Parallel and Distributed Computing

Alberto Paoluzzi – Lecture 21

Mon 4-05-2022

from Templates for the Solution of Linear Sustems: Building Blocks for Iterative Methoda

1. Why Use Templates in Linear Algebra?
2. Iterative Methods
3. Preconditioners
4. GraphBLAS: A linear algebraic approach for high-performance graph algorithms

Section 1

Why Use Templates in Linear Algebra?

Why Use Templates in Linear Algebra?

Templates offer three significant advantages

- First, templates are general and reusable

Second, templates exploit the expertise of two distinct groups

And third, templates are not language specific

Why Use Templates in Linear Algebra?

Templates offer three significant advantages

- First, templates are general and reusable
- Thus, they can simplify ports to diverse machines

Second, templates exploit the expertise of two distinct groups

And third, templates are not language specific

Why Use Templates in Linear Algebra?

Templates offer three significant advantages

- First, templates are general and reusable
- Thus, they can simplify ports to diverse machines
- This feature is important given the diversity of parallel architectures.

Second, templates exploit the expertise of two distinct groups

And third, templates are not language specific

Why Use Templates in Linear Algebra?

Templates offer three significant advantages

- First, templates are general and reusable
- Thus, they can simplify ports to diverse machines
- This feature is important given the diversity of parallel architectures.

Second, templates exploit the expertise of two distinct groups

- The expert numerical analyst creates a template reflecting in-depth knowledge of a specific numerical technique

And third, templates are not language specific

Why Use Templates in Linear Algebra?

Templates offer three significant advantages

- First, templates are general and reusable
- Thus, they can simplify ports to diverse machines
- This feature is important given the diversity of parallel architectures.

Second, templates exploit the expertise of two distinct groups

- The expert numerical analyst creates a template reflecting in-depth knowledge of a specific numerical technique
- The computational scientist then provides `value-added` capability to the general template description, customizing it for specific contexts or applications needs.

And third, templates are not language specific

Why Use Templates in Linear Algebra?

Templates offer three significant advantages

- First, templates are general and reusable
- Thus, they can simplify ports to diverse machines
- This feature is important given the diversity of parallel architectures.

Second, templates exploit the expertise of two distinct groups

- The expert numerical analyst creates a template reflecting in-depth knowledge of a specific numerical technique
- The computational scientist then provides `value-added` capability to the general template description, customizing it for specific contexts or applications needs.

And third, templates are not language specific

- Rather, they are displayed in an Algol-like structure

Section 2

Iterative Methods

Iterative Methods

The term `iterative method` refers to a wide range of techniques that use successive approximations to obtain more accurate solutions to a linear system at each step

- Stationary methods are older, simpler to understand and implement, but usually not as effective

Iterative Methods

The term `iterative method` refers to a wide range of techniques that use successive approximations to obtain more accurate solutions to a linear system at each step

- Stationary methods are older, simpler to understand and implement, but usually not as effective
- Nonstationary methods are a relatively recent development; their analysis is usually harder to understand, but they can be highly effective

Iterative Methods

The term `iterative method` refers to a wide range of techniques that use successive approximations to obtain more accurate solutions to a linear system at each step

- Stationary methods are older, simpler to understand and implement, but usually not as effective
- Nonstationary methods are a relatively recent development; their analysis is usually harder to understand, but they can be highly effective

Iterative Methods

The term `iterative method` refers to a wide range of techniques that use successive approximations to obtain more accurate solutions to a linear system at each step

- Stationary methods are older, simpler to understand and implement, but usually not as effective
- Nonstationary methods are a relatively recent development; their analysis is usually harder to understand, but they can be highly effective

The nonstationary methods we present are based on the idea of sequences of orthogonal vectors

- (An exception is the Chebyshev iteration method, which is based on orthogonal polynomials.)

Stationary Iterative Methods 1/3

Stationary iterative method: Iterative method that performs in each iteration the same operations on the current iteration vectors

Nonstationary iterative method: Iterative method that has iteration-dependent coefficients

- Dense matrix: Matrix for which the number of zero elements is too small to warrant specialized algorithms

Stationary Iterative Methods 1/3

Stationary iterative method: Iterative method that performs in each iteration the same operations on the current iteration vectors

Nonstationary iterative method: Iterative method that has iteration-dependent coefficients

- Dense matrix: Matrix for which the number of zero elements is too small to warrant specialized algorithms
- Sparse matrix: Matrix for which the number of zero elements is large enough that algorithms avoiding operations on zero elements pay off

Stationary Iterative Methods 1/3

Stationary iterative method: Iterative method that performs in each iteration the same operations on the current iteration vectors

Nonstationary iterative method: Iterative method that has iteration-dependent coefficients

- Dense matrix: Matrix for which the number of zero elements is too small to warrant specialized algorithms
- Sparse matrix: Matrix for which the number of zero elements is large enough that algorithms avoiding operations on zero elements pay off
- Matrices derived from partial differential equations typically have a number of nonzero elements that is proportional to the matrix size, while the total number of matrix elements is the square of the matrix size.

Stationary Iterative Methods 2/3

The rate at which an iterative method converges depends greatly on the spectrum of the coefficient matrix

- Hence, iterative methods usually involve a second matrix that transforms the coefficient matrix into one with a more favorable spectrum

Stationary Iterative Methods 2/3

The rate at which an iterative method converges depends greatly on the spectrum of the coefficient matrix

- Hence, iterative methods usually involve a second matrix that transforms the coefficient matrix into one with a more favorable spectrum
- The transformation matrix is called a preconditioner

Stationary Iterative Methods 2/3

The rate at which an iterative method converges depends greatly on the spectrum of the coefficient matrix

- Hence, iterative methods usually involve a second matrix that transforms the coefficient matrix into one with a more favorable spectrum
- The transformation matrix is called a preconditioner
- A good preconditioner improves the convergence of the iterative method, sufficiently to overcome the extra cost of constructing and applying the pre-conditioner

Stationary Iterative Methods 2/3

The rate at which an iterative method converges depends greatly on the spectrum of the coefficient matrix

- Hence, iterative methods usually involve a second matrix that transforms the coefficient matrix into one with a more favorable spectrum
- The transformation matrix is called a preconditioner
- A good preconditioner improves the convergence of the iterative method, sufficiently to overcome the extra cost of constructing and applying the pre-conditioner
- Indeed, without a preconditioner the iterative method may even fail to converge.

Stationary Iterative Methods 3/3

- The Jacobi Method

Stationary Iterative Methods 3/3

- The Jacobi Method
- The Gauss-Seidel Method

Stationary Iterative Methods 3/3

- The Jacobi Method
- The Gauss-Seidel Method
- The Successive Overrelaxation Method

Stationary Iterative Methods 3/3

- The Jacobi Method
- The Gauss-Seidel Method
- The Successive Overrelaxation Method
- The Symmetric Successive Overrelaxation Method

Data structures

If the coefficient matrix is sparse, large-scale linear systems of the form can be most efficiently solved if the zero elements of are not stored

- Sparse storage schemes allocate contiguous storage in memory for the nonzero elements of the matrix, and perhaps a limited number of zeros

There are many methods for storing the data

Data structures

If the coefficient matrix is sparse, large-scale linear systems of the form can be most efficiently solved if the zero elements of are not stored

- Sparse storage schemes allocate contiguous storage in memory for the nonzero elements of the matrix, and perhaps a limited number of zeros
- This, of course, requires a scheme for knowing where the elements fit into the full matrix.

There are many methods for storing the data

Data structures

If the coefficient matrix is sparse, large-scale linear systems of the form can be most efficiently solved if the zero elements of are not stored

- Sparse storage schemes allocate contiguous storage in memory for the nonzero elements of the matrix, and perhaps a limited number of zeros
- This, of course, requires a scheme for knowing where the elements fit into the full matrix.

There are many methods for storing the data

- Here we will discuss Compressed Row and Column Storage, Block Compressed Row Storage, Diagonal Storage, Jagged Diagonal Storage, and Skyline Storage.

Data structures

The efficiency of any of the iterative methods considered in previous sections is determined primarily by the performance of the matrix-vector product and the preconditioner solve, and therefore on the storage scheme used for the matrix and the preconditioner

- Since iterative methods are typically used on sparse matrices, we will review here a number of sparse storage formats

Data structures

The efficiency of any of the iterative methods considered in previous sections is determined primarily by the performance of the matrix-vector product and the preconditioner solve, and therefore on the storage scheme used for the matrix and the preconditioner

- Since iterative methods are typically used on sparse matrices, we will review here a number of sparse storage formats
- Often, the storage scheme used arises naturally from the specific application problem.

Storage scheme:

The way elements of a matrix are stored in the memory of a computer

- For dense matrices, this can be the decision to store rows or columns consecutively

Storage scheme:

The way elements of a matrix are stored in the memory of a computer

- For dense matrices, this can be the decision to store rows or columns consecutively
- For sparse matrices, common storage schemes avoid storing zero elements; as a result they involve integer data describing where the stored elements fit into the global matrix.

Data structures

- Compressed Row Storage (CRS)

Data structures

- Compressed Row Storage (CRS)
- Compressed Column Storage (CCS)

Data structures

- Compressed Row Storage (CRS)
- Compressed Column Storage (CCS)
- Block Compressed Row Storage (BCRS)

Data structures

- Compressed Row Storage (CRS)
- Compressed Column Storage (CCS)
- Block Compressed Row Storage (BCRS)
- Compressed Diagonal Storage (CDS)

Data structures

- Compressed Row Storage (CRS)
- Compressed Column Storage (CCS)
- Block Compressed Row Storage (BCRS)
- Compressed Diagonal Storage (CDS)
- Jagged Diagonal Storage (JDS)

Data structures

- Compressed Row Storage (CRS)
- Compressed Column Storage (CCS)
- Block Compressed Row Storage (BCRS)
- Compressed Diagonal Storage (CDS)
- Jagged Diagonal Storage (JDS)
- Skyline Storage (SKS)

Nonstationary Iterative Methods

- Nonstationary methods differ from stationary methods in that the computations involve information that changes at each iteration.

Nonstationary Iterative Methods

- Nonstationary methods differ from stationary methods in that the computations involve information that changes at each iteration.
- Typically, constants are computed by taking inner products of residuals or other vectors arising from the iterative method.

Section 3

Preconditioners

Preconditioners

The convergence rate of iterative methods depends on spectral properties of the coefficient matrix

- Hence one may attempt to transform the linear system into one that is equivalent in the sense that it has the same solution, but that has more favorable spectral properties

For instance, if a matrix approximates the coefficient matrix in some way, the transformed system has the same solution as the original system, but the spectral properties of its coefficient matrix may be more favorable.

Preconditioners

The convergence rate of iterative methods depends on spectral properties of the coefficient matrix

- Hence one may attempt to transform the linear system into one that is equivalent in the sense that it has the same solution, but that has more favorable spectral properties
- A preconditioner is a matrix that effects such a transformation.

For instance, if a matrix approximates the coefficient matrix in some way, the transformed system has the same solution as the original system, but the spectral properties of its coefficient matrix may be more favorable.

Section 4

GraphBLAS: A linear algebraic approach for high-performance graph algorithms

GraphBLAS: A linear algebraic approach for high-performance graph algorithms

The GraphBLAS

The GraphBLAS Forum is an open effort to define standard building blocks for graph algorithms in the language of linear algebra.

https://graphblas.org

A key insight behind this work is that when a graph is represented by a sparse incidence or adjacency matrix, sparse matrix-vector multiplication is a step of breadth first search

- By generalizing the pair of scalar operations involved in the linear algebra computations to define a semiring, we can extend the range of these primitives to support a wide range of parallel graph algorithms.

GraphBLAS lecture

GraphBLAS: A linear algebraic approach for high-performance graph algorithms