

## Table of Contents

<b>Preface</b>	v
<b>Notations</b>	vi
<b>1 Introduction</b>	<b>1</b>
1.1 Parallelism	1
1.2 Numerical linear algebra	7
1.3 Parallel solution of linear systems	11
Bibliographic Notes	12
References for Chapter 1	13
<b>2 Models of Computation</b>	<b>15</b>
2.1 Preliminary notions	15
2.1.1 Models of sequential computation	17
2.1.2 Algorithms and computational complexity	24
2.2 Models of parallel computation	29
2.2.1 Arithmetic circuits and networks	29
2.2.2 Parallel Random Access Machines	38
2.2.3 Network models	42
2.2.4 VLSI models	44
Bibliographic Notes	44
References for Chapter 2	45
Problems	47
<b>3 Arithmetic Circuits</b>	<b>49</b>
3.1 Introduction	49
3.2 Basic linear algebra circuits	50
3.3 Circuits for general systems	54
3.3.1 Csanky's algorithm	55
3.3.2 Extensions to Csanky's result	60
3.3.3 A Monte Carlo algorithm for the solution of linear systems	63
3.3.4 Newton's method	66
3.4 Computation of the rank of a matrix	70
3.5 Circuits for special systems	82
3.5.1 Triangular matrices	82
3.5.2 Triangular Toeplitz systems	84
3.5.3 Band matrix computations	87
Bibliographic Notes	93
References for Chapter 3	94
Problems	96

**viii Table of Contents**

<b>4 PRAM Algorithms</b>	<b>99</b>
4.1 Introduction	99
4.2 Basic linear algebra operations on PRAMs	102
4.3 Linear-time algorithms	105
4.3.1 LU decomposition through Gaussian elimination	106
4.3.2 The Gauss-Jordan algorithm for matrix inversion	108
4.3.3 QR factorization	110
4.4 Polylogarithmic-time algorithms	121
4.4.1 Csanky's algorithm	122
4.4.2 A randomized PRAM algorithm for the solution of linear systems	122
4.5 Algorithms for special systems	124
4.5.1 Triangular systems	125
4.5.2 Tridiagonal systems	127
Bibliographic Notes	130
References for Chapter 4	130
Problems	131
<b>5 Distributed Algorithms</b>	<b>133</b>
5.1 Preliminaries	133
5.2 Lower bounds	140
5.2.1 Gentleman's model	140
5.2.2 Varman and Ramakrishnan's model	143
5.3 Linear-Time algorithms	144
5.3.1 Basic concepts	144
5.3.2 Matrix multiplication	147
5.3.3 Systolic solution of linear systems	150
5.4 Very fast algorithms	152
5.5 Algorithms for the solution of special linear systems	155
5.5.1 Solving a banded linear system on distributed machines	155
5.5.2 Solving the recurrences governing the solution of banded systems	159
5.5.3 Conjugate gradient method	164
Bibliographic Notes	167
References for Chapter 5	168
Problems	170
<b>6 VLSI Networks</b>	<b>173</b>
6.1 Preliminaries	173
6.2 VLSI complexity	176
6.2.1 Thompson's model	177
6.2.2 Other models	178
6.2.3 Area-Time complexity	178

6.2.4 Lower bounds for matrix inversion	185
6.3 Basic linear algebra operations	188
6.3.1 Scalar product	188
6.3.2 Matrix by vector multiplication	189
6.4 Solution of general linear systems	194
6.4.1 Elimination methods	194
6.4.2 Very fast algorithms	198
6.5 Solution of tridiagonal linear systems	199
6.5.1 A lower bound	200
6.5.2 An upper bound	201
Bibliographic Notes	202
References for Chapter 6	203
Problems	205

## Bibliography

207

## Subject Index

215

In this book we actually deal with models of parallel computation. We give descriptions of computational environments. In fact, in this book we take account four different parallel environments, namely arithmetic circuits (Chapter 2), Parallel Random Access Machines or, simply, PRAM machines (Chapter 4), distributed systems (Chapter 5), and VLSI networks (Chapter 6), and we analyze problems of solving linear systems assuming that the computations are governed by rules specified by these models.

When using the term parallelism, we mean a field of computer science which deals with the analysis of the inherent parallelism of the problems and the design of efficient parallel algorithms. We can illustrate the concepts of inherent parallelism of problems and of efficient parallel algorithms using two simple examples.

We have first to introduce the notions of bounded fanin and bounded fanout and of unbounded fanin and unbounded fanout operator. An operator is called bounded fanin (fanout) if it takes (computes) a fixed number (e.g., up to a small constant) of inputs (outputs). On the contrary, an operator with unbounded fanin (fanout) can take (compute) as many inputs (outputs) as desired.

**Example 1.1.** Consider the problem of computing the sum of  $n$  numbers  $a_1, a_2, \dots, a_n$ . For simplicity, assume  $n = 2^k$ . The sum can be performed by means of a sequential algorithm, according to the standard sequential algorithm which accumulates on a single register (previously set to 0) the partial results  $s + a_i$ . Each step of this algorithm consists of one addition. Note that there are no intrinsic sequential constraints, i.e., no reason for accumulating sequentially the partial results is due to the fact that only one operation can be performed at any time.

On the contrary, if we assume to have as many processors (or operators)

as we want, the only constraint is that we must compute a single quantity (the sum). This is the basic idea of parallel computation. In this book we will study "machines" for pure parallel computing, which is intended to cover both the abstract environment (i.e., the model, typically used by theoretical computer scientists in their analyses), and the real computer.