Regular expressions are a syntactic means to describe exactly the class of languages accepted by deterministic finite state automata, i.e. the class of regular languages. Ignoring the initial state for the moment, a deterministic automata is a set of states equipped with function determining for each state whether or not it is final, and assigning for each input symbol a next state.

There is a well-known correspondence between regular expressions and deterministic automata. For instance the automata

\[
\begin{array}{c}
\text{a} \\
\text{b} \\
\end{array}
\begin{array}{c}
\text{a} \\
\text{b} \\
\end{array}
\]

accepts the same language as described by the expression \( b^*a(a + bb^*a)^* \). There are methods to obtain a regular expression that accepts the same language as a given automaton and vice-versa.

Deterministic automata can be generalized to coalgebras for an endo-functor \( G \) on the category \( \text{Set} \). A coalgebra is pair \( (S, g) \) consisting of a set of states \( S \) and a transition function \( g : S \rightarrow GS \), where the functor \( G \) determines the type of the dynamic system under consideration.

For polynomial set functors \( G \), we generalize the notion of regular expressions and introduce a language of expressions for describing elements of the final \( G \)-coalgebra. We show that every state of a finite \( G \)-coalgebra corresponds to an expression in the language, in the sense that they both have the same semantics. Conversely, we give a compositional synthesis algorithm which transforms every expression into a finite \( G \)-coalgebra. The language of expressions is equipped with an equational system that is sound, complete and expressive with respect to \( G \)-bisimulation.