

Discovering the software process by means of stochastic workflow analysis

Alberto Colombo, Ernesto Damiani, Gabriele Gianini *

University of Milan, Department of Information Technology, via Bramante 65, 26013 Crema (CR), Italy

Available online 10 August 2006

Abstract

A fundamental feature of the software process consists in its own stochastic nature. A convenient approach for extracting the stochastic dynamics of a process from log data is that of modelling the process as a Markov model: in this way the discovery of the short/medium range dynamics of the process is cast in terms of the learning of Markov models of different orders, i.e. in terms of learning the corresponding transition matrices. In this paper we show that the use of a full Bayesian approach in the learning process helps providing robustness against statistical noise and over-fitting, as the size of a transition matrix grows exponentially with the order of the model. We give a specific model–model similarity definition and the corresponding calculation procedure to be used in model-to-sequence or sequence-to-sequence conformance assessment, this similarity definition could also be applied to other inferential tasks, such as unsupervised process learning.

© 2006 Elsevier B.V. All rights reserved.

Keywords: Software process; Workflow management; Stochastic dynamics; Markov chains; Machine learning; Bayesian methods; Time-sequence analysis; Similarity measures

1. Introduction

The software industry shows an increasing awareness of the need to invest in accountability and quality of information technology (IT) services, providing all methodological support to ensure that IT-related processes will achieve companies' business goals. One of the main issues to be addressed is the lack of a solid IT governance methodology, dealing with software process control on the basis of fine-grained process metrics.

In principle, business process specifications should include a well-understood *workflow*, designed before enactment and adjusted whenever change happens. Applying top-down workflow analysis to a business process definition would bring the details of that process into focus, specifying by whom, where and when each business process activity is carried out.

In the software development practice, however, analysis and tuning are mostly performed on the high level model of the process, whereas little is done about the underlying workflows: they are less known and therefore less defined, tuneable or manageable. Lack of workflow analysis makes software process governance difficult and sometimes ineffective. In this paper, we shall discuss a bottom-up

* Corresponding author.

E-mail address: gianini@dti.unimi.it (G. Gianini).

approach where workflow knowledge is inferred from process log data by means of *process mining*.

The term *process mining* refers to a collection of methods for distilling a structured process description from low-level process metrics and workflow traces. When purposely designed logs are available [1,2], process mining results in some form of a posteriori process model that can be compared with the a priori model, supporting process tuning and fine-grained analysis.

Generally speaking, process mining can be used to exploit non-purposely designed information sources (e.g. collaborative development and design environments, or transactional system) in order to extract indicators suitable for governance purposes (e.g. as defined by the *Control Objectives for Information and Related Technology process*, COBIT [3]), or for maturity scales like the one introduced by the Software Engineering Institute's *Capability Maturity Model* (CMM) [4].

Conceptually, the starting point of process mining is an *extended process log*, containing information about the process as it is actually being executed. The extended process log is assumed to have the following properties: (i) each event refers to a task (i.e., a well-defined step in the workflow), (ii) each event refers to a case (i.e., a workflow instance), and (iii) events are totally ordered. Usually, existing process logs need to be integrated (e.g., by encoding them in a XML data format) and carefully filtered before fulfilling those requirements.

Potentially valuable sources for setting up an extended process log include, for instance – but are not limited to – collaborative development and design environments, project management tools and WFM (workflow management) systems.

In order to make process log information useful, some synthetic knowledge will have to be drawn from raw data, either in the form of local patterns or in terms global features of the dynamics of the process.

A key issue is that of mapping this synthetic knowledge onto a specific software process representation. Much work has been done on highly expressive metadata for process engineering, including reusable, modular *ontologies* for process description [5]. In our overall approach [6], a process-independent metamodel is instantiated in specific process description models whose instances are then linked to synthetic knowledge, deterministically drawn from process data. However, experience has

shown that the process of learning a process model from empirical data must have a statistical character to cope with an apparent variability in the manifestation of an underlying process dynamics.

In this paper, we focus on dealing with the software process stochastic nature when extracting synthetic knowledge from process logs. A time-honoured approach for extracting the stochastic dynamics of a process from extended log data is that of modelling the process as a Markov model (further candidate models will be proposed during the discussion): the discovery of the short or medium range dynamics of the process can then be cast in terms of the learning of Markov models of different orders, i.e. in terms of learning the corresponding transition matrices. However, since the size of a transition matrix grows exponentially with the order of the model, the lack of statistics can become a problem: adopting a full Bayesian approach in the learning process can provide the required robustness against statistical noise and against the risk of over-fitting.

In the next sections, after a review of the state of the art (Section 2), we provide the basic definitions of Markov models and the basic formulas for their inference from data, for the instance and model comparison (for validation/conformance assessment) and give a specific similarity definition and the corresponding calculation procedure (Section 3), some experimental results obtained over the only time texture of a real world development process will show the effectiveness of this technique in process validation (Section 4), a discussion and the outline of possible developments will close the paper (Section 5).

2. State of the art

Several types of (stochastic or deterministic) models can be inferred from an extended process log: *generative* models (trace languages, grammars) as well as *recognition* or *parsing* models (automata, graphs). In general, heuristics and approximate analysis techniques are used, since the computational problem of finding a minimum finite-state acceptor compatible with given data is well known to be NP-hard [7].

For quite a long time, the focus of process mining research has been on finding heuristics (e.g., based on ordering relations of the events in the process log) capable of providing efficient algorithms for building process models in special cases.

متن کامل مقاله

دریافت فوری ←

ISIArticles

مرجع مقالات تخصصی ایران

- ✓ امکان دانلود نسخه تمام متن مقالات انگلیسی
- ✓ امکان دانلود نسخه ترجمه شده مقالات
- ✓ پذیرش سفارش ترجمه تخصصی
- ✓ امکان جستجو در آرشیو جامعی از صدها موضوع و هزاران مقاله
- ✓ امکان دانلود رایگان ۲ صفحه اول هر مقاله
- ✓ امکان پرداخت اینترنتی با کلیه کارت های عضو شتاب
- ✓ دانلود فوری مقاله پس از پرداخت آنلاین
- ✓ پشتیبانی کامل خرید با بهره مندی از سیستم هوشمند رهگیری سفارشات