fn then_deintegerize_vec

Michael Shoemate

This proof resides in **"contrib"** because it has not completed the vetting process.

Proves soundness of the implementation of `then_deintegerize_vec` in `mod.rs` at commit f5bb719 (outdated[1]).

1 Hoare Triple

Precondition

Compiler-Verified

- Generic `T0` implements trait `CastInternalRational`

User-Verified

None

Pseudocode

```
def then_deintegerize_vec(k: i32) -> Function[Vec[IBig], Vec[T0]]:

    if k == i32.MIN: #
        raise ValueError("k must be greater than i32.MIN")

    def element_function(x_i):
        return T0.from_rational(x_mul_2k(RBig.from_(x_i), k))

    return Function.new(lambda x: [element_function(x_i) for x_i in x])
```

Postcondition

Theorem 1.1. For every setting of the input parameters (`k, T0`) to `then_deintegerize_vec` such that the given preconditions hold, `then_deintegerize_vec` raises an exception (at compile time or run time) or returns a valid postprocessor. A valid postprocessor has the following property:

1. (Data-independent errors). For every pair of elements x, x' in `input_domain`, `function`(x), `function`(x') either neither or both raise an error. If both raise an error, then they both raise the same error.

Proof. By the postcondition of `T0.from_rational`, the outcome of the function is the nearest representable float, and may saturate to positive or negative infinity. The precondition of `x_mul_2k` that k is not `i32.MIN` is satisfied on line 3. Since `T0.from_rational` and `x_mul_2k` are both infallible, the function is infallible, meaning that the function cannot raise data-dependent errors. Therefore the function is a valid postprocessor. □

[1] See new changes with `git diff f5bb719..dfb22e0c rust/src/measurements/noise/nature/float/mod.rs`