Decision Support Systems xxx (2017) xxx–xxx

Contents lists available at ScienceDirect

Decision Support Systems

journal homepage: www.elsevier.com/locate/dss

Retrieving batch organisation of work insights from event logs

Niels Martin*, Marijke Swennen, Benoît Depaire, Mieke Jans, An Caris, Koen Vanhoof

Hasselt University, Agoralaan Building D, Diepenbeek 3590, Belgium

ARTICLE INFO

Available online xxxx

Keywords:
Batch processing
Event log
Event log insights
Process mining
Business process management
Process metrics

ABSTRACT

Resources can organise their work in batches, i.e. perform activities on multiple cases simultaneously, concurrently or intentionally defer activity execution to handle multiple cases (quasi-) sequentially. As batching behaviour influences process performance, efforts to gain insight on this matter are valuable. In this respect, this paper uses event logs, data files containing process execution information, as an information source. More specifically, this work (i) identifies and formalises three batch processing types, (ii) presents a resource-activity centered approach to identify batching behaviour in an event log and (iii) introduces batch processing metrics to acquire knowledge on batch characteristics and its influence on process execution. These contributions are integrated in the Batch Organisation of Work Identification algorithm (BOWI), which is evaluated on both artificial and real-life data.

1. Introduction

Business processes are composed of a series of connected activities executed by resources. Resources, such as process participants or equipment [1], are assigned to activities and typically carry these out on multiple cases such as files or products. Assuming that arriving cases are handled immediately when the resource becomes available is an undue simplification of reality. Employees might deem it more efficient to accumulate files and treat the entire stack later or machines can process multiple products at the same time. This type of resource behaviour is referred to as batch processing.

While the occurrence of batch processing might be readily observable for *passive resources* such as machines, it is typically less straightforward to determine how human resources, or *active resources* [1], organise their work. The latter is especially the case for processes in which staff members have a lot of freedom to arrange their tasks as they desire. Direct observation of staff members' behaviour has limitations as it is both time-consuming and the Hawthorne effect can cause observed behaviour to deviate from real behaviour when humans know they are being observed [2]. Consequently, investigating the use of more readily available information sources is valuable. In this respect, event logs containing process execution information can be analysed, which belongs to the process mining domain. While batch processing is studied widely in the operations management, operations research literature and, to a lesser extent, the process modelling domain, limited attention is attributed to this topic in process mining.

This paper is the first paper to systematically analyse batching behaviour using an event log. More specifically, the key contributions of this paper are threefold. Firstly, three types of batch processing are distinguished and formalised. Secondly, a resource-activity centered approach is presented to identify these batch processing types from an event log. Finally, batch processing metrics are defined to describe the identified batches and the implications of batching on process execution. These contributions are included in the Batch Organisation of Work Identification algorithm (BOWI). Even though the contributions of this paper are of general interest, they are especially useful for business processes in which human resources have significant freedom in their work organisation. As extensive observations would, for instance, be required in such contexts, using the proposed technique allows companies to gain insight in batching behaviour from event data.

The current paper is situated at the intersection between Business Process Management and process mining, which corresponds to one of the focal points of this special issue. More specifically, the generated insights in batching behaviour will support process modelling activities and decision-making within the BPM life cycle [1]. Process modelling is facilitated as event log analysis can provide suitable values for parameters such as the batch size. Integrating batching behaviour will lead to more realistic process models which can, for instance, be used for simulation purposes. Simulation models serve as a decision support tool as it enables organisations to evaluate policy alternatives prior to implementation [3]. Besides shouldering process modelling, BOWI also provides direct support for decision-making. The algorithm allows companies to judge the desirability

* Corresponding author.
E-mail address: niels.martin@uhasselt.be (N. Martin).

http://dx.doi.org/10.1016/j.dss.2017.02.012
0167-9236/

of batching behaviour by showing its influence on process performance. Consequently, a company can determine whether batching behaviour should be encouraged or discouraged. Besides positioning it in the BPM life cycle, this work can also be framed within the BPM use case *extend model* as it leverages resource information in the event log, which is useful to extend a process model [4].

The paper is structured as follows. Section 2 presents a running example and defines the three types of batch processing. The BOWI-algorithm is outlined in Section 3, after which it is evaluated on both artificial and real-life data in Section 4. Finally, related work, the algorithm's limitations and conclusions are presented in Sections 5, 6 and 7, respectively.

2. Preliminaries

This section presents some preliminary topics that will be used in the remainder of this paper. A running example is introduced in Section 2.1 and three batch processing types are distinguished in Section 2.2.

2.1. Running example

Throughout this paper, a simplified process at the emergency department of a hospital will serve as a running example. The process model, annotated with all assumed parameters, is visualised in Fig. 1. After a patient registers at the reception (R), an initial triage and assessment by a doctor follows (T). Next, a patient either (i) undergoes an X-ray examination (X) or (ii) has laboratory tests performed on his blood samples (L) and is subjected to an MRI scan (M). When the required tests are completed, the patient discusses the further treatment with a medical specialist (S) after which the patient checks out (C). All time units are expressed in minutes and patient interarrival times and activity durations are assumed to follow an exponential and triangular distribution, respectively.

2.2. Batch processing type definition

Batch processing is defined as a type of work organisation in which a resource executes a particular activity on multiple cases simultaneously or concurrently, or intentionally defers activity execution to handle multiple cases (quasi-) sequentially. As in Martin et al. [5], three batch types are distinguished: simultaneous, concurrent and sequential batch processing. To exemplify the difference between these types, Fig. 2 shows the activities executed for six patients, where an activity is always executed by the same resource across all cases.

- **Sequential batch processing.** Activity instances are in a sequential batch when a resource intentionally defers the execution of this activity such that multiple cases can be handled (almost) immediately after each other. Consequently, all cases included in a batch need to be present at the activity before the resource starts processing the batch's first case. The latter distinguishes sequential batch processing from mere queue handling, stressing its intentional nature. In Fig. 2, the initial assessment by a doctor takes place in sequential batches. Given the doctor's busy schedule, he occupies himself with other tasks until several patients need to undergo an initial assessment, after which they are handled sequentially.
- **Simultaneous batch processing.** Activity instances are in a simultaneous batch when they are executed by the same resource for distinct cases at exactly the same time. For example: blood samples from several patients can be analysed in the same run, as shown in Fig. 2.
- **Concurrent batch processing.** Activity instances are in a concurrent batch when they are executed by the same resource for distinct cases partially overlapping in time. This indicates that the resource can handle multiple cases at the same time, but is flexible as it is not required that processing starts and ends at the same time for all cases. In Fig. 2, registrations and check-outs illustrate different types of concurrent batch processing, because, e.g., the receptionist already starts registering the next patient while the current patient is filling out a drug allergy form which is required to finish his registration.

The above batch processing types are largely consistent with Wu [6], where simultaneous and sequential batch processing correspond to the concepts of parallel and serial process batches, respectively. Concurrent batch processing is not included in Wu [6]. In operations management literature, batch processing is commonly referred to as the intermittent production of a particular type of product [7,8], where production volumes are situated between a job shop setting with small volumes and mass production [7].

3. Batch organisation of work identification algorithm

This section proposes the Batch Organisation of Work Identification Algorithm (BOWI), which generates insights in batching behaviour from an event log. A general overview is presented in Section 3.1. Afterwards, Sections 3.2–3.6 present the algorithm in more detail. In Section 3.7, the implementation of the algorithm is briefly discussed.

3.1. General overview

As shown in Fig. 3, BOWI's input is an event log. This event log, consisting of atomic events, is converted to an activity log by mapping start events to their corresponding complete events. This activity log is restructured in a resource-activity matrix (*RAM*), where each cell contains activity instances of a particular resource-activity combination. Using the *RAM* as an input, a batching matrix (*BM*) is

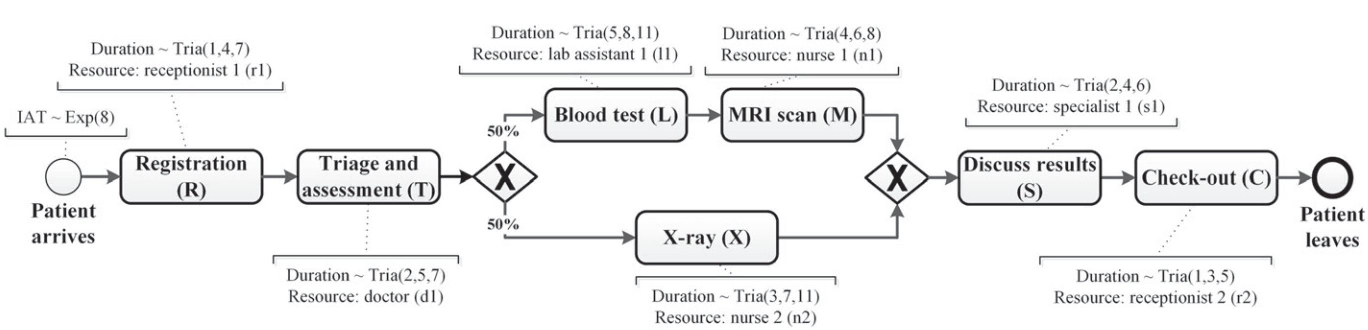

Fig. 1. Process model running example.