Reviews follow-up for "Certifying expressive power and algorithms of Reversible Primitive Permutations with LEAN"

General remark. The length of the text has grown mainly because: - we answered referee's observations and questions exhaustively;

- we reformatted the text in few points, to make it more readable, sometimes adding itemize environment;
- we added a new figure, again to ease readability;
- we added references.

Reponse to Observations/Questions

1. **Observation/Question**

- abstract, l. 11: I would remove the sentence about the novel contribution from the abstract, if you want you can move it to the intro at lines 93-95 where you discuss the novelty of your paper

Done. We have just dropped the sentence from the abstract. Lines 98-100 already underline how this work extends the one of RC2022.

2. **Observation/Question**

- p. 1, l. 7-9: do you have any citation to refer to here?

Done. References [1], [2], [3], and [4] added (p.1, l.7–8).

- p. 2, l. 37: it seems to me you never come back to this. Also, what is the fix-point problem?

Done. Statement and meaning of fix-point problem recalled (p.2, l.39–42)

3. **Observation/Question**

- p. 4, l. 106: in in

Done.

4. **Observation/Question**

- p. 5, fig. 2: please use code font for It (and maybe also f) in the comment, otherwise it seems simple text

Done. This is not a simple request: the package `lstlisting` takes under control which font to use and where. We add 'It f' and 'f' in the comment of Fig.2

5. **Observation/Question**

- p. 5, l. 129: strange space at the beginning of the row

Done.

6. **Observation/Question**

- p. 7, l. 152: is this standard Lean syntax? Or did you defined it?

Done. See p. 7, l. 153 - 169 with a new description.

7. **Observation/Question**

- p. 7, l. 160: property -> properties

Done.

8. **Observation/Question**

- p. 8, l. 197-198: something seems missing here: there is a both followed by a single item, and also the code seems missing something. Indeed, I was expecting an equality (or maybe this is just my poor understanding of Lean)

Done. See p. 8, l. 214 - 218, starting from sentence "This can be restated . . . ". We have cleaned up the text.

9. **Observation/Question**

- p. 8, l. 199: I understand your proof technique (and it is interesting), but could you also have used copy-cat from the previous one?

Done. See p. 9, l. 221 - 222 to read a brief answer to this question.

10. **Observation/Question**

- p. 9, l. 237: do you miss arguments x0 . . . xn-2 ?

Done. See p. 10, l. 259 where we added the missing arguments. Thanks.

11. **Observation/Question**

- p. 10, l. 260: could you give an intuition about how the general schema works?

Done. We think we now give both intuition and technical details to appreciate how the general scheme represented by `step (_)` works. Specifically:

- p. 11, l. 283 - 289 and Fig.7: we give the idea about how `step (_)` has to work;
- p. 11, l. 290 - 291: read the meaning of Fig. 7
- p. 11, l. 292 - 303: justify why the definition of `step (_)` in Fig.8 looks more involved than expected: the reason is that we are in a reversible setting and we cannot afford to erase values.

Factoring out a general behavior of `step (_)`, allows us to slightly improve the presentation of Cantor (Un-)Pairing, Quotient and reminder, from line 309 to line 341 across pages 12 - 14.

12. **Observation/Question**

- p. 13, l. 299 - 302: since it is relevant for your proof you should provide a description

Done. p. 14, l. 345 - 346: as far as we could understand, the referee asked for the analytical definition of `mkpair`. We now we give it explicitly.

13. **Observation/Question**

- p. 15, l. 360: why do you require forall z, instead of just using 0 for z?

In relation to this question we choose not to modify the text. We do not use just 0, in place of z, because our statement is slightly more general, and its proof is not substantially more complex.

14. **Observation/Question**

- p. 16, Rem. 6: could you provide a general proof, for each pair of packing/unpacking functions satisfying some suitable requirements?

In relation to this question we choose not to modify the text. We interpret the question as a curiosity, not as a request to prove `theorem completeness` (p. 17, l. 441) with the function `cp` in place of `mkpair` in LEAN. Of course `cp` can replace `mkpair`: both functions have identical interface. Clearly, the two isomorphisms have different graphs (seen as mathematical functions). The point is that the proof of `theorem completeness` does not rely on a specific strategy to encode pairs of values into a single value.

15. **Observation/Question**

- p. 18, l. 424: need -> needs

Done.

16. **Observation/Question**

- p. 18, l. 431: if you choose different immersions which differ for some non primitive recursive automorphisms, would you get different notions of encodable? If so, in which sense the one you get is good?

Done. See p. 19, l. 474 - 481: We recognized that our argument was not sufficiently clear. To provide further precision, we emphasize that mathlib library in LEAN does not permit exiting from primitive recursive definitions, when starting from primitive recursive ones.

17. **Observation/Question**

- p. 19, l. 449: could you clarify why the class mechanism is useful here, and how?

Done. We think we have clarified the point by explaining the role of the class mechanism in LEAN. (p. 19, l.474 - 478)

18. **Observation/Question**

- p. 19, l. 468: are are

Done.

19. **Observation/Question**

- p. 20, l. 499: Lean tactics mode with by refl: this sentence seems ungrammatical; also, could you clarify what is refl?

Done. We have reorganized the paragraph at p. 21, l. 544 - 546 better explaining the role of LEAN tactic “refl”.

20. **Observation/Question**

- p. 21, l. 517: into two steps -> in two steps

Done.

21. **Observation/Question**

- p. 23, l. 587: could you provide a citation for Pendulum ISA?

References [25, 26, 27] added.

22. **Observation/Question**

- At the end of section 2 the authors claim that It is more primitive, but their arguments are not convincing without having also a definition of It in terms of ItR.

Done. Very likely there is a misunderstanding about: “an operator being more primitive than another one”. Our solution is to avoid stating that our iterator “It” is *more primitive.* We just say that it subsumes the behavior of “ItR”, “for”, and we explain why. (p. 9, l. 253 - 255)

1. **Observation/Question**

- Line 311. “here” should be removed.

Done.

24. **Observation/Question**

- LEAN 4 releases are available (non the stable one). May be a small example with C code could be added.

Done. We investigated a little bit more deeply what the designers of LEAN4 promise. We realized that LEAN4 does not directly export C code. So, we cannot add any C code snippet to our work. In fact, any reference to LEAN4 sounds useless, so we decided to drop any reference to it.