

M.Sc.CSIT
Second Semester
Syllabus

Compiler Optimization

Title: Compiler optimization

Course No: CSIT.521

Nature of the Course: Theory + Lab

Full Marks: 60 + 40

Pass Marks: 30 + 20

Credit Hrs: 3

Course Description:

This course is designed to provide students with a comprehensive understanding of compiler optimization. The course covers different dimensions of compiler optimization. The course includes basics of compiler optimization, dependence analysis, transformations, loop optimization, register allocation, control dependency, data flow analysis and optimization.

Course Objectives:

The course aims to achieve the following objectives:

- Introduce concepts of compiler optimization
- Provide concepts of parallelism
- Familiarize with dependence analysis and testing
- Provide concepts data flow analysis
- Familiarize with loop optimization
- Know about register allocations strategies
- Analyze control dependencies
- Perform interprocedural data flow analysis and appropriate optimization

Unit I: Introduction (5 Hrs.)

Compiler and its Phases, Significance of code optimization, Optimizing Compilers, Compiler challenges for high performance architectures: Pipelining, Vector Instructions, Superscalar and VLIW Processors, Processor Parallelism, Memory Hierarchy

Unit 2: Dependence Analysis (8 Hrs.)

Dependence and its Properties, Load-Store Classification, Dependence in Loops, Dependence and Transformations, Distance and Direction Vectors, Loop-carried and Loop-independent Dependences, Loop-Carried Dependence, Loop-Independent Dependences, Iteration Reordering

Dependence Testing, Single-Subscript Dependence Tests, ZIV and SIV Tests, Multiple Induction Variable Tests, Delta Test

Unit 3: Preliminary Transformations (6 Hrs.)

Loop normalization, Data Flow Analysis: Definition-Use Chains, Dead Code Elimination, Constant Propagation, Static Single-Assignment Form, Induction-Variable Exposure: Forward Expression Substitution, Induction-Variable Substitution, Driving the Substitution Process

[Handwritten signatures and initials at the bottom of the page]

Unit 4: Parallelism (8 Hrs.)

Loop Interchange, Loop Interchange and Vectorization, Scalar Expansion, Scalar and Array Expansion, Node Splitting, Recognition of Reductions, Index-set Splitting, Run-time Symbolic Resolution, Loop skewing, Single-Loop Methods, Perfect Loop Nests, Imperfectly Nested Loops

Unit 5: Control Dependence (7 Hrs.)

Branch classification, Forward, Exit and Backward branches, Iterative dependences, Control dependence: Control dependence in loops, Application of control dependence to parallelization, Program dependence graph

Unit 6: Register Allocation (2 Hrs.)

Introduction to register allocation, Graph coloring-based register allocation, Linear scan register allocation

Unit 7: Interprocedural Data Flow Analysis and Optimization (9 Hrs.)

Interprocedural Analysis, Need for Interprocedural Analysis, Flow-insensitive and flow-sensitive problems, Constant propagation and alias analysis, Kill analysis, Array section analysis, Interprocedural Optimization, Inline substitution, Linkage tailoring and Procedure cloning, Management of interprocedural analysis and optimization.

Laboratory Works:

Laboratory works include implementing and simulating the concepts in above mentioned chapters using appropriate platforms and tools.

Text/Reference Books

1. Optimizing Compilers for Modern Architectures, Randy Allen and Ken Kennedy, Morgan-Kaufmann
2. Advanced Compiler Design Implementation, Steven S. Muchnick, Morgan-Kaufmann
3. Modern Compiler Design, Dick Grune, Kees van Reeuwijk, Henri E. Bal, Criel J.H. Jacobs, Koen Langendoen
4. Compilers Principles, Techniques, and Tools, Alfred V. Aho, Pearson

Handwritten signatures and initials:
gizt, Azkwe, JH, G, h, ju, RZ

Soft Computing

Title: Soft Computing

Course No: CSIT.522

Nature of the Course: Theory + Lab

Full Marks: 60 + 40

Pass Marks: 30 + 20

Credit Hrs: 3

Course Description:

This course is designed to provide students with a comprehensive understanding of concepts of soft computing. The course includes the concepts of soft computing, fuzzy logic and fuzzy systems, artificial neural networks, evolutionary computing, and swarm intelligence.

Course Objectives:

The course aims to achieve the following objectives:

- Introduce concepts of soft and intelligent computing
- Provide concepts of fuzzy logic
- Familiarize with fuzzification and defuzzification
- Understand developing fuzzy systems
- Familiarize with artificial neural networks
- Provide concepts evolutionary computing and build models based on the computing
- Acquaint with the concepts of swarm intelligence

Unit I: Introduction (5 Hrs.)

Soft Computing, Computational Intelligence, Paradigms of Computational Intelligence, Approaches to Computational Intelligence, Synergies of Computational Intelligence Techniques, Applications of Computational Intelligence

Unit II: Fuzzy Logic (12 Hrs.)

Fuzzy Logic, Fuzzy Sets, Membership Functions, Features of Membership Functions, Operations on Fuzzy Sets, Linguistic Variables, Linguistic Hedges, Fuzzy Relations, Operations of Fuzzy Relations, Fuzzy If-Then Rules, Approximate Reasoning, Extension Principle, Fuzzification, Defuzzification, Defuzzification Functions, Inference Mechanisms, Mamdani Inference Model, Sugeno Inference Model, TSK Inference Model, Fuzzy System, Fuzzy Modeling, Fuzzy Control, Design of Fuzzy Controller, Fuzzy Clustering, Fuzzy Classification

Unit III: Artificial Neural Networks (12 Hrs.)

Artificial Neuron Model, Activation Functions, Network Architecture, Learning in Neural Networks, Gate Realization, Perceptron Learning, Back-propagation Algorithm, Radial Basis Function Neural Networks, Hopfield Networks, Deep Neural Networks, Convolutional Neural

Signature

Signature

Signature

Signature

Signature

Networks, Generative Adversarial Networks, Recurrent Neural Networks, Transformers, Fuzzy Neuro Systems, Adaptive Neuro Fuzzy Information Systems

Unit IV: Evolutionary Computing (8 Hrs.)

Evolutionary Computing, Terminologies of Evolutionary Computing, Genetic Operators, Evolutionary Algorithms, Evolutionary Programming, Evolution Strategies, Genetic Algorithm, Genetic Programming, Differential Evolution, Cultural Algorithm, Performance Measures of Evolutionary Algorithms, Evolutionary Systems, Multi-objective Optimization, Coevolution, Parallel Evolutionary Algorithm, Fuzzy Adaptive Evolutionary Algorithms

Unit V: Swarm Intelligence (8 Hrs.)

Particle Swarm Optimization, Social Network Structure: The Neighborhood Principle, Particle Swarm Optimization Algorithm, Particle Swarm Optimization System Parameters, Applications of Particle Swarm Optimization, Ant Colony System, Ant Colony Optimization, Applications of Ant Colony Optimization, Cuckoo Search Algorithm, Artificial Bee Colony Optimization

Laboratory Works:

Laboratory works include implementing and simulating the concepts in above mentioned chapters using appropriate platforms and tools.

Text books/ References:

1. Computational Intelligence Synergies of Fuzzy Logic, Neural Networks And Evolutionary Computing, Nazmul Siddique and Hojjat Adeli, Wiley Publication
2. Soft Computing With MATLAB Programming, NP Padhy and SP Simon, Oxford University Press
3. Computational Intelligence An Introduction, Andries P. Engelbrecht, John Wiley & Sons Ltd
4. Fundamentals Of Computational Intelligence, Neural Networks, Fuzzy Systems, and Evolutionary Computation, James M. Keller, Derong Liu, and David B. Fogel, IEEE Press, Wiley
5. Principles Of Soft Computing, S. N. Sivanandan, and S.N. Deepa, Wiley
6. Fuzzy Logic With Engineering Applications, Timothy J Ross, Mcgraw Hill
7. Computational Intelligence: Principles, Techniques and Applications, A. Konar, Springer Verlag

[Handwritten signatures and marks at the bottom of the page]

Machine Learning

Course Title: Machine Learning
Course No: CSIT.523
Nature of the Course: Theory+Lab
Semester: II

Full Marks: 60+40
Pass Marks: 30+20
Credit Hrs: 3

Course Description:

The notion of machine learning and its use in practical applications are covered in this course. Algorithms for supervised, unsupervised, and reinforcement learning are all included. To select the optimal algorithm for a specific task, evaluation measures are also explored.

Course Objectives:

- To discuss and demonstrate various machine learning algorithms.
- To familiarize with the applications of machine learning algorithms.
- To enable students to analyze machine algorithms.

Unit 1: Introduction (4 Hrs.)

- 1.1. Introduction to Machine Learning, Machine Learning vs. AI, Categories of Machine Learning algorithms.
- 1.2. Machine Learning Tasks and Applications, Machine Learning Theory.

Unit 2: Supervised Learning (14 Hrs.)

- 2.1. Concept of Gradient Descent, Batch, Stochastic and Mini-batch Gradient Descent and their advantages and drawbacks.
- 2.2. Concept of Regression, Derivation and Demonstration of Linear Regression, Locally Weighted Regression and Logistic Regression.
- 2.3. Classification, Classification vs. Regression, Naïve Bayes Classifiers, KNN Classifiers, Decision Trees, Support Vector Machines.
- 2.4. Ensemble Learning, Random Forest, ensemble Size, Bagging, Boosting, Stacking Cross-Validation, Underfitting and Overfitting.

Unit 3: Unsupervised Learning (12 Hrs.)

- 3.1. Clustering, Partitioning Method: K-Means, Agglomerative and Divisive Clustering, DBSCAN, Mixture Models and EM Algorithm.
- 3.2. Dimensionality Reduction: Principal Component Analysis, Singular Value Decomposition, Factor Analysis, Independent Component Analysis.
- 3.3. Outlier Detection: Outlier Detection Methods, Clustering based approaches, Classification based Approach.

Unit 4: Model Evaluation and Selection (8 Hrs.)

- 4.1. Classification Evaluation: Confusion Matrices. Evaluation Metrics for Binary Classification and Multi-class Classification (Accuracy, Recall, Precision, F1-score, and Specificity), ROC curves, Log-loss.

[Handwritten signatures and initials at the bottom of the page]

4.2. Regression Evaluation: Mean Squared Error (MSE), Root Mean Squared Error (RMSE), Root Mean Squared Log Error (RMSLE), Mean Absolute Error (MAE), Mean Absolute Percentage Error (MAPE). R^2 , Adjusted R^2 .

4.3. Evaluation metrics for Clustering: Dunn Index, Silhouette Coefficient, Elbow Method.

Unit 5: Reinforcement Learning (8 Hrs.)

5.1. Reinforcement Learning, Terminologies related to RL, Markov Decision Process (MDP), Value & Policy Functions, Bellman Expectation Equation.

5.2. Optimal value and policy functions, Bellman Optimality Equation, Value Iteration algorithm, Policy Iteration algorithm.

Laboratory Works:

Student should implement and assess performance of all algorithm's discussed under classification, regression, clustering, and dimensionality reduction.

Recommended Book:

1. Christopher M. Bishop: Pattern Recognition and Machine Learning (Information Science and Statistics), Springer.
2. Shai Shalev-Shwartz and Shai Ben-David: Understanding Machine Learning: From Theory to Algorithms, Cambridge University Press.
3. Marc Peter Deisenroth, A. Aldo Faisal and Cheng Soon Ong: Mathematics for Machine Learning, Cambridge University Press.
4. Mehryar Mohri, Afshin Rostamizadeh and Ameet Talwalkar: Foundations of Machine Learning (Adaptive Computation and Machine Learning series), The MIT Press.

The bottom of the page contains several handwritten signatures and initials. From left to right, there is a signature that appears to be 'Jasp', a large stylized signature, a signature that looks like 'ABand' with a diagonal line through it, a signature that looks like 'Sah.', a signature that looks like 'Gur', and a signature that looks like 'Ravi'.

Principles of Programming Language

Title: Principles of Programming Languages

Course No: CSIT.524

Nature of the Course: Theory + Lab

Full Marks: 60 + 40

Pass Marks: 30 + 20

Credit Hrs: 3

Course Description:

This course is designed to provide students with a comprehensive understanding of concepts of programming principles. The course covers different aspects including syntax, semantics, bindings, scopes, data types, expressions, assignment and control statements, subprograms, abstract data types, object oriented concepts, concurrency, exception and event handling.

Course Objectives:

The course aims to achieve the following objectives:

- Introduce different concepts principles of programming
- Provide concepts of syntax and semantics
- Familiarize with binding and scopes
- Provide concepts of various data types
- Familiarize with expressions and statements
- Present concepts of subprograms and abstract data types
- Acquaint with exception and event handling in language constructs

Unit I: Introduction (4 Hrs.)

Programming Language, Programming Domain, Language Evaluation Criteria, Language Design, Language Categories, Language Design Trade-offs, Implementation Methods, Programming Environment, Evolution of Programming Languages

Unit II: Syntax and Semantics (4 Hrs.)

Describing Syntax, Attribute Grammars, Describing Semantics, Lexical Analysis, Syntax Analysis, Parsing

Unit III: Binding and Scopes (3 Hrs.)

Names, Variables, Binding, Scope, Scope and Lifetime, Referencing Environments, Named Constants

Unit IV: Data Types (5 Hrs.)

Primitive Data Types, Character String Types, Enumeration Types, Array Types, Associative Arrays, Record Types, Tuple Types, List Types, Union Types, Pointer and Reference Types, Optional Types, Type Checking, Strong Types, Type Equivalence

Unit V: Expression, Assignment Statements and Control Structures (6 Hrs.)

[Handwritten signatures and marks at the bottom of the page]

Expression, Arithmetic Expression, Overloaded Operators, Type Conversions, Relational and Boolean Expression, Short-Circuit Evaluation, Assignment Statements, Mixed Mode Assignment,

Control Statements, Selection Statements, Iterative Statements, Unconditional Branching, Guarded Commands

Unit VI: Subprograms (7 Hrs.)

Fundamentals of Subprograms, Design Issues, Local Referencing Environments, Parameter Passing, Calling Subprograms, Overloaded Subprograms, Generic Subprograms, User-Defined Overloaded Operators, Closures, Coroutines, Implementing Subprograms, Semantics of Calls and Returns, Implementing Simple Subprograms and Subprograms with Stack-dynamic Local Variable, Nested Subprograms, Blocks, Implementing Dynamic Scoping

Unit VII: Abstract Data Types and Encapsulation Constructs (5 Hrs.)

The Concept of Abstraction, Introduction to Data Abstraction, Design Issues for Abstract Data Types, Parameterized Abstract Data Types, Encapsulation Constructs, Naming Encapsulations, Inheritance, Polymorphism

Unit VIII: Concurrency (6 Hrs.)

Concurrency, Introduction to Subprogram-Level Concurrency, Semaphores, Monitors, Message Passing, Ada Support for Concurrency, Java Threads, C# Threads, Concurrency in Functional Languages, Statement-Level Concurrency

Unit IX: Exception Handling and Event Handling (5 Hrs.)

Introduction to Exception Handling, Exception Handling in C++, Exception Handling in Java, Exception Handling in C#, Exception Handling in Python, Introduction to Event Handling, Event Handling with Java, Event Handling with C#, Event Handling with Python

Laboratory Works:

Laboratory works include implementing and simulating the concepts in above mentioned chapters using appropriate platforms and tools.

Text/Reference Books

1. Concepts of Programming Languages, 12th edition; Robert W. Sebesta, Pearson
2. Programming Languages: Design and Implementation. T.W. Pratt and M.V. Zelkowitz, Prentice Hall.
3. Programming Language Design Concepts, D. A. Watt, Wiley Dreamtech
4. Programming Languages, 2nd Edition, A.B. Tucker, R.E. Noonan, TMH.
5. Programming Languages: Concepts and Constructs, Ravi Sethi, Pearson Education.

[Handwritten signatures and initials at the bottom of the page]

Systems Programming

Course Title: Systems Programming

Course Code: CSIT.525

Nature of the course: Theory+Lab

Semester: II

Full Marks: 60+40

Pass Marks: 30+20

Credit Hrs.: 3

Course Description:

The design and implementation of machine dependent and independent assembler, loader, linker, microprocessor, and some compiler elements will be covered in this course. The course will also include a project that involves the implementation of an assembler, a linker, a loader, and a compiler.

Course Objectives

- To discuss hypothetical machine architectures SIC and SIC/XE
- To present the basic structure and design of a micro-assembler, a linker, a loader, and a compiler.
- To complete a project independently which will involve the design and implementation of micro-assembler, linker, and loader.

Course Contents

Unit 1: SIC and SIC/XE Architecture (4 Hrs.)

- 1.1. System software and machine architecture
- 1.2. SIC and SIC/XE architectures, SIC Programs.
- 1.3. RISC and CISC machine architectures.

Unit 2: Assembler Design (20 Hrs.)

- 2.1. Basic assembler functions: Simple SIC Assembler, Algorithms and Data structures for Assembler
- 2.2. Machine dependent assembler features: Instruction formats and addressing modes, program relocation
- 2.3. Machine independent assembler features: Literals, Symbol defining statements, program blocks, control sections, program linking
- 2.4. Assemblers Design Options: One pass assembler, multi-pass assembler

Unit 3: Loader and Linker Design (8 Hrs.)

- 3.1. Basic loader functions: Design of absolute loader, Simple Bootstrap loader
- 3.2. Machine dependent features: Relocation, Program Linking

[Handwritten signature]

[Handwritten signature]

[Handwritten signature]

[Handwritten signature]

[Handwritten signature]

[Handwritten signature]

[Handwritten signature]

- 3.3. Machine independent Loader Features: Automatic library Search, loader options
3.4. Design Options: Linking Loader, Linkage Editor, Dynamic Linking

Unit 4: Macro processor Design (8 Hrs.)

- 4.1. Basic Macro processor functions: Macro Definition and Macro Expansion, Data structures for Macro processor
4.2. Machine Independent features: Concatenation of Macro Parameters, Generation of Unique Labels, Conditional Macro Expansion
4.3. Macro processor Design Options: Recursive Macro Expansion, General Purpose Macro Processors

Unit 5: Fundamentals of Compiler Design (5 Hrs.)

- 5.1. Basic Compiler Functions: Grammar, Lexical Analysis, Syntactic Analysis (operator precedence parsing, Recursive Descent Parsing), intermediate code generation.

Laboratory Works:

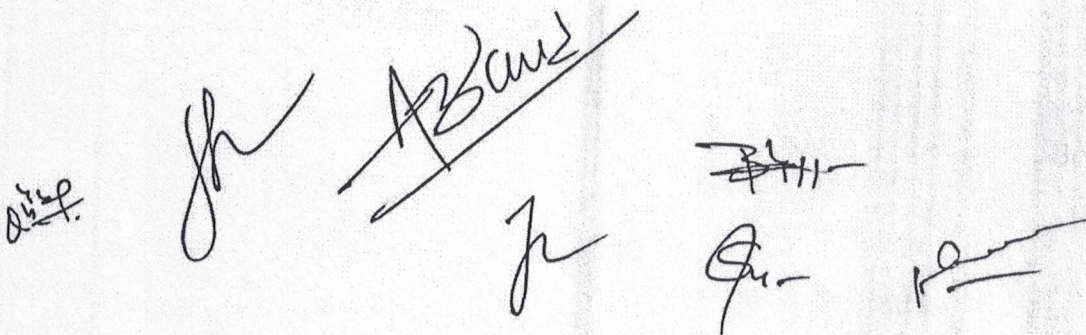
Student should design and implement Assembler and Linking Loader for simple hypothetical computer architecture

Textbook

1. System Software: An Introduction to Systems Programming by Leyland L. Beck: Addison Wesley Publishing Company.

Reference Books:

1. Daniel H. Marcellus, Systems programming for Small Computers, prentice-Hall, New 44 Jersey
2. Roy S. Ellzey, Computer System Software, The Programmer/Machine Interface, Inc.,
3. Robert M. Graham, Principles of Systems Programming, John Wiley & Sons
4. Thomas G. Windeknecht, 6502 Systems programming, Little Brown & Company

The bottom of the page contains several handwritten signatures and initials. From left to right, there is a small signature, a large stylized 'Jh', a signature that appears to be 'ABand' with a long horizontal line through it, a stylized 'Jh', a signature that looks like 'ZHI', and a signature that looks like 'KQ'.

Seminar II

Course Title: Seminar II
Course No: CSIT 526
Nature of the Course: Seminar
Semester: II

Full Marks: 25
Pass Marks: 12.5
Credit Hrs.: 1

Course Description:

Students will gain experience speaking in front of a scientific audience in this one credit course as well as deepen their understanding of the topic. Students will organize presentations for faculty members and other students after conducting topic research. The topic may include any area of computer science and information technology, but the topic and seminar document must first receive the advisor's approval.

Course Objectives:

- To improve presentation and communication skills.
- To give pupils the confidence to speak to an audience.
- To promote self-learning behaviors.
- To improve academic writing abilities.

Seminar Report Format:

Cover Page

Letter of Approval

Abstract

Table of Contents

Chapter 1: Introduction

- 1.1. Introduction
- 1.2. Problem Statement
- 1.3. Objectives

Chapter 2: Background and Literature Review

- 2.1. Background
- 2.2. Related Works

Chapter 3: Methodology

- 3.1. Methodology
- 3.2. Tools Used
- 3.3. Experimental Environment
- 3.4. Performance Measures

Handwritten signatures and initials:
Aizel
h
Arzame
Pr.
h
Brie
Ker

Chapter 4: Results and Discussion

4.1. Experimental Results

4.2. Result Analysis and Interpretation

Chapter 5: Conclusion and Recommendation

5.1. Conclusion

5.2. Recommendations

References

Referencing and citation should be done in IEEE format.

[Handwritten signatures and marks]