Enhancing EB4EB Framework by Introducing Lexicographic Variants

Peter Rivière[1], Yamine Aït Ameur[2], Toshiaki Aoki[1], Guillaume Dupont[2], Neeraj Kumar Singh[2], Takashi Tomita[1], Duong Dinh Tran[1]

[1] *JAIST - Japan Advanced Institute of Science and Technology, Ishikawa, Japan*
[2] *INPT-ENSEEIHT/IRIT, University of Toulouse, France*
{priviere, toshiaki, tomita, duongtd, }@jaist.ac.jp
{yamine, guillaume.dupont, nsingh}@enseeiht.fr

The EB4EB framework (Event-B for Event-B) [4,5] has been developed to extend the capabilities of Event-B. It enables the manipulation of Event-B machine concepts by providing a formalisation of Event-B as first-class elements within an algebraic theory, combined with the standard use of contexts and machines.

This framework has already been used to formalise new proof obligations that can be generated for instances of the framework, thereby supporting additional analyses [7,3,6]. In its early development, the framework only formalised states and events, which limited its expressiveness compared with the classical use of Event-B machines. One of these limitations has been addressed in a more recent version of EB4EB, which now supports the formalisation of event parameters [8].

Another limitation concerns the treatment of variants. The initial versions of EB4EB adopted the notion of variants as presented in the Event-B book [1]. However, with the evolution of the Rodin platform, Event-B has been extended to support lexicographic variants, enabling the handling of convergence properties that cannot be captured by simple (primitive) functions [2].

Lexicographic variant. We propose a new formalisation of the EB4EB framework that supports lexicographic variants, together with an adaptation of the liveness rules for temporal properties introduced in [6] to accommodate such variants.

Listings 1 presents the data type used to capture lexicographic variants as an inductive structure, `DTVariant`. The lexicographic ordering is defined according to the structure of this induction. The constructor `NilVar` represents the base case, corresponding to the empty variant. The constructor `addVar` takes three arguments: the current variant, accessible via the `head` operator; the remaining variants of lower priority, defined recursively and accessible via the `tail` operator; and a set specifying the domain of definition of the variant. This last argument allows, for example in EB4EB, the domain of definition to be constrained by the invariant.

```
THEORY VariantTheo
TYPE PARAMETERS S
DATATYPE
    DTVariant ≙
     | NilVar
     | addVar(
         head : ℙ(S × ℤ),
```

$tail : DTVariant(S)$,
$setDef : \mathbb{P}(S)$)

Listing 1. Lexicographic Datatype

From this data type, we define the corresponding ordering, and in particular a set of predicate operators that are essential for characterising convergence and divergence: `Decreased`, `Natural`, and `NotIncreased`.

These operators are presented in Listing 2, and their definitions rely on an inductive structure combined with the standard ordering on natural numbers.

```
OPERATORS
    WellConsDTVariant ...
    NotNilVar ...
    Natural ...
    NotIncreased ...
    Decreased (set : ℙ(S), dtvar : DTVariant(S), s : S, sp : S)
        well−definedness condition
            WellConsDTVariant(set, dtvar) ∧ s ∈ set ∧ sp ∈ set
        recursive definition
            case dtvar
                Decreased(set, NilVar, s, sp) ≙ ⊤
                Decreased(set, addVariant(v, t, se), s, sp) ≙
                    v(s) > v(sp)
                    ∨(v(s) ≥ v(sp) ∧ Decreased(set, t, s, sp) ∧ NotNilVar(t))
    ...
```

Listing 2. Operators for the Lexicographic variant

Temporal Operator In the current definition of the temporal operators, for the divergence case, one premise of the proof obligation requires the variant in the after-state to be `Natural`. In this situation, the event should not increase the variant. By strengthening this premise, we obtain a weaker form of the proof obligation, which simplifies the proof process.

However, this approach is only valid for primitive variants. Indeed, the property that the variant is `Natural` in the after-state implies that it is also `Natural` in the before-state holds for primitive variants, but does not necessarily hold for lexicographic variants.

Conclusion The new release of EB4EB supports the definition of lexicographic variants and also enables a more intuitive modelling style when no variant is required, by using the empty base case. This new approach makes it possible to identify and correct issues in previous analyses, and has been successfully applied to simplify the proof of the Bakery algorithm.

References

1. Abrial, J.R.: Modeling in Event-B: System and software engineering. Cambridge University Press (2010)
2. Ackermann, W.: Zum hilbertschen aufbau der reellen zahlen. Mathematische Annalen **99**(1), 118–133 (1928)
3. Mendil, I., Riviere, P., Aït Ameur, Y., Singh, N.K., Méry, D., Palanque, P.A.: Non-intrusive annotation-based domain-specific analysis to certify event-b models behaviours. In: 29th Asia-Pacific Software Engineering Conference, APSEC. pp. 129–138. IEEE (2022)

4. Riviere, P., Singh, N.K., Aït Ameur, Y.: EB4EB: A Framework for Reflexive Event-B. In: International Conference on Engineering of Complex Computer Systems, ICECCS 2022. pp. 71–80. IEEE (2022)
5. Riviere, P., Singh, N.K., Aït Ameur, Y.: Reflexive Event-B: Semantics and Correctness the EB4EB Framework. IEEE Transactions on Reliability pp. 1–16 (2022)
6. Riviere, P., Singh, N.K., Aït Ameur, Y., Dupont, G.: Formalising liveness properties in event-b with the reflexive EB4EB framework. In: NFM. Lecture Notes in Computer Science, vol. 13903, pp. 312–331. Springer (2023)
7. Riviere, P., Singh, N.K., Aït-Ameur, Y., Dupont, G.: Standalone Event-B models analysis relying on the EB4EB meta-theory. In: ABZ. Lecture Notes in Computer Science, vol. 14010, pp. 193–211. Springer (2023)
8. Rivière, P., Singh, N.K., Aït-Ameur, Y., Dupont, G.: Extending the EB4EB framework with parameterised events. Sci. Comput. Program. **243**, 103279 (2025). https://doi.org/10.1016/J.SCICO.2025.103279, `https://doi.org/10.1016/j.scico.2025.103279`