with an enterprise's electronic medical record (EMR) system. It lacks the ability to obtain data from EMRs to enrich brain monitoring and the ability to provide results back into EMRs to enrich patient medical records.

To improve the analysis of brain monitoring data, we designed a pilot study by building a prototype system to analyze physiological data streams in real time. The analysis consists of advanced brain monitoring algorithms that we deployed on the IBM InfoSphere Streams computing platform (or Streams). Streams is a scalable middleware designed for high-performance real-time streaming analysis [4, 5]. Our prototype integrated an algorithm that we had developed previously to detect unstable ICP dynamics by analyzing ICP pulse morphology. With this system, we are able to continuously monitor and analyze the ICP steady state on the fly without persisting raw ICP data. The main objective of this study is to test the engineering feasibilities of obtaining real-time data, executing the algorithm, and reporting the results through a Web interface.

Materials and Methods

To process continuous physiological signals from standard bedside monitors in real time, a system should be:

1. Scalable, to handle large volumes of the high-rate streaming data from different sources
2. Extensible, to allow the addition of new analysis modules into the current system to process the streaming data
3. Interactive, to enable end users such as clinicians and nurses to meet their diverse needs and queries

These requirements were addressed with a prototype system that we designed, as illustrated in (Fig. 1). The system primarily consists of three modules:

1. A data acquisition module, which aggregates multiple physiological signals from different bedside monitors
2. A data processing module which processes streams in real time with advanced data analysis algorithms
3. An output module, which delivers the results produced to end users

We present details on these modules in the subsections.

Data Acquisition Module

A key requirement for acquiring brain monitoring data in enterprise environments is to minimize the usage of hardware to reduce maintenance costs and to increase the compatibility of the system with the rest of the information management infrastructure. For this study, we used the BedMasterEx server software from Excel Medical Electronics as a data hub to interface with bedside patient monitors. This server component continuously receives various data elements broadcast from all bedside monitors on a dedicated mission-critical network. In our case, the mission-critical network is a GE Unity Network. The data elements received are then archived in a relational database and flat files. The relational database of BedMasterEx maintains a table constantly refreshed with the last 10 s of vital sign and waveform data sampled at 240 Hz. This table contains all the data points needed to drive the real-time analysis that we are performing in this study.

Data Processing Module

The core of our prototype system is its data processing module, which is built on top of IBM InfoSphere Streams

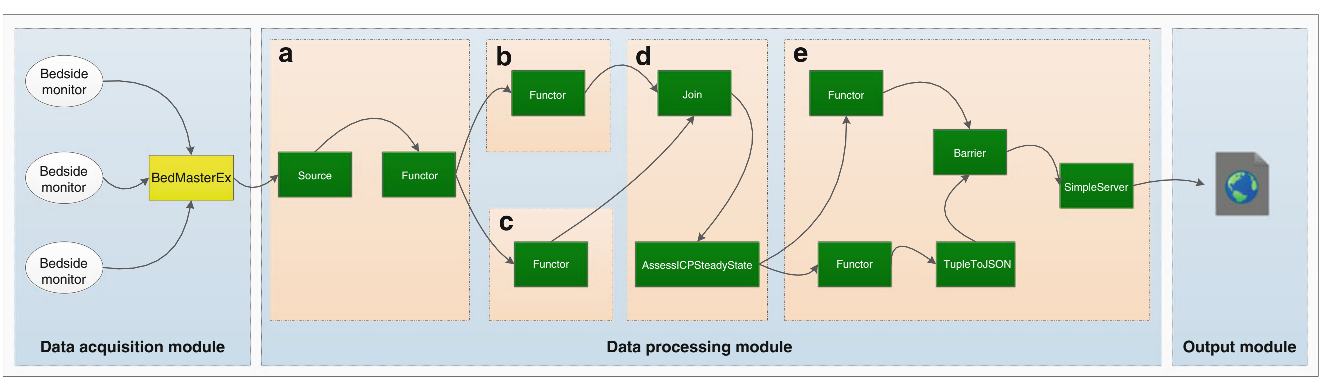

Fig. 1 Illustration of data streams flowing within the prototype system built upon InfoSphere Streams to detect unstable intracranial pressure (ICP) dynamic. (**a**) Adaption sub-module; (**b**) electrocardiography (ECG) sub-module; (**c**) ICP sub-module; (**d**) fusion and ICP assessing sub-module; (**e**) delivery sub-module

[4, 5]. InfoSphere Streams is an advanced computing platform that handles high-volume streams of structured and unstructured data from multiple sources. One of the key features of InfoSphere Streams is that it offers a new paradigm of stream computing that allows the continuous processing and analysis of data streams in real time with high performance. InfoSphere Streams is purely software. It runs on X86 or Power architecture running specific versions of Linux. It has three main components: the stream processing language (SPL) and its integrated development environment (IDE), used to author applications, a runtime system, and toolkits of stream processing operators. SPL is a high-level declarative flow composition language that facilitates the authoring of streaming application. With SPL, developers express streaming analytics as directed graphs of operators. Edges in these graphs represent data streams, while nodes are operators encapsulating arbitrary business logic. The SPL language is extensible. It allows programmers to define custom operators written in lower-level programming languages such as C++ and Java. These custom operators can be incorporated seamlessly with built-in operators to compose SPL applications. The InfoSphere Streams runtime system provides components and services to orchestrate the execution of SPL graphs on the underlying hardware. For instance, specific services continuously monitor resource utilization to optimize the placement of SPL operators on the underlying hardware.

With the aim of running a nontrivial algorithm to analyze data streams from ongoing hospitalized patient in this pilot study, we selected one of the algorithms we developed previously, called *AssessICPSteadyState*, to detect unstable ICP dynamics by analyzing ICP pulse morphology using ECG and ICP physiological signals [6]. To this end, the data processing module of our prototype system contains five components: adaption sub-module, ECG sub-module, ICP sub-module, fusion and ICP assessing sub-module, and delivery sub-module (Fig. 1). We describe each of these submodules as follows.

The *Source* operator in the adaption sub-module serves interface with the BedMasterEX SQL database to ingest physiological data streams. It polls the BedMasterEX underlying database at regular time intervals to get the latest physiological readings for a predefined set of patients admitted in this study. The ECG and ICP sub-modules are responsible for formatting ECG and ICP streams respectively. These two sub-modules operate in parallel. In the fusion and ICP assessing sub-module, 10-s sliding windows of ECG and ICP data streams are merged by the *Join* operator, which synchronizes and joins together ECG and ICP data streams. The stream produced flows into our customized *AssessICPSteadyState* operator to detect the stability of ICP dynamics. The output stream of *AssessICPSteadyState* operator is converted into a JavaScript Object Notation (JSON) stream with additional routing meta data and flows into a WebSocket sink operator that we developed in the delivery sub-module. This WebSocket sink is built on top of a Jetty web server and provides a publish–subscribe mechanism that is used by the web application to subscribe to the results of this real-time analysis. This entire sequence of data analysis and results delivery is deployed on InfoSphere Streams and is executed on demand as a patient is admitted into this study.

Continuous Detection of Steady-State Intracranial Pressure Dynamics With the AssessICPSteadyState Algorithm

For this proof of concept, we selected a modified ICP analysis algorithm that had been developed by our group [6], called *AssessICPSteadyState*. This algorithm was developed to continuously detect the deviation of ICP dynamics from the steady state. These deviations are important to detect as they may be caused by acute hidden intracranial changes (e.g., growth of a hematoma or acute dilation of the ventricles), which can lead to compliance changes of the intracranial compartments. The original algorithm was validated based on induced acute hydrocephalus among patients with slit ventricle syndrome due to chronic cerebrospinal fluid shunt insertion. The basic principle of the algorithm is that ICP pulse morphologies of different ICP pulses from a given patient in a steady state of ICP dynamics resemble each other when their mean ICP levels are similar. For example, a B-wave may have been identified to be in steady state because the variations of mean ICP in a B-wave cycle merely mean that the system moves along a single pressure–volume curve. Therefore, different ICP pulses, with comparable mean ICP, from different B-wave cycles are still similar in shape.

To detect deviations from the steady state, our algorithm calculates the distance between pulses with mean ICP differences within ±1 mmHg and then quantifies the distribution of this distance from all qualified pulses. In the paper [6], we described three different metrics for measuring the difference between a pair of pulses and showed that the geodesic distance works best for this problem. The other two metrics, based on the Euclidean distance and Pearson's correlation coefficient respectively, also showed typical signs of unstable ICP dynamics for patients with acute ventricular changes between two consecutive brain imaging studies, which include either a broadening of the histogram or a multimodal histogram. To update the histogram per each new pulse, a 4-h search window is used within which pulses with a similar mean ICP to that of the new pulses are analyzed. In this way, we can provide beat-by-beat characterization of the histogram, focusing on the illustration of any emerging multimodal histogram or broadening of the histogram of inter-pulse distances [7, 8].

The *AssessICPSteadyState* algorithm was initially implemented in MATLAB. To integrate this MATLAB implementation in SPL for real-time processing, we leveraged the MATLAB C++ compiler [9] to produce a shared library. We then implemented a custom SPL operator (i.e., *AssessICPSteadyState* operator in Fig. 1), to invoke a mini-server that executes the *AssessICPSteadyState* algorithm from this shared library.

Output Module

We developed a web application to visualize the results of the real-time implementation of the *AssessICPSteadyState* algorithm running on InfoSphere Streams, as explained in the previous section. This application is capable of parsing the JSON messages published by the WebSocket sink described above. To receive these JSON messages from this sink, the web application establishes a web socket connection with a web server hosted within the web socket sink on InfoSphere Streams. The web application then sends a subscription message to the server requesting the output from the *AssessICPSteadyState* computation. Upon reception of the *AssessICPSteadyState* results, the web socket sink then transmits these results to the web application in real time. We designed this web application to allow end users to subscribe to such results on a per patient basis.

Results

We used IBM InfoSphere Streams v2.0 to deploy this prototype system on Cent OS v6.0, a 64-bit system. Both Streams and the web application have been successfully tested with ongoing hospitalized patients in a neurocritical care unit at UCLA Ronald Regan Medical Center. The Institutional Review Board waiver for consent was obtained for this study.

Figure 2 shows screenshots of the web application after running the system for a specific real patient. In this case, we were able to observe in real time that the patient was initially in a stable ICP state (panel a in Fig. 2), but then moved into an unstable state (panel b in Fig. 2). The hallmark of an unstable ICP dynamic system is that the ICP pulses at a similar mean ICP level are different in their morphologies.

Discussion

We demonstrated that IBM InfoSphere Streams is capable of hosting a complex ICP signal analysis algorithm to process continuous signals in real time. Based on various resource utilization measures, we extrapolate from this result that this InfoSphere Streams is capable of hosting multiple algorithms of this complexity to process data from multiple patients.

InfoSphere Streams has its own programming language and toolboxes (e.g., time series analysis toolbox) that can be utilized to implement various algorithms. We believe that the implementation of algorithms in this native way further increases the computational efficiency of the overall solution, as internal resource optimization may have more flexibility to effectively schedule and allocate computational resources to accomplish the specified task. Furthermore, InfoSphere Streams also provides mechanisms to incorporate algorithms developed using other languages including C/C++, R, and MATLAB. This flexibility is very desirable, not only for proof of concepts, but also for the use of cases requiring the integration of large amounts of legacy software.

For future work, the system developed in this pilot project needs to be further tested in a real-world situation to assess its scalability and to test the effectiveness of the algorithm in detecting the emergence of unstable ICP dynamics.

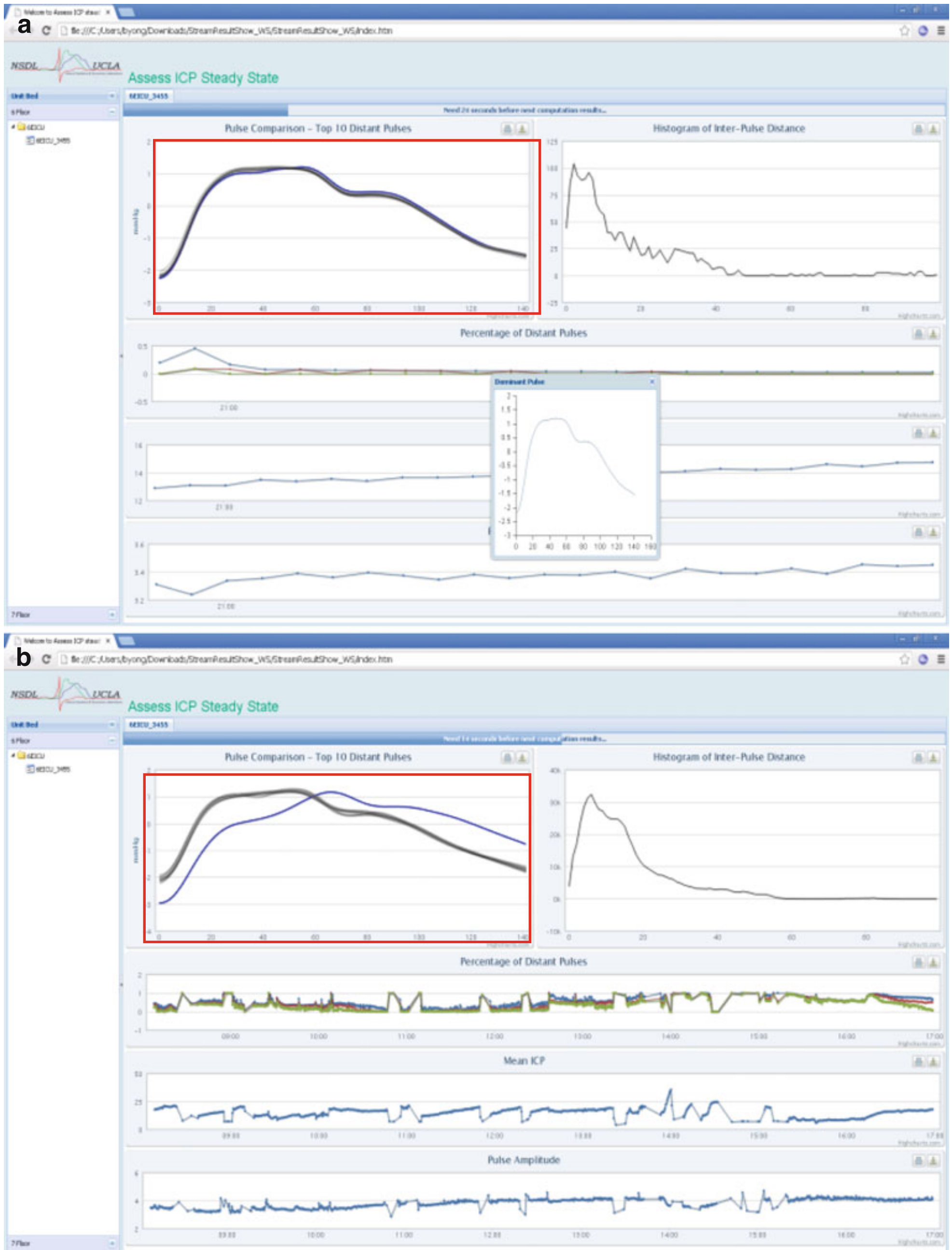

Fig. 2 Screenshots of the web application showing that a patient was initially in a stable ICP state, but later became unstable. (**a**) Patient ICP dynamic state is stable as pulses at similar mean ICP levels resemble each other, as highlighted by the pulses displayed in the *red rectangle*. (**b**) Patient ICP dynamic state is unstable, as pulses at similar mean ICP levels do not resemble each other, as highlighted by the pulses displayed in the *red rectangle*

Conflicts of Interest There are no conflicts of interest to declare. This work is partially supported by NS076738 and UCSF Institute for Computational Health Sciences.

References

1. Laney D(2012) The importance of 'big data': a definition. gartner report
2. Czosnyka M, Batorski L, Laniewski P, Maksymowicz W, Koszewski W, Zaworski W (1990) A computer system for the identification of the cerebrospinal compensatory model. Acta Neurochir (Wien) 105(3–4):112–116
3. Smielewski P, Czosnyka Z, Kasprowicz M, Pickard JD, Czosnyka M (2012) ICM+: a versatile software for assessment of CSF dynamics. Acta Neurochir Suppl 114:75–79
4. Hu X, Gonzalez N, Bergsneider M (2012) Steady-state indicators of the intracranial pressure dynamic system using geodesic distance of the ICP pulse waveform. Physiol Meas 33(12):2017–2031
5. Roger R. IBM Infosphere streams: redefining real time analytic processing. http://public.dhe.ibm.com/software/data/sw-library/ii/whitepaper/InfoSphereStreamsWhitePaper.pdf. Accessed 28 Jan 2014
6. http://pic.dhe.ibm.com/infocenter/streams/v2r0/index.jsp. Accessed 28 Jan 2014
7. Alain B, Eric B, Hanhua F, Anand R, Anton R, Olivier V, Haris K, Carlos M (2010) In: IBM infosphere streams for scalable, real-time, intelligent transportation services. Proceedings of the 2010 ACM SIGMOD International Conference on Management of data, 6–11 June, Indianapolis
8. Blount M, Ebling MR, Eklund JM, James AG, McGregor C, Percival N, Smith KP, Sow D (2010) Real-time analysis for intensive care: development and deployment of the artemis analytic system. IEEE Eng Med Biol Mag 29(2):110–118
9. http://www.mathworks.com/products/compiler/. Accessed 28 Jan 2014