

---

## Preface

The J programming language is rich in mathematical functionality and ideally suited to analytical computing methods. It is, however, a somewhat terse language and not entirely intuitive at first, particularly if one is used to more conventional programming languages such as C, Pascal or Java. J functions (called *verbs*) are denoted by punctuation symbols. New functions can be developed by combining sequences of existing verbs into *phrases*. The compositional rules of J govern how the functions are combined and thus how they are applied to data objects. *Conjunctions* provide the programmer with additional control over the behavior of verbs and the composition of phrases.

The J programming language was conceived by Kenneth Iverson (also the author of APL) and Roger Hoi in 1991 (note that J is not in anyway related to Java). It is an interpretive language written in C. It is, therefore, highly portable and can run on a number of architectures and operating systems. Architectures currently supported are Intel, PowerPC and ARM, as well as 64 bit AMD. There are runtime versions of J for a number of Unix variants, such as Linux, Mac OS X, and Solaris, for example. Windows and Windows CE are also supported.

The last platform, Windows CE, is of particular interest. If, like the author, you have tried a number of scientific calculators, you may have found them packed with functionality but cumbersome to use. PDAs (personal digital assistants), on the other hand, have better user interfaces. They are also quite powerful and have relatively high-resolution colour graphics. However, in terms of mathematical features, they are somewhat limited and are little more than electronic organisers. For an engineer then, J is the “killer app” for the PDA. A PDA (running Windows CE) becomes the ultimate scientific calculator. In this environment, a terse language like J is a benefit, as the input method is typically limited to a touch screen with a stylus. Programming in more verbose languages, like C or Java makes the PDA impractical.

In this book, we use the J programming language as a tool for carrying out performance analysis of packet data networks. All the examples in this book were developed on the J.504 version of the interpreter. There are many excellent books on the

market on network performance analysis. However, they tend to be heavy in mathematical rigour. Rather than covering these topics in terms of proofs and theorems (which the other books do so well anyway), it is the aim of this book to introduce concepts and principles of network performance analysis by example, using J.

J is available for download from [www.jsoftware.com](http://www.jsoftware.com), where installation instructions are provided as well as online documentation. There are also a number of forums that you may subscribe to that will give you access to the wealth of experience and knowledge of the J community.

## Acknowledgments

The author would like to thank the following people for the valuable contribution they made to this book: Dr Lin-Yi Chou (University of Waikato), Dr Sue Casson (Leeds University), Dr Adrian Davies, Michael Dewsnip (DL Consulting), Dr Chi-Yu Huang (DL Consulting), Zhiwei Liu (University of Waikato), and Prof. John Monk (Open University). Thanks also to Catherine Brett of Springer for guiding me through this process.

I especially want to thank my wife for all her love and encouragement throughout this period. I could not have undertaken this project without her support.

Alan Holt



<http://www.springer.com/978-1-84628-822-7>

Network Performance Analysis  
Using the J Programming Language

Holt, A.

2008, XVI, 216 p., Hardcover

ISBN: 978-1-84628-822-7