The text is entitled "Monad Plus".
The article is devoted to Monad Plus – monads with additional laws. It is spoken in detail about definition and examples. Monad Plus could be used in parallel parsing, another approach is to use it to define guard function. The fact that List and Maybe are in Monad Plus class is stressed.

`https://en.wikibooks.org/wiki/Haskell/MonadPlus`

The text is entitled "All About Monads".
The paper is concerned with monads. The text gives a detailed account of definition of monads, describing its construction and applications of monads in real programs. Much attention is given to show internal structure of monads and why we need this concept. The article has a lot of examples of using List, Maybe, IO, State and many other monads. In the end of the paper it gives a detailed analysis of combinations of monads.

`https://wiki.haskell.org/All_About_Monads`

The next annotations are about chapters of the book:
"Thinking functionally with Haskell", Richard Bird

The chapter 1 is entitled "What is functional programming?".
The text is concerned with explanation of concept of functional programming using examples from math functions. It is considered in detail program which shows most common words in the text. It is shown how to install The Haskell Platform to use Haskell on your computer.

The chapter 2 is entitled "Expressions, types and values".
The text gives a detailed account of ghci. It explains Glasgow Haskell Compiler or ghc with ghci – Haskell interpreter. As the title implies the chapter describes expressions and operators using in them. It also

shows operator sections and lambda functions – tools that allow us to write shorter and more clear programs. Much attention is given to Haskell type structure with type inference system.

The chapter 3 is entitled "Numbers".
As the title implies the chapter describes numbers where are Integral (Int, Integer) and Floting (Float, Double) number types in Haskell. It gives a detailed analysis of Num type class, which helps us to discribe numeric typs i.e. types whith $+, -, *$ operations. It is shown the whole hierarchy of Haskell numeric data types.

The chapter 4 is entitled "Lists".
The chapter is concerned with lists. It provides list data type definition and shows three types of lists: finite, partial, infinite. The main idea of the chapter is list constructors which includes enumerators and list comprehensions. Then it draws our attention to main list functions: map, filter, zipWith, which help us replace cycles from imperative languages in Haskell.

The chapter 5 is entitled "A simple Sudoku solver".
As the title implies the chapter describes how to write Sudoku solver whith Haskell. The text gives a detailed analysis of data types using for describe cells and board. It is shown how to write Sudoku solver, using composition of simple functions, working with lists.

The chapter 6 is entitled "Proofs".
The main idea of the article is that list functions are the same as induction. The article begins with describing induction over natural numbers. Than it shows induction over lists: finite, partial and infinite.In conclusion the author proves correctness of some function definitions, using folding functions.

The chapter 7 is entitled "Efficiency".
The chapter is concerned with efficiency of program evaluation and computations. It gives a detailed analysis of lazy evaluation – evaluation strategy used in Haskell by default. Much attention is given to controlling space and time used by program. It shows in detail ghc's profiling tools, but it also noted that the best way to improve a programs performance is to use a better algorithm.

The chapter 8 is entitled "Pretty-printing".
This chapter is devoted to an example of how to build a small library in Haskell. The main idea of the text is to show steps of building pretty-printing library or embedded domain specific language (EDSL). In conclusion the author says that the growing success of Haskell is due to its ability to host a variety of EDSLs without fuss.