A theory for defining and handling matrices in Event-B

Guillaume Dupont[1]

INPT-ENSEEIHT/IRIT, University of Toulouse, France
guillaume.dupont@enseeiht.fr

The formalisation of certain systems, especially cyber-physical systems and quantum algorithm, has given rise to the necessity of handling *matrices* in various ways. Matrices pose an interesting challenge in term of modelling as most of their associated operators (sum, multiplication, etc.) require a form of iteration/recursion over natural numbers, the dimensions of the matrices.

In this paper, we present a new set of theories for handling matrices of complex numbers. For that, we have used the previous work of Dominique Cansell on recursion [1], that has been formalised in a theory as well. We note that the matrices presented here may be adapted to other need; we decided against making the theory generic as it would require a lot of boiling plate parameters to generalise multiplication and addition.

Recursion in Event-B theories When it comes to iterating over natural numbers, a first intuition is to redefine them using Peano's arithmetic (inductive type with zero + successor), enabling the use of the theory plug-in's recursively defined operators. Although this is an option, it considerably hinders usage and proof when it comes to matrices, where checking for number equality and so on is extremely common, and quite cumbersome using bare, not tool-supported Peano's arithmetic.

For this reason, we decided to define matrices using Event-B's native natural numbers, which is completely supported by the modelling and proving infrastructure of Rodin.

This comes with a significant downside, compared to Peano's arithmetic: natural numbers are not inductively defined in Event-B, making it impossible to define classical recursive algorithms involving them.

To overcome this issue, we turned to the recent work of Cansell on recursive functions [1]. In said work, the author has essentially formalised recursion from scratch (in a constructive way), and has provided a way to define recursive functions on well-founded sets, which fortunately $\mathbb{N}$ is.

To achieve our formalisation of matrices, we have modelled most of the work of Cansell in a theory, and have proven a few of the theorems proposed in his workshop paper. This led to the definition of a general `fold` operator, that represents general iteration over an ordered set, applying a reducing function on the elements of that set.

Complex numbers As mentioned previously, we focus on the definition of matrices of *complex numbers*. This in turns requires the definition of complex numbers in a specific theory, based on the theory of reals we defined and used in our work[1].

Although real numbers are defined axiomatically, we defined complex numbers constructively in the traditional way (essentially, as pairs of real numbers). The usual operations are defined on the type, and we may proof that we have defined a *field*.

Importantly, we define a generalised sum of complex numbers ($\sum_{i=0}^{n} c_i$) using the previously defined `fold` operator, which will be useful for defining matrix multiplication. A number of theorems and proof rules are added to the theory, useful to reason on complex numbers in a fairly transparent way.

Matrices Matrices are essentially defined as function of natural coordinates to the set of complex numbers ($\mathbb{N} \times \mathbb{N} \pfun \mathbb{C}$) plus a height and a width. As the bare type is not sufficient, we leverage WD conditions of each operator to make sure we handle well-defined matrices (e.g., the support function is defined for each coordinate within $1..width \times 1..height$).

Addition, multiplication, tensor product, transposition, scalar multiplication, and adjunction are defined, as well as a few constructors for particular matrices (identity, zero, etc.). Additionally, we defined the calculation of the *trace* of a matrix, as well as a general sum operator (using `fold`, again).

We have also defined the notion of eigenvectors and eigenvalues, although not in a way that makes it possible to calculate them (only check if a particular value-vector pair is eigen).

Since the goal was to define quantum computing elements, we have formalised the notion of *bra* and *ket*, of inner and outer product, as well as the set of unitary matrices. Once again, a number of proof rules are defined to help the user handle basic aspects of matrix calculus (calculating dimensions, etc.).

Limitations and future work Some concepts are still missing from the theory, either because we did not need them or because it was too difficult to model (e.g., the determinant of a matrix). Although most proofs have been carried out, there remain some tricky bits of proof that are still to be done.

Once the theories are fully proven (and cleaned up a bit), we plan to provide them on a repository for users to get. Meanwhile, any interested reader may send a mail to the author to get a copy of the set of theories.

Last, we note that the theories are tailored for complex numbers; but the modeling technique used here can easily be generalised to matrices of any type of element.

References

1. Cansell, D.: Schemata of recursive functions and iterative algorithms. In: The 11th Rodin User and Developer Workshop, 25th June, 2024, Bergamo, Italy (2024), `https://wiki.event-b.org/index.php/Rodin_Workshop_2024`

[1] `https://www.irit.fr/~Guillaume.Dupont/models/#theories`