Reasoning and Verification with Data Petri Nets

(Discussion/Short Paper)

Paolo Felli, Marco Montali and Sarah Winkler

Free University of Bozen-Bolzano, Italy

Abstract

Data-aware processes represent and integrate structural and behavioural constraints in a single model, and are thus increasingly investigated in business process management and artificial intelligence. In this spectrum, Data Petri nets (DPNs) have gained growing popularity thanks to their ability to balance simplicity with expressiveness. To faithfully model real-world processes, the data perspective often requires arithmetic, which renders basic problems undecidable.

Nonetheless, we show here that by appropriately restricting the constraint language or structure, a number of important tasks becomes decidable for practical classes of systems: This includes linear- and branching-time model checking, strategic reasoning, and verification of data-aware soundness.

Keywords

process mining, Data Petri nets, verification

1. Introduction

Integrating control-flow and data aspects to holistically capture the dynamic evolution of processes is a central problem in AI [1] and business process management (BPM) [2]. Multi-perspective models that reflect this interdependency have consequently emerged [2, 3], in turn calling for corresponding analysis techniques. *Data Petri nets* (DPNs) have recently been put forward as a concise yet expressive formalism to capture data-aware processes where the control-flow backbone of the process is specified using Petri nets, the data component consists of a set of variables, and the data-process interplay is reflected in read-write conditions over variables, attached to net transitions. DPNs are interesting for two reasons. On the one hand, they can be used to formalize sophisticated processes combining control-flow, case data, and decision models [4]. On the other hand, they can be discovered automatically from logs [5, 6, 7].

Combined modeling of data and processes is notoriously error-prone. However, automatic discovery techniques typically come without any correctness guarantee. This makes verification crucial. Verification of DPNs, in turn, is highly challenging, as the interplay of control-flow and data requires to deal with infinitely many states. This difficulty is further aggravated if action guards involve arithmetic operations, despite the fact that arithmetic is essential to faithfully model practical applications [8]: model checking of transition systems operating over data with arithmetic constraints is known to be undecidable, as it is easy to model a two-counter system.

PMAI'22: Process Management in the AI Era
pfelli@unibz.it (P. Felli); montali@inf.unibz.it (M. Montali); winkler@inf.unibz.it (S. Winkler)
0000-0001-9561-8775 (P. Felli); 0000-0002-8021-3430 (M. Montali); 0000-0001-8114-3107 (S. Winkler)

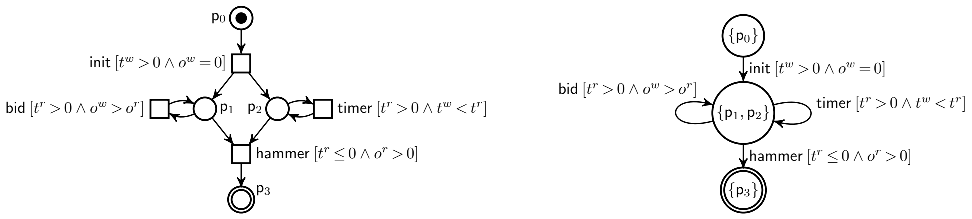
p0
init [$t^w > 0 \wedge o^w = 0$]
bid [$t^r > 0 \wedge o^w > o^r$]
p1
p2
timer [$t^r > 0 \wedge t^w < t^r$]
hammer [$t^r \leq 0 \wedge o^r > 0$]
p3
{p0}
init [$t^w > 0 \wedge o^w = 0$]
bid [$t^r > 0 \wedge o^w > o^r$]
{p1, p2}
timer [$t^r > 0 \wedge t^w < t^r$]
hammer [$t^r \leq 0 \wedge o^r > 0$]
{p3}

Figure 1: DPN (left) and DDS (right) for simple auction model.

However, we here outline how for a number of relevant DPN classes where the constraint language and/or the structure are suitable restricted, important analysis tasks become decidable. These include linear and branching time model checking, of which *data-aware soundness* checking is a special case, as well as strategic reasoning. We analyze DPNs for realistic processes from a variety of domains that were presented in the literature, and show that many of these examples actually fall into one of the decidable classes. This extended abstract summarizes recent publications [9, 4, 10, 11, 12, 6], as follows: We first give an informal summary of DPNs (2). Sec. 3 points to decidability results for linear- and branching-time model checking. Diving into an important special case of the latter, Sec. 4 is devoted to checking data-aware soundness, and Sec. 5 discusses strategic reasoning. We conclude with some directions for future work.

2. Data Petri Nets

A *Petri net with data* (DPN) $\mathcal{N}$ is a Petri net that maintains a fixed, finite set of process variables V of some numeric type, and where actions are associated with guard expressions called *constraints* that can at the same time read and write the variables V. For instance, Fig. 1 shows a DPN that models a simple auction process. Here $V = \{o, t\}$, where variable o holds the last offer, and t is a timer. For instance, the constraint $t^r > 0 \wedge o^w > o^r$ associated with action bid expresses that the current value of t must be positive; and the value of o is increased by this action (i.e., superscript r refers to the read, w to the written value). Here we assume that all constraints are linear arithmetic expressions, but sometimes restrict to the following forms: (i) *monotonicity constraints* (MCs) admit only variable-to-variable/constant comparisons wrt. $=, <, \leq, \neq$ for variables with domain $\mathbb{Q}$ or $\mathbb{R}$, such as $x < y$ or $3 \leq x$; (ii) *integer periodicity constraints* (IPCs) have the form $x = y$, $x \odot d$ for $\odot \in \{=, \neq, <, >\}$, $x \equiv_k y + d$, or $x \equiv_k d$, for variables x, y with domain $\mathbb{Z}$ and $k, d \in \mathbb{N}$; (iii) *gap-order constraints* (GCs) have the form $x - y \geq k$ for x, y integer variables or constants, and $k \in \mathbb{N}$.

A *state* in a DPN $\mathcal{N}$ is a pair (M, α) of a marking M and an assignment α that maps V to values in $\mathbb{Z}$ or $\mathbb{Q}$. We fix some initial assignment, and call a *run* a finite sequence of transition firings in the underlying Petri net that respect guards. For instance, a possible run for the DPN in Fig. 1 is $(\{\mathsf{p}_0\}, \left[\begin{smallmatrix} t=0 \\ o=0 \end{smallmatrix}\right]) \xrightarrow{\mathsf{init}} (\{\mathsf{p}_1, \mathsf{p}_2\}, \left[\begin{smallmatrix} t=1 \\ o=0 \end{smallmatrix}\right]) \xrightarrow{\mathsf{timer}} (\{\mathsf{p}_1, \mathsf{p}_2\}, \left[\begin{smallmatrix} t=0 \\ o=0 \end{smallmatrix}\right])$. For analysis tasks, it is convenient to work on a simpler model. To this end, in place of DPNs we consider *data-aware dynamic systems* (DDSs), which are labeled transition systems where transitions are associated with constraints as above. Every bounded DPN can be transformed into a DDS, by considering all possible markings. The right system in Fig. 1 shows the auction process as a DDS.

3. Model Checking

From now on, we assume a given DDS $\mathcal{B}$. As process executions are typically assumed to be finite, we adopt for both linear and branching time model checking a finite trace semantics.

Linear time. We use as specification language an enriched version of LTL, where constraints over the variables V and the control states of $\mathcal{B}$ occur as atoms. The verification problem is then to decide whether, given $\mathcal{B}$ and an LTL$_f$ formula ψ, there is a run of $\mathcal{B}$ that satisfies ψ. In [10], we define an abstract property of $\mathcal{B}$ and ψ called *finite summary*, which guarantees decidability of the verification task. This property holds, e.g., if all constraints are in one of the classes MC, IPC, or GC, but also if certain structural properties hold. Moreover, the finite summary property is *modular*, in the sense that if a DDS $\mathcal{B}$ can be suitably decomposed into subsystems, then $\mathcal{B}$ enjoys the property. E.g., all constraints in the DDS of Fig. 1 are MC; so model checking is decidable with respect to an MC property, like $\psi = \Box(\mathsf{p}_3 \rightarrow o > 0)$.

Branching time. We next consider a CTL*-like extension of the above specification language, and the verification task to decide whether the runs of $\mathcal{B}$ satisfy a CTL* formula χ. (More precisely, we aim to find conditions on the initial valuation of $\mathcal{B}$ such that χ is satisfied.) In [12], we extend the above approach from LTL$_f$ to CTL$^*_f$, and prove decidability for a similar abstract property. E.g., we can check that the DDS in Fig. 1 has deadlocks: $\mathsf{A}\,\mathsf{G}\ \mathsf{E}\,\mathsf{F}\,\mathsf{p}_3$ does not hold, expressing that there are states where no final state is reachable (namely $\{\mathsf{p}_1, \mathsf{p}_2\}$ with $t \leq 0$).

4. Soundness Checking

A well-established notion of correctness for dynamic systems is that of *soundness* [13]. Intuitively, this property requires that (i) all activities in the process occur in some execution; (ii) the process can reach a *final* state from every reachable state and (iii) final states are reached in a 'clean' way, without leaving any thread of the process still hanging. For instance, as mentioned in Sec. 3, Fig. 1 violates the second property. In [6] it is shown that DPNs over variable-to-constant comparisons constitute a decidable case that is expressive enough to capture data-aware process models equipped with S-FEEL DMN decisions [4]. In [9] this result is extended to full monotonicity constraints, and in [11] to other systems that satisfy (a more restricted version of) the property of finite summary.

5. Strategic Reasoning

The evolution of systems is often nondeterministic when it comes to resolving decision points with multiple enabled conditions, or to choosing which value to pick when updating a variable. While in reality these sources of nondeterminism are typically controlled by multiple, autonomous actors, verification has almost uniformly adopted the simplifying assumption that these actors cooperate. In [14] this premise is relaxed, and it is shown how well-established strategy synthesis approaches for temporal specifications can be combined with faithful data abstraction methods, towards a constructive, readily implementable technique for strategy synthesis in DDSs over MCs.

Conclusion. We described how several important analysis tasks become decidable for certain classes of DDSs, and hence DPNs. Linear and branching time verification, as well as soundness checking, are implemented in the tool ada (https://ctlstar.adatool.dev). The used procedures based on [10, 11, 12] are decision procedures for systems with finite summary. We evaluated ada on DPNs from the literature, and found that most examples fall into one of the decidable classes. In future work, we plan to study further decidable classes for strategic reasoning, monitoring of DPNs, and applicability of these approaches to richer, array-based systems [15].

Acknowledgements. This work is partially supported by the UNIBZ projects SMART-APP and VERBA, and the PRIN 2020 Italian project PINPOINT.

References

[1] C. Baral, G. De Giacomo, Knowledge representation and reasoning: What's hot, in: Proc. 29th AAAI, 2015, pp. 4316–4317.

[2] M. Reichert, Process and data: Two sides of the same coin?, in: OTM 2012, volume 7565 of *LNCS*, 2012, pp. 2–19.

[3] K. Batoulis, A. Meyer, E. Bazhenova, G. Decker, M. Weske, Extracting decision logic from process models, in: 27th CAiSE, volume 9097 of *LNCS*, Springer, 2015, pp. 349–366.

[4] M. de Leoni, P. Felli, M. Montali, Integrating BPMN and DMN: modeling and analysis, J. Data Semant. 10 (2021) 165–188.

[5] F. Mannhardt, Multi-perspective Process Mining, Ph.D. thesis, Technical University of Eindhoven, 2018.

[6] M. de Leoni, P. Felli, M. Montali, A holistic approach for soundness verification of decision-aware process models, in: 37th ER, volume 11157 of *LNCS*, 2018, pp. 219–235.

[7] F. Mannhardt, M. de Leoni, H. A. Reijers, W. van der Aalst, Decision mining revisited: Discovering overlapping rules, in: 28th CAiSE, volume 9694 of *LNCS*, 2016, pp. 377–392.

[8] A. Deutsch, R. Hull, Y. Li, V. Vianu, Automatic verification of database-centric systems, ACM SIGLOG News 5 (2018) 37–56.

[9] P. Felli, M. de Leoni, M. Montali, Soundness verification of data-aware process models with variable-to-variable conditions, Fundam. Inform. 182 (2021) 1–29.

[10] P. Felli, M. Montali, S. Winkler, Linear-time verification of data-aware dynamic systems with arithmetic, in: 36th AAAI, 2022. To appear. doi.org/10.48550/arXiv.2203.07982.

[11] P. Felli, M. Montali, S. Winkler, Soundness of data-aware processes with arithmetic conditions, in: 34th CAiSE, 2022. To appear. doi.org/10.48550/arXiv.2203.14809.

[12] P. Felli, M. Montali, S. Winkler, CTL* model checking for data-aware dynamic systems with arithmetic, in: 11th IJCAR, 2022. To appear.

[13] W. van der Aalst, The application of Petri Nets to workflow management, J. Circuits Syst. Comput. 8 (1998) 21–66.

[14] M. de Leoni, P. Felli, M. Montali, Strategy synthesis for data-aware dynamic systems with multiple actors, in: Proc. 17th KR, 2020, pp. 315–325.

[15] D. Calvanese, S. Ghilardi, A. Gianola, M. Montali, A. Rivkin, SMT-based verification of data-aware processes: a model-theoretic approach, Math. Struct. Comput. Sci. 30 (2020) 271–313.