

High-Performance Computing Series

Daisuke Takahashi

Fast Fourier Transform Algorithms for Parallel Computers



Springer

High-Performance Computing Series

Volume 2

Series Editor

Satoshi Matsuoka, RIKEN Center for Computational Science, Kobe, Hyogo, Japan

The series publishes authored monographs, textbooks, and edited state-of-the-art collections covering the whole spectrum of technologies for supercomputers and high-performance computing, and computational science and engineering enabled by high-performance computing series (HPC).

Areas of interest include, but are not limited to, relevant topics on HPC:

- Advanced algorithms for HPC
- Large-scale applications involving massive-scale HPC
- Innovative HPC processor and machine architectures
- High-performance / low-latency system interconnects
- HPC in the Cloud
- Data science / Big Data / visualization driven by HPC
- Convergence of machine learning / artificial intelligence with HPC
- Performance modeling, measurement, and tools for HPC
- Programming languages, compilers, and runtime systems for HPC
- Operating system and other system software for HPC
- HPC systems operations and management

More information about this series at <http://www.springer.com/series/16381>

Daisuke Takahashi

Fast Fourier Transform Algorithms for Parallel Computers

Daisuke Takahashi
University of Tsukuba
Tsukuba, Japan

ISSN 2662-3420 ISSN 2662-3439 (electronic)
High-Performance Computing Series
ISBN 978-981-13-9964-0 ISBN 978-981-13-9965-7 (eBook)
<https://doi.org/10.1007/978-981-13-9965-7>

© Springer Nature Singapore Pte Ltd. 2019

This work is subject to copyright. All rights are reserved by the Publisher, whether the whole or part of the material is concerned, specifically the rights of translation, reprinting, reuse of illustrations, recitation, broadcasting, reproduction on microfilms or in any other physical way, and transmission or information storage and retrieval, electronic adaptation, computer software, or by similar or dissimilar methodology now known or hereafter developed.

The use of general descriptive names, registered names, trademarks, service marks, etc. in this publication does not imply, even in the absence of a specific statement, that such names are exempt from the relevant protective laws and regulations and therefore free for general use.

The publisher, the authors and the editors are safe to assume that the advice and information in this book are believed to be true and accurate at the date of publication. Neither the publisher nor the authors or the editors give a warranty, expressed or implied, with respect to the material contained herein or for any errors or omissions that may have been made. The publisher remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

This Springer imprint is published by the registered company Springer Nature Singapore Pte Ltd. The registered company address is: 152 Beach Road, #21-01/04 Gateway East, Singapore 189721, Singapore

Preface

The fast Fourier transform (FFT) is an efficient implementation of the discrete Fourier transform (DFT). The FFT is widely used in numerous applications in engineering, science, and mathematics. This book is an introduction to the basis of the FFT and its implementation in parallel computing. Parallel computation is becoming indispensable in solving the large-scale problems that arise in a wide variety of applications. Since there are many excellent books on FFT, this book focuses on the implementation details of FFTs for parallel computers. This book provides a thorough and detailed explanation of FFTs for parallel computers. The algorithms are presented in pseudocode, and a complexity analysis is provided.

The performance of parallel supercomputers is steadily improving, and it is expected that a massively parallel system with more than hundreds of thousands of compute nodes equipped with manycore processors and accelerators will be exascale supercomputers in the near future. This book also provides up-to-date computational techniques relevant to the FFT in state-of-the-art parallel computers.

This book is designed for graduate students, faculty, engineers, and scientists in the field. The design of this book intends for readers who have some knowledge about DFT and parallel computing. For several implementations of FFTs described in this book, you can download the source code from www.ffte.jp.

This book is organized as follows. Chapter 2 introduces the definition of the DFT and the basic idea of the FFT. Chapter 3 explains mixed-radix FFT algorithms. Chapter 4 describes split-radix FFT algorithms. Chapter 5 explains multidimensional FFT algorithms. Chapter 6 presents high-performance FFT algorithms. Chapter 7 explains parallel FFT algorithms for shared-memory parallel computers. Finally, Chap. 8 describes parallel FFT algorithms for distributed-memory parallel computers.

I express appreciation to all those who helped in the preparation of this book.

Tsukuba, Japan
March 2019

Daisuke Takahashi

Contents

1	Introduction	1
	References	2
2	Fast Fourier Transform	5
2.1	Definitions of DFT	5
2.2	Basic Idea of FFT	5
2.3	Cooley–Tukey FFT Algorithm	9
2.4	Bit-Reversal Permutation	10
2.5	Stockham FFT Algorithm	10
2.6	FFT Algorithm for Real Data	11
2.6.1	FFT of Two Real Data Simultaneously	11
2.6.2	n -Point Real FFT Using $n/2$ -Point Complex FFT	12
	References	13
3	Mixed-Radix FFT Algorithms	15
3.1	Two-Dimensional Formulation of DFT	15
3.2	Radix-3 FFT Algorithm	16
3.3	Radix-4 FFT Algorithm	17
3.4	Radix-5 FFT Algorithm	17
3.5	Radix-8 FFT Algorithm	18
	References	19
4	Split-Radix FFT Algorithms	21
4.1	Split-Radix FFT Algorithm	21
4.2	Extended Split-Radix FFT Algorithm	23
	References	33
5	Multidimensional FFT Algorithms	35
5.1	Definition of Two-Dimensional DFT	35
5.2	Two-Dimensional FFT Algorithm	36

5.3	Definition of Three-Dimensional DFT	37
5.4	Three-Dimensional FFT Algorithm	37
	Reference	40
6	High-Performance FFT Algorithms	41
6.1	Four-Step FFT Algorithm	41
6.2	Five-Step FFT Algorithm	43
6.3	Six-Step FFT Algorithm	44
6.4	Blocked Six-Step FFT Algorithm	46
6.5	Nine-Step FFT Algorithm	48
6.6	Recursive Six-Step FFT Algorithm	51
6.7	Blocked Multidimensional FFT Algorithms	53
6.7.1	Blocked Two-Dimensional FFT Algorithm	53
6.7.2	Blocked Three-Dimensional FFT Algorithm	53
6.8	FFT Algorithms Suitable for Fused Multiply–Add (FMA) Instructions	54
6.8.1	Introduction	54
6.8.2	FFT Kernel	55
6.8.3	Goedecker’s Technique	55
6.8.4	Radix-16 FFT Algorithm	56
6.8.5	Radix-16 FFT Algorithm Suitable for Fused Multiply–Add Instructions	59
6.8.6	Evaluation	59
6.9	FFT Algorithms for SIMD Instructions	63
6.9.1	Introduction	63
6.9.2	Vectorization of FFT Kernels Using Intel SSE3 Instructions	64
6.9.3	Vectorization of FFT Kernels Using Intel AVX-512 Instructions	65
	References	66
7	Parallel FFT Algorithms for Shared-Memory Parallel Computers	69
7.1	Implementation of Parallel One-Dimensional FFT on Shared-Memory Parallel Computers	69
7.1.1	Introduction	69
7.1.2	A Recursive Three-Step FFT Algorithm	70
7.1.3	Parallelization of Recursive Three-Step FFT	72
7.2	Optimizing Parallel FFTs for Manycore Processors	72
7.2.1	Introduction	72
7.2.2	Parallelization of Six-Step FFT	73
7.2.3	Performance Results	74
	References	76

8	Parallel FFT Algorithms for Distributed-Memory Parallel Computers	77
8.1	Implementation of Parallel FFTs in Distributed-Memory Parallel Computers	77
8.1.1	Parallel One-Dimensional FFT Using Block Distribution	77
8.1.2	Parallel One-Dimensional FFT Using Cyclic Distribution	79
8.1.3	Parallel Two-Dimensional FFT in Distributed-Memory Parallel Computers	81
8.1.4	Parallel Three-Dimensional FFT in Distributed-Memory Parallel Computers	82
8.2	Computation–Communication Overlap for Parallel One-Dimensional FFT	83
8.2.1	Introduction	83
8.2.2	Computation–Communication Overlap	84
8.2.3	Automatic Tuning of Parallel One-Dimensional FFT	84
8.2.4	Performance Results	87
8.3	Parallel Three-Dimensional FFT Using Two-Dimensional Decomposition	88
8.3.1	Introduction	88
8.3.2	Implementation of Parallel Three-Dimensional FFT Using Two-Dimensional Decomposition	89
8.3.3	Communication Time in One-Dimensional Decomposition and Two-Dimensional Decomposition	92
8.3.4	Performance Results	93
8.4	Optimization of All-to-All Communication on Multicore Cluster Systems	96
8.4.1	Introduction	96
8.4.2	Two-Step All-to-All Communication Algorithm	97
8.4.3	Communication Times of All-to-All Communication Algorithms	98
8.4.4	Performance Results	99
8.5	Parallel One-Dimensional FFT in a GPU Cluster	102
8.5.1	Introduction	102
8.5.2	Implementation of Parallel One-Dimensional FFT in a GPU Cluster	103
8.5.3	Performance Results	106
	References	109
	Index	113

Chapter 1

Introduction



Abstract The fast Fourier transform (FFT) is an efficient implementation of the discrete Fourier transform (DFT). The FFT is widely used in numerous applications in engineering, science, and mathematics. This chapter describes the history of the FFT briefly and presents an introduction to this book.

Keywords Discrete Fourier transform (DFT) • Fast Fourier transform (FFT) • Parallel processing

The fast Fourier transform (FFT) is a fast algorithm for computing the discrete Fourier transform (DFT).

The fast algorithm for DFT can be traced back to Gauss's unpublished work in 1805 [11]. Since the paper by Cooley and Tukey [6] was published in 1965, the FFT has become to be widely known. Then, Gentleman and Sande presented a variant of the Cooley-Tukey FFT algorithm [10]. In the Cooley-Tukey FFT algorithm, the input data is overwritten with the output data (i.e., in-place algorithm), but bit-reversal permutation is required. It is also possible to construct an out-of-place algorithm that stores input data and output data in separate arrays. Stockham algorithm [5] is known as an out-of-place algorithm and it does not require bit-reversal permutation. Bergland proposed an FFT algorithm for real-valued series [4]. Yavne [18] presented a method that is currently known as the split-radix FFT algorithm [7]. Singleton proposed a mixed-radix FFT algorithm [16]. As FFT algorithms of a different approach from the Cooley-Tukey FFT algorithm, Rader proposed an FFT algorithm that computes the DFTs when the number of data samples is prime [15]. Kolba and Parks proposed a prime factor FFT algorithm (PFA) [13]. Winograd extended Rader's algorithm and proposed a Winograd Fourier transform algorithm (WFTA) that can be applied to the DFTs of the powers of prime numbers [17]. Bailey proposed a four-step FFT algorithm and a six-step FFT algorithm [2]. Johnson and Frigo proposed a modified split-radix FFT algorithm [12], which is known as the FFT algorithm with the lowest number of arithmetic operations.

As early studies of parallel FFTs, Pease proposed an adaptation of the FFT for parallel processing [14]. Moreover, Ackins [1] implemented an FFT for the ILLIAC IV parallel computer [3]. Since then, various parallel FFT algorithms and

implementations have been proposed. Frigo and Johnson developed the FFTW (The fastest Fourier transform in the west), which is known as the fastest free software implementation of the FFT [8, 9].

Examples of FFT applications in the field of science are the following:

- Solving partial differential equations,
- Convolution and correlation calculations, and
- Density functional theory in first principles calculations.

Examples of FFT applications in the field of engineering are the following:

- Spectrum analyzers,
- Image processing, for example, in CT scanning and MRI, and
- Modulation and demodulation processing in orthogonal frequency multiplex modulation (OFDM) used in wireless LAN and terrestrial digital radio and television broadcasting.

The rest of this book is organized as follows. Chapter 2 introduces the definition of the DFT and the basic idea of the FFT. Chapter 3 explains mixed-radix FFT algorithms. Chapter 4 describes split-radix FFT algorithms. Chapter 5 explains multi-dimensional FFT algorithms. Chapter 6 presents high-performance FFT algorithms. Chapter 7 explains parallel FFT algorithms for shared-memory parallel computers. Finally, Chap. 8 describes parallel FFT algorithms for distributed-memory parallel computers. Performance results of parallel FFT algorithms on parallel computers are also presented.

References

1. Ackins, G.M.: Fast Fourier transform via ILLIAC IV. Illiac IV Document 198, University of Illinois, Urbana (1968)
2. Bailey, D.H.: FFTs in external or hierarchical memory. *J. Supercomput.* **4**, 23–35 (1990)
3. Barnes, G.H., Brown, R.M., Kato, M., Kuck, D.J., Slotnick, D.L., Stokes, R.A.: The ILLIAC IV computer. *IEEE Trans. Comput.* **C-17**, 746–757 (1968)
4. Bergland, G.D.: A fast Fourier transform algorithm for real-valued series. *Commun. ACM* **11**, 703–710 (1968)
5. Cochran, W.T., Cooley, J.W., Favin, D.L., Helms, H.D., Kaenel, R.A., Lang, W.W., Maling, G.C., Nelson, D.E., Rader, C.M., Welch, P.D.: What is the fast Fourier transform? *IEEE Trans. Audio Electroacoust.* **15**, 45–55 (1967)
6. Cooley, J.W., Tukey, J.W.: An algorithm for the machine calculation of complex Fourier series. *Math. Comput.* **19**, 297–301 (1965)
7. Duhamel, P., Hollmann, H.: Split radix FFT algorithm. *Electron. Lett.* **20**, 14–16 (1984)
8. Frigo, M., Johnson, S.G.: FFTW: an adaptive software architecture for the FFT. In: *Proceedings of 1998 IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP '98)*, vol. 3, pp. 1381–1384 (1998)
9. Frigo, M., Johnson, S.G.: The design and implementation of FFTW3. *Proc. IEEE* **93**, 216–231 (2005)
10. Gentleman, W.M., Sande, G.: Fast Fourier transforms: for fun and profit. In: *Proceedings of AFIPS '66 Fall Joint Computer Conference*, pp. 563–578 (1966)

11. Heideman, M.T., Johnson, D.H., Burrus, C.S.: Gauss and the history of the fast Fourier transform. *IEEE ASSP Mag.* **1**, 14–21 (1984)
12. Johnson, S.G., Frigo, M.: A modified split-radix FFT with fewer arithmetic operations. *IEEE Trans. Signal Process.* **55**, 111–119 (2007)
13. Kolba, D.P., Parks, T.W.: A prime factor FFT algorithm using high-speed convolution. *IEEE Trans. Acoust. Speech Signal Process* **ASSP-25**, 281–294 (1977)
14. Pease, M.C.: An adaptation of the fast Fourier transform for parallel processing. *J. ACM* **15**, 252–264 (1968)
15. Rader, C.M.: Discrete Fourier transforms when the number of data samples is prime. *Proc. IEEE* **56**, 1107–1108 (1968)
16. Singleton, R.C.: An algorithm for computing the mixed radix fast Fourier transform. *IEEE Trans. Audio Electroacoust.* **17**, 93–103 (1969)
17. Winograd, S.: On computing the discrete Fourier transform. *Math. Comput.* **32**, 175–199 (1978)
18. Yavne, R.: An economical method for calculating the discrete Fourier transform. In: *Proceedings of AFIPS '68 Fall Joint Computer Conference, Part I*, pp. 115–125 (1968)