DESIGN AND DEVELOPMENT OF A REUSABLE SOFTWARE COMPONENTS REPOSITORY

Ph.D. THESIS

By
RAJESH KUMAR
(9020356)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
THAPAR UNIVERSITY, PATIALA
PATIALA – 147 004 (INDIA)
DECEMBER 2007

THAPAR UNIVERSITY, PATIALA

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled "DESIGN AND DEVELOPMENT OF A REUSABLE SOFTWARE COMPONENT REPOSITORY" in partial fulfillment of the requirements for the award of the Degree of Doctor of Philosophy and submitted in the Department of Computer Science and Engineering of Thapar University, Patiala is an authentic record of my own work carried out during a period from January 2003 to November 2007 under the supervision of **Dr. R. C. Joshi**, Professor, Department of Electronics and Computer Engineering, Indian Institute of Technology, Roorkee and **Dr. Mayank Dave**, Assistant Professor, Department of Computer Engineering, National Institute of Technology, Kurukshetra.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other Institute/University.

(RAJESH KUMAR)
Regn No. 9020356

This to certify that the above statement made by the candidate is correct to the best of our knowledge.

Date: 3/10/08

Dave 4/10/2008
(Dr. Mayank Dave)
Assistant Professor
NIT, Kurekshetra

3/10/08
(Dr. R. C. Joshi)
Professor
IIT, Roorkee

Abstract

Software reuse has great potential to improve the productivity and quality. Software reuse helps to produce quick prototypes with lower cost. We can envision the scenario of the near future where a software manager by using his mobile device can check the available software assets in the organization repository and accordingly plan the schedule and cost of the software product. However various constraints such as non-availability of rich quality reuse components repository, efficient classification and retrieval techniques add a new dimension to the technical challenges before this vision becomes reality. Thus component rich repositories and efficient classification and retrieval techniques are needed to make software reuse a routine practice. The work presented in this thesis is an effort to address these issues by proposing new and efficient classification and retrieval schemes for a reusable software repository.

First part of the thesis proposes a flexible weighted approach that makes use of both formal as well as natural specifications to classify and retrieve the components from the repository. Weights are assigned to each type of specifications depending upon the software development phase in which user is working. In early phases of development, requirements are descriptive in nature, so user may assign higher weight to natural specifications. In later development phases, when requirements are in concrete form and formal view is available, more weight can be assigned to formal specifications for matching. Depending upon user requirements weights can be adjusted and appropriate component can be retrieved for given user context. We have been able to locate the component with even 100% match by assigning weight of 80% to formal and 20% to natural specifications. This has been observed if we further increase the percentage weight assigned to formal specifications then it will be able to locate only exact matches but it may miss all near matches.

In second part of the thesis a genetic algorithm based component repository is proposed. In this repository components are described using keyword based description and genetic description to store and retrieve the components. Thirty-two reusability attributes of the components are identified and are used to describe components in the

repository for classification and retrieval. This technique has been demonstrated by considering three component queries: simple, moderate and complex. It took 28 iterations to converge moderate query and 42 iterations to converge complex type of query. It is observed in genetic algorithm based step for complex query only 17% of the components are visited to match the appropriate component with precision of 0.178 and without missing any relevant component.

Third part of the thesis introduces the concept of ant colony algorithm in component clustering and retrieval. On the basis of component interactions various terms are identified to represent a particular cluster of components. Clusters are sequenced for search in increasing order of average attribute factor. 1,50,000 component descriptions from 22 domains are considered to verify the presented technique. Proposed technique generates rules that help in components classification and retrieval. If any term participates in rule generation then pheromone is deposited and it will be visited in next iteration otherwise pheromone will get evaporated. In best case only 10% of the components are visited to locate a component with precision and recall of 1.

Lastly, the contribution made in the dissertation is summarized and scope for the future work is outlined.

Acknowledgements

I would like to express my deepest gratitude to my supervisors Dr. R. C. Joshi and Dr. Mayank Dave for their encouragement, painstaking supervision, innovative suggestions and invaluable help during the entire period of my Ph.D. studies. It has been a great honor and pleasure for me to work under their supervision. I learned a great deal from them, not only about research but also about matters touching many other aspects which will benefit me in future life and career.

My sincere thanks are due to Dr. Abhijit Mukherjee, Director, Thapar University, Patiala and Dr. Seema Bawa, Head, Department of Computer Science and Engineering, Thapar University for providing me the necessary administrative support for the completion of the work.

I am extremely grateful to the celebrated authors whose precious work have been consulted and referred in my research work. I also wish to convey my appreciation to fellow research scholars and colleagues who provided encouragement and timely support in the hour of need.

Finally, I could not reach the important milestone of my life without the support and encouragement of my family. Thanks to my parents who have made it possible for me to reach where I am today. I express my sincere appreciation and gratitude to my wife Reena and children Kashish and Kartikay for their patience and encouragement when it was most required.

And above all, I am thankful to Almighty whose divine grace gave me the required courage, strength and perseverance to overcome various obstacles that stood in my way.

Rajesh Kumar

Contents

List of Figures

List of Tables

Chapter 1

Introduction and Statement of the Problem

1.1 Introduction

As consumer products are becoming more and more software intensive, the demand of producing and maintaining software products is increasing rapidly. Software productivity has been steadily rising for past several years. However, even with the steady rise in the number of computer professionals, it has not kept up with the rising demand for developing new ever more complex software systems and for maintaining existing software systems [11, 13, 94]. According to Boehm, *"the only factor that can yield that kind of productivity leverage is the number of software instructions that have to be developed to deliver a given functionality, instead of searching ways of writing code faster, we have to look for ways of writing less of it"* [7, 154]. Automatic programming, whereby a computer system is capable of producing executable code based on informal and incomplete user requirements is very difficult, if ever possible [12]. A considerable number of software projects miss schedules, exceed budgets, deliver poor quality software and sometimes deliver wrong functionality. Software organizations and researchers have been looking for effective ways to deliver quality software products on schedule and on cost every time. The proposed answer to this problem is software reuse. Software productivity can be increased with software product reuse because reused software components need not to be developed from scratch. Software reuse can save time to market that results in larger market share. With software reuse prototypes can be developed very quickly and at very low cost as instead of developing specifications, designs and code from scratch, existing components can be reused [20, 33, 68, 74, 75, 86, 115].

Software reuse is a promising alternative to enhance software productivity [39, 41, 72, 130]. It can improve quality and reliability of the software. It may help to identify design errors at early stage by plugging different candidate components. Still software reuse is not in practice to its potential due to number of reasons. Foremost reason is that it requires a rich repository of quality software components. Then it is very difficult to

retrieve reusable software components effectively from the repository [48, 63, 134, 137, 139, 160]. Software organizations face problem in practicing software reuse because of following reasons [29, 57, 71, 83, 84, 86, 91]:

- Object oriented analysis, CASE tools, formal methods, agile methods etc. are proposed but no clear-cut methodology for software reuse is defined [15, 25, 26, 27, 74, 78, 87].
- There is an absence of models and theories that can help to comment or estimate about the software without developing it.
- Rapid evolution of technologies does not allow proper assessment.
- Collecting context independent knowledge or representing context is very difficult [156].
- Software development for reuse especially for large complex systems is very cumbersome [25, 30].
- There is lack of empirical studies and technologies that can help to estimate what a component can perform better and what it can not, without using it [124].

The current software component retrieval methods include the free text based methods, the pre-enumerated vocabulary methods, signature-matching methods, behavior based methods and facet classification methods [108, 149]. Most of these methods use natural language specification based matching. Retrieval methods based on natural language based specifications suffer from inherited ambiguity. Other existing methods based on formal specifications are very complex and are difficult to adapt to user environment. In this context, we agree with Bayer et al. when they said , *"Existing methods have been either not flexible enough to meet the needs of various industrial situations, or they have been too vague, not applicable without strong additional interpretations and support. A flexible method that can be customized to support various enterprise situations with enough guidance and support is needed"*. Under this motivation, this research proposes a an effective approach to specify reusable components in software repository and then to classify and retrieve these components for possible reuse. Our approach uses both natural as well as formal specifications for specifying components and for retrieving components

from repository. Applying a combination of both type of specifications have following benefits [3, 109, 116, 117, 133, 134]:

- Using only natural specification based retrieval may suffer from ambiguity problem and using only formal specifications may miss all near matches.
- First search space can be reduced with natural specifications, and then on candidate components formal matching can be applied for better results.
- For automation formal specifications remove subjectivity.
- Formal specifications remove ambiguity, incompleteness and inconsistencies due to natural language specifications.
- Combining both type of specifications help user to understand specifications properly, otherwise formal specifications are difficult to understand.
- In Genetic Algorithm based approach with keywords first initial population is improved then in further iterations combination of specification are used.

In summary, software reuse is especially important in improving quality and reliability of software [136]. The main problem in software reuse is effective design and development of software repository and also absence of effective retrieval and classification techniques for reusable components. Most of the existing repositories are either domain specific or are very difficult to use. Also most of the retrieval and classification techniques suffer from ambiguity problem. What derives from the above discussion is that further research needs to be done to come up with more effective reusable software component repositories and more effective retrieval and classification techniques.

1.3 Statement of the Problem

Software reuse has two major categories of activities; Development for reuse and development with reuse. First part of the research proposes various factors that affect reusability of component; these factors are used to describe components in the repository. This part will fall in category of development for reuse activities. Following 32 reusability factors have been identified for describing components in repository for effective retrieval:

- Component Name
- Alternate names
- Language used
- Development Platform
- Arguments
- Size
- Cyclomatic Complexity
- Domain
- Designed for reuse
- Specifications
- Test Cases
- Test Plans
- Certification
- Developed by
- Price
- Component history
- Performance
- Pre-Condition
- Post-Condition
- Algorithmic Complexity
- Effort
- Data Structures
- Resource Utilization
- Bugs (Known)
- Limitations
- Flexibility and Interoperability
- Quality
- Criticality
- Adaptability
- Distribution / Availability
- Keywords
- Remarks

In this research work, focus has been put on developing reusable component repository in which components are described using above highlighted factors to address the main problem of component classification and retrieval. Second part of the research proposes a classification and retrieval technique for reusable components that will help in development with reuse.

The main objective of the present research work can be stated as " to design and develop a reusable software component repository with effective component

classification and retrieval". In order to handle the above problem, it can be subdivided into smaller objectives as follows:

1. To explore various factors affecting software component retrieval.
2. To draw the relationship between formal specifications and software reusability.
3. To explore various formal specification models and to select most appropriate one which is suitable for software repositories.
4. To explore various techniques/technologies for constructing reusable component repository.
5. Designing an effective component classification and retrieval technique for repository.
6. Constructing a component repository of reusable components.
7. To create a friendlier user interface for component retrieval.

1.3 Organization of the Thesis

The rest of the thesis is organized as follows. Chapter 2 gives the background and reviews the related work. We first provide some background on software reusability, software reuse products and reuse activities. Next various existing classification and retrieval techniques have been reviewed. Further in this chapter existing key software reuse repositories have been reviewed and compared.

In chapter 3, various formal specification languages and methods have been discussed and compared. We suggest a component description technique that makes use of both types of specifications, natural as well as formal. By using combination of specifications a flexible weighted component retrieval technique has been proposed. The performance of the proposed technique is studied extensively by considering various example cases.

Chapter 4 proposes a Genetic Algorithm based software repository that makes use of natural description and genetic descriptions to describe components. A genetic algorithm based component classification and retrieval technique has been implemented. The performance of proposed technique has been demonstrated by considering three, simple, moderate and complex types of queries. Retrieval effectiveness has been measured by considering precision, recall and coverage ratio.

In chapter 5, an Ant Colony algorithm based component clustering technique is developed. The component clustering is done with the help of component interactions. Further an ant colony optimization based rule generation approach for component classification and retrieval has been developed. The performance of the proposed technique is shown by considering empirical results. The retrieval effectiveness has been measured with precision, recall and coverage ratio of various cases.

Finally, Chapter 6 summarizes the contributions of this research work. In addition, we describe a number of directions for future research on the work presented here.

Chapter 2

Background and Related Work

2.1 Introduction

Software Engineering is the application of a systematic, disciplined, quantified approach to the development, operation and maintenance of software. It involves analysis, design, construction, verification and management of the entities needed to develop the quality software [35, 36]. It is exhaustively time consuming process. Wherever developers try to compromise with the process it results in software crisis [94]. Researchers are looking for various ways to cope up with software crisis. Researchers have been looking for ways and means to develop quality software economically and on time. Software Reuse can be one of the alternatives to develop reliable software on time and on budget. Software Reuse is developing new software with already existing software assets. Like any other engineering discipline, software engineering surely gets benefited greatly from software reuse. But unlike most other engineering disciplines software reuse does not arise naturally in software engineering, and when it does, it comes with costs and risks. Paul G. Basset in 1997 said that reuse is neither a silver bullet nor a magic weight loss pill. It is a diet and exercise program. Various benefits and limitations of software reuse are discussed. Software reuse benefits are classified into three categories [96, 97, 99]:

- Gains in Productivity: Gains in productivity are the main motivation for software reuse. By reusing existing assets we save manpower effort (person-months) to develop them again.
- Gains in Quality: Gains in quality are from two sets of measures: First, when a component is developed for reuse, one can rationalize large investments in its quality, on the premise that these investments will be amortized over its multiple uses; Second, when a reusable component is used by a larger community it gets debugged more thoroughly.

- Gains in Development Schedule: By using reusable assets, we not only save effort in person-months but also schedule in months. This translates into shorter time to market, which may in turn mean bigger market share.

Even with all these benefits reuse has not been a matter of routine practice because of following limitations [93, 96, 161]:

- Limited Reuse Potential: software assets are very information-rich; this limits the likelihood of a match between a specific query and an available asset. Modern programming disciplines emphasize on information hiding; this helps us to control this problem but do not overcome entirely.
- Non-negligible Obstacles: The introduction of reuse requires profound changes in the operational procedures of an organization; these changes are usually met with resistance. Also reuse requires healthy investments and does not bring returns immediately hence it requires long-term managerial support.
- Risks: It is possible for a poorly planned reuse initiative to actually cost more than it saves.
- General Software Development Methodologies do not support software reuse.

These limitations of software reuse would pose many challenging problems for software reuse practitioners and researchers that do not exist otherwise in traditional software engineering. So software reuse requires careful planning, realistic expectations and a long-term perspective. Reusable assets can be executable code, source code, requirements documents, designs, test data, documentation etc. We need large software component repositories to support software reuse. There is a need to come up with new effective mechanisms to manage software repositories.

2.2 Software Reuse Products

The most common reusable artifact is the source code. But it is not the only one reuse candidate; actually other software engineering work products can also be reused. In this section we review some of the common reusable artifacts [96].

- *Executable Code*: It is usually represented in machine-readable form, and is indexed by means of its functionality.
- *Source Code*: It is the main candidate for reuse. Source code has both structural as well as functional details. It can be viewed as problem solving knowledge. It can be indexed either by its functional properties or by structural properties.
- *Requirements Specifications*: Code artifacts are executable but requirements are not. Requirements are recorded in form of natural language or in form of formal specifications or in mixed format. They represent basic functionalities of any software product. Specifications can be indexed by means of functionalities. Requirements specifications may be reused to construct either compound specifications or for variations/up gradations on the original product.
- *Designs*: Designs are generic representations of the problem solving knowledge. As compared to specification assets, designs capture structural information rather than functional information. Designs are represented by patterns. They can be indexed by features of the family of problems they intend to solve.
- *Test Data*: For testing similar kind of products or to test the product after some maintenance operations, test data can be reused. Representation of test data is straightforward and depends upon the type of format user is following. Test data can be indexed either by description of input domain or by general indication of the functionality performed by the system.
- *Documentation*: Many reusable assets such as design, specifications etc are represented by natural language documentation. Documentation is most likely represented in natural language and can be indexed by the asset that it documents or it can be indexed using hypertext.
- *Architectures*: Software architecture defines the structure of a software system. It is an aggregate set of components that exchange data. The architectures have a higher level of abstraction than programming language codes and are of different nature. They represent information flow, control

flow or communication protocols between components. Architectures are represented by means of specialized notations and are indexed by means of their architectural features.

2.3 Reuse Activities

Software reuse activities are basically divided into two categories [96]:

- Development FOR Reuse
  - Identifying what can be widely reused
  - Defining components and understanding them for reuse
  - Classifying and storing the reusable assets in a library
  - Representing components in standard form
- Development WITH Reuse
  - Locating reusable components
  - Understanding the reusable component
  - Adapting the reusable component for possible reuse
  - Combine and integrate the components

The domain engineering team is responsible for producing, maintaining and cataloging reusable assets, to make them available to application engineering. The focus of application engineering team is to develop applications using reusable assets. These activities are shown in figure 2.1.

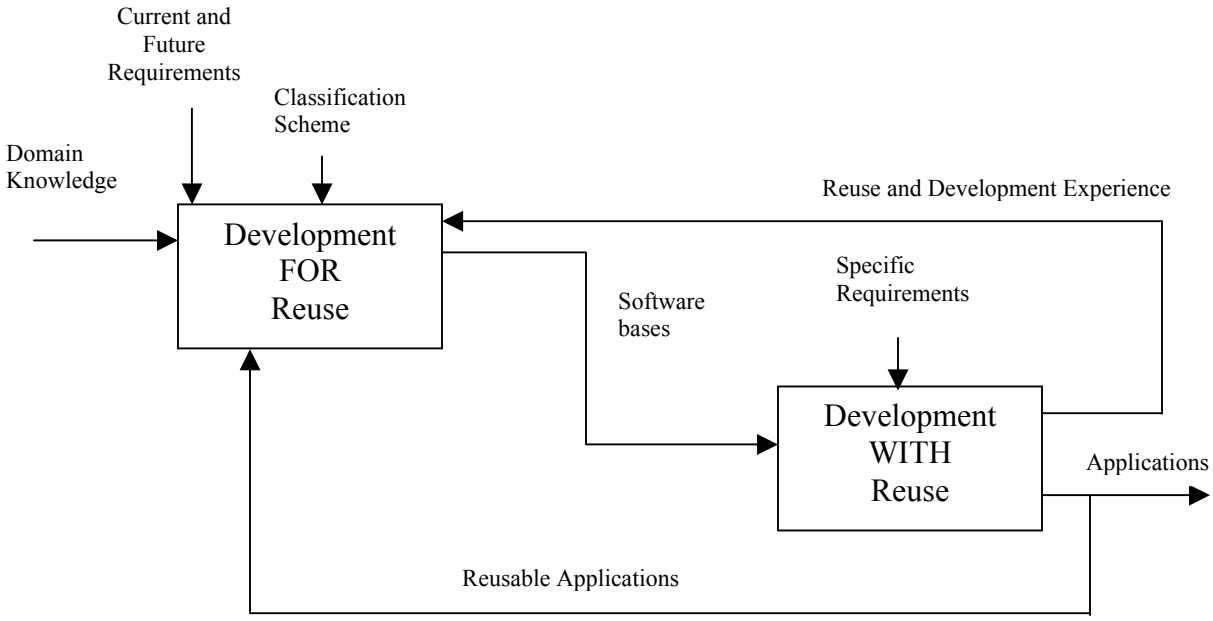


Figure 2.1: Reuse Life Cycle Activities

2.4 Component Classification and Retrieval Techniques

Component classification techniques are used to categorize the software components in various domains. At the time of storing any component in the repository, it is desired to store it under appropriate category. It will help later at the time of retrieval. But many times the same general-purpose functionality may be stored in more than one cluster or domain.

Retrieval techniques are used to locate the appropriate asset from the repository for current set of requirements. It require one to one matching of component specifications with the given set of requirements. Appropriate component classification and precise component specifications are very useful for efficient and convenient retrieval. One of the major activities in software reuse is component classification and retrieval. They include even repository organization techniques also. There are various classification and retrieval techniques based on natural keywords, formal specifications and intelligent systems [48, 146, 147]. Existing classification and retrieval techniques are discussed in this section.

2.4.1 Faceted Classifications

For cataloging components in software repositories Prieto-Diaz proposed faceted classification scheme [123]. In this scheme some attributes called facets are used as descriptors of a software component. These descriptors mainly focus on the action that component performs and on objects manipulated by the component. Scheme consists of four facets: function performed by the component, object manipulated by the function, data structure where the function takes place and system to which the function belongs. The facet classification has the ability to explore domain knowledge with the use of terms extracted from fix number of mutually exclusive categories/facets. Each faceted scheme may have several facets, which may consist of several terms. While retrieving the components, in query the value of each facet is given and is matched to the facets of stored component. The similarity between the components is considered using the conceptual distance between the facets. The components having the threshold value of similarity are retrieved. Facet classification is beneficial in domain specific applications. The major

drawback of facet classification is that it requires manual indexing which makes it expensive [85].

2.4.2 Automatic Indexing

Free text indexing systems automatically extract keywords from user queries in natural language and these keywords are used to locate documents. Similarly software components are classified in a software repository by indexing them according to keywords extracted from the natural language documentation of the software. The system extracts lexical, syntactic and semantic information from the documents and this information is permanently kept in an index. Based upon the frequency of the keywords, most often keywords are included in the index. Classification of software components is done by indexing the software component according to keywords extracted from the natural language documentation of software components. In retrieval of component the keywords extracted from query is matched with the index keywords and the relative components are retrieved. Automatic indexing methods in information retrieval are of two types: Statistical approaches, these approaches make use of a stop list to exclude high frequency words from documents and queries in natural language. Linguistic approaches have been applied at the lexical, syntactic and semantic level [2, 10, 132].

2.4.3 Signature Matching

Type matching and type transformation are mainly used in signature matching. Components are represented by a multi-set of feature signature. In functional matching the type of function, type of input parameters, local variables and type of output parameters is matched. Signature matching approach assumes that the set of allowable signatures in the system is known. Feature can be: the simple type, the constructed type, and user-defined type. Major problem in signature matching is extraction of signatures based upon functionality of the component. Selection of appropriate signature for multidimensional functionality in itself is a big problem [1].

2.4.4 Formal Specifications

Search techniques primarily use natural language to specify components and user queries. Natural language specifications are inherited by ambiguity, incompleteness and inconsistency. Formal language is the precise way to specify components. While specifying the components, the state, invariant and operations are specified using formal specification language. Formal specification languages are good for modeling. Mainly used formal specification languages are Z, Larch, Petri Net and Promela. The formal specification languages are used to represent the components and the query. Theorem prover is used to match the query with component library specification. The main disadvantage of this technique is that user must know the complex representations, as these formal specifications are tightly syntax sensitive [133, 152, 153]. Also formal specification matching may miss all nearly matched components [5, 22, 38, 168, 169].

2.4.5 Behavior-Based Retrieval

Behavior retrieval exploits the executability of software components. Programs are executed using components and their responses are recorded. Retrieval is achieved by selecting those components whose responses are closest to the pre-determined set of desired responses. Component behavior is testing the components under different constraints with various arguments yielding the dynamic responses. The components are classified in the order of their behavior. In behavioral retrieval technique dynamic behavior of the component is considered to be most important while creating the application. In this approach the execution of software components that is dynamic behavior is used to classify components in the repository. It is tried to locate those components, which behave in the similar manner as the query components [54, 122, 167].

2.4.6 Browsing

Browsing is considered to be search where the goal is not well defined in prior; the goal is ultimately dependent on what is discovered during the search. Browsing is viewed as a user navigating through a graph where the nodes represent library items and the arcs connecting relationships. Thus for a library to be browse able it must be represented as a labeled graph. In this approach the starting point is determined from the user query and to

retrieve the desired component the user navigates from the starting point. The main objective behind this type of search is when the goal of the search is not known a priori and user needs to develop navigation process himself. The main disadvantage of browsing is that it requires user to derive the navigation process, which becomes cumbersome when component repository is very large [127].

2.4.7 Hypertext

Components are organized as interconnected similarity graph in hypertext system. The similar components have a connected edge between them. The user navigates through the graph to reach to the desired node representing the desired component. The semantic related with each node helps user to navigate to the desired component. The main problem of this approach is that developing and maintaining a software base requires great human effort [16, 98].

2.4.8 Fuzzy Logic Based Retrieval

The software components are described using the descriptors in fuzzy based retrieval techniques describing the key characteristic of components. These descriptors are composed from term pairs, reflecting the functionality of component and are constructed from the code and accompanied documentation [4]. These features are in form of list and are assigned a relevance measure expressed by a *weight, a* number between 0 and 1. The classification of software components is done using a descriptor in fuzzy based retrieval techniques. Software descriptor is a *list of features* describing the key characteristics of a software component. These features are extracted automatically. The weights assigned to features are viewed as the fuzzy sets and, fuzzy techniques are used for retrieval. A feature weighing function is used to assign the weights to descriptors. Fuzzy matching between the software descriptors is used for computing the conceptual distance between the components. Compatibility between two descriptors is given by a fuzzy relation. The relation gives the value between 0 and 1, which describes the similarity between the two components. A pre set threshold value is compared with compatibility value, if the value is greater than the threshold value the component is retrieved. The similarity value

depends upon number and weight of the common features and their importance [79, 112, 114, 131].

2.4.9 Neural Network Based Organization

Neural network are used to construct the software library. The library is semantically organized i.e. according to functional similarity of the stored component. The components with similar behavior should be stored near to each other. Self-organizing maps are unsupervised neural network are used to organize component library. The input data represents the feature of the storing component. This input data is mapped to a grid of neurons. The output unit assigned the weight vector of n dimensional. During the learning process the input vectors are repeatedly presented to all output units. The outputs produced by these components are measured in terms of their similarity. For retrieving the component the required functionality is described in a query. This query in turn can easily be transformed into a feature vector, which is fired on to the grid, and the best-matching stored components are searched for. The best matching unit and its neighboring components are retrieved [111].

2.4.10 Genetic Algorithm Based Organization

In this technique components are specified with faceted classification. Components, with similar properties, are classified into clusters. Genetic Algorithms (GAs) are used to form clusters with similar components so that user is able to find components easily [42, 43]. GAs are applied to clustering so that reusable components are classified into optimized clusters $P_k = \{C_1, C_2...C_k\}$ whose number is not fixed. The components are represented by chromosome and its corresponding cluster is represented by gene.

Component	1	2	3	4	5	6	7	8	9	10
Cluster	2	3	3	1	5	3	1	6	1	2

The cluster 1 has three components 4,7 and 9. There are two objective functions one for clustering and another for retrieval. For clustering the objective function is based on the concept that similarity within the cluster is high and similarity between the clusters is

low. The goal of multi-way clustering is to find clusters $P_k = \{C_1, C_2, ..., C_k\}$ minimizing this objective function. For retrieving the objective function is based on the concept of retrieving those clusters, which has high precision and high recall. For retrieving the components user queries are matched with the clusters. The cluster with maximum value for the given objective function is retrieved [8, 23, 110, 113, 129, 140, 148].

The given retrieval and classification techniques are widely used. Comparison between these techniques are analyzed on the basis of

- Component Description: The way the components are described in repository.
- Component Classification: How components are stored in the repository.
- Component Retrieval: Technique used for retrieving the components for reuse.
- User Output: The way components are shown to user.
- Search Process: The technique used to locate the appropriate component.

Table 2.1 Comparative Analysis of Classification and Retrieval Techniques

Scheme	Component Description	Component Classification	Component Retrieval	User Output	Search Process
Faceted Classification	Faceted described functionality	Table of descriptor or facets	Based on facet matching	Approximate Components	Automatic
Automatic indexing	Phrases and keywords	Index is made on the basis of some statically techniques	Retrieved on the basis of term matching	Approximate Matching	Automatic
Signature Matching	Signature of the component and its sub component	Keywords from Semantic	On the basis of signature or partial keywords	Approximate Matching	Automatic
Specification Matching	Specifications are given in formal language in the form of predicates	A hierarchy is formed on the basis of their inclusion	Predicate matching with Theorem Prover or Specification Matching	Exact Component	Automatic
Behavioral Retrieval	Dynamic results or behavior	In the order of their behavior	Components plugging for asked behavior	Components shown according to their responses	Semiautomatic

				their responses	
Browsing	Keyword based	Navigation	Navigation Based	Approximate match	Approximate Search
Hypertext	Graph based Connectivity	Through edge connectivity	User navigates through nodes	Node representing the component	Manual
Fuzzy Logic Based Retrieval	Keywords	Keywords are stored with their fuzzy weights	Retrieved using fuzzy relations	Components are ranked according to their similarity value.	Automatic
Neural Network Based Retrieval	Features	A grid is formed Representing Components	Retrieved by firing feature vector to grid	Best matching unit and its neighbours are shown	Automatic
Genetic Algorithm Based Retrieval	Facet	Clusters are formed	Cluster which has maximum value of objective function	Cluster is shown with their similarity value.	Automatic

2.5 Reuse Libraries

Any organization can obtain the actual benefits of software reuse if all components are stored in the repository in unified and organized manner [35, 46, 155]. In order to make effective use of a reuse library or repository, a user must have complete knowledge of its contents. Contents of the libraries can be represented in following two ways.

2.5.1 Vertical Library

Vertical library contains domain specific components [56, 125]. The advantage of such a library is that reusing its contents can reduce the amount of code that we must develop by up to 65%. The drawback of such a library is that it can be used for only a specific application domain. Vertical libraries are particularly useful for organizations that develop and maintain large scale and long-lived products.

2.5.2 Horizontal Library

Horizontal library contains generic components. Generic components provide general-purpose programming support on top of programming language [63]. Such components include abstract data types, menu drivers, graphics utilities and mathematical routines.

The advantage of such library is that it is useful across application domains. As it provides general-purpose components, its leverage is limited to about 20% of the size of the application.

In horizontal libraries, components serve to various application domains so their number grows very large. It is very difficult to manage large repositories. In present work an attempt has been made to develop a horizontal repository and to manage it efficiently. In chapter 3, a retrieval technique using natural and formal specifications has been developed. This technique helps to retrieve relevant component depending upon the development phase in which user wishes to locate reusable component. In chapter 4, a genetic algorithm based technique to retrieve most relevant component from the repository has been developed. This technique makes use of various attributes of reusable assets. This has been validated considering three example cases. In chapter 5, an ant colony based rule generation for storing and retrieving the reusable components has been developed. This technique first creates clusters of components on the basis of attribute interactions and then allows retrieving the relevant component.

2.6 Existing Software Reuse Repositories

2.6.1 Code Broker - Active Reuse Repository System

Component repository systems equipped with active information delivery mechanisms are called active component repository systems [118, 119]. Active reuse repository system (ARRS) does not depend on the user to generate the query but itself generate the queries based on semantic analysis and signature matching.

The benefits of active delivery are that it helps in reduction of perceived reuse cost and also help easily to access components that exist in repository but are unknown to the programmer. *Code Broker* is a prototype of an active reuse repository system. Code Broker consists of three software agents Listener, Fetcher, and Presenter that function in response to the changes in its running environment without requiring human guidance or intervention [121]. Listener agent extracts and formulates reuse queries by monitoring developer's interaction. Those queries are then passed to Fetcher, which retrieves the matching components from the reuse repository. Reusable components retrieved by

Fetcher are passed to Presenter that uses the user profile to filter out unwanted components and display the result to the user. Code Broker uses both probabilistic model-based indexing and retrieval techniques and the Latent Semantic Analysis (LSA) technique to compute concept similarity. Code Broker is similar text-based approach. The component repository in Code Broker is created by its indexing subsystem called Code Indexer that extracts and indexes functional descriptions (concepts) and signatures (constraints) from the documentation. It may be able to match more diverse set of examples, however effectiveness of this approach may be limited because of the difficulty in writing appropriate comments.

2.6.2 SHORE – A Hypertext Repository in the XML

Software Design & Management (SD & M) **H**ypertext **O**bjects **RE**pository (SHORE) stores information extracted from XML documents [170]. These documents are generated by parsers that convert project documents of any type into XML. SHORE architecture can be easily mapped to the four level model architecture of XML metadata Interchange (XMI). Due to this mapping SHORE can be complemented by other XMI capable tools. SHORE combines the features of a repository with a hypertext system based on Internet technologies. It is used in development projects at SD & M to create a consistent view of documents developed during different project phases, e.g. requirement specifications, class models, and source code. In SHORE repository Project documents are integrated by transforming them into a uniform XML based format and their navigable information is stored in an object-oriented database. SHORE's meta meta model consist of three hierarchical types of elements: *document types, object types and relationship*. More over attribute a (name, value)-pair can be associated to document, object and relationship type instances. An attribute must be exclusively bound to a resource or relationship type. SHORE uses an exchange format based on XML. For each document type, there is a parser that transforms code into a well-formed XML document. Connections between objects in the repository can be followed by hyperlinks that are inserted at locations where objects or relationships are defined.

None of the standards for multimedia document models HTML, DHTML and MHEG-5 offers sufficient support for all requirements arising from advanced multimedia

applications. HTML can hardly be characterized as a multimedia document model because it lacks support for even the most basic multimedia requirement, a temporal model. Though XML can become a quite powerful multimedia document model, it still lacks support for reuse at all levels of granularity and suffers from a low semantic level of content description. It mainly describes the presentation and not the structure of document; furthermore, reuse at the level of fragments is severely hampered due to the inflexible scene-based document structure.

2.6.3 CKBR-P - Knowledge based repository

CKBR is knowledge based repository system that classifies and describes business components for their effective storage and retrieval using the classification and coding (C&C) scheme. The CKBR-P was developed using the Visual Basic programming language and contains a Graphical User Interface to interactively guide users through a set of screens during component search. Repository used structured identifiers to classify components on well-defined characteristics (such as domain and implementation environment) and semi-structured descriptor facets to define functionality, business rules, and role of components for characterizing business components at various levels of the abstraction hierarchy. During searching for components structures identifiers help users to quickly reduce a large set of components to a select few, and semi-structured descriptors (coded as facets of the components) provide the ability to closely examine the smaller number of components to determine a match with specific requirements.

Under this scheme first identifiers (e.g., name, industry type) are listed for structured information and after that descriptor facets (e.g., rules, functionality) for unstructured information of the component are listed. Repositories classify each component according to the industry in which it could be used and the application domain it represents. User can query the CKBR to find components that fulfill their requirements. Searching of components is based on structured identifiers to reduce the choice to a smaller number of candidate components. User can also refine the search by using the semi structured knowledge and their constituent parts. The main limitation of knowledge-based repository is that it is very difficult to change or upgrade the knowledge. Also there exists problem in capturing knowledge of new situations [142, 155].

2.6.4 The ORDB-based SFB-501 Reuse Repository

Object Relational Data Base Repository (ORDB) based 501-Reuse-Repository is designed to support all phases of a reuse process and the accompanying improvement cycle by providing adequate functionality [162]. Its implementation is based on the object relational database. Main objective of this repository is to store every kind of software engineering experience called experience element (EE) to improve the quality in future projects. This Reuse repository structure consists of two logical sections called Organization-Wide Section (OWS) and Experiment-Specific Section (ESS). OWS is an instantiation of Basili's EB and the ESS consist of experiment documentations. Inside repository each representations of EE is stored in a set of BLOB (binary large objects), which is a special kind of ORDBMS data type. For retrieval purposes, each EE needs to be associated with a *characterization vector* (CV) containing describing data, e.g., relevant for similarity-based search of potentially reusable design artifacts. For performance reasons, EEs (containing large objects) and corresponding CVs (comparably small amount of data) are stored separately.

Whereas all EEs are stored using the same data structure, the CVs are classified into semantically different data structures. CV attributes depend on the section (OWS, ESS) that the corresponding EE belongs to. More precisely, the sections are divided into logical areas, and the areas determine the CV attributes.

The object-oriented capabilities of object-relational database technology allowed to effectively map the data structures to a database schema. At its user interface, the repository provides functions specifically supporting the different quality improvement paradigm phases especially for planning and executing a project as well as analyzing project data and integrating results. Some conceptual XML structures such as aggregation, composition and association can also be preserved using these new data types. Unfortunately, very often when ORDB is used for XML repository, the complex structures are not utilized. The collection is usually flattened or split into an entirely separate table

2.6.5 AIRS - Artificial Intelligence Repository System

AIRS is an AI-based library system for software reuse, which was developed by E.J. Ostertag, J.A. Hendler, R. Prieto-Diaz, C. Braun. AIRS allows a developer to browse a software library in search of components that best meet some stated requirement. A component is described by a set of (feature, term) pairs. A feature represents a classification criterion, and is defined by a set of related terms. AIRS also allow representation of packages that is logical units that group a set of related components. As with components, packages are described in terms of features. But unlike components, a package description includes a set of member components. Candidate reuse components (and packages) are selected from the library based on the degree of similarity between their descriptions and a given target description. Similarity is quantified by a non-negative magnitude (called distance) that represents the expected effort required to obtain the target given a candidate. Distances are computed by functions called comparators. Three such functions used are subsumption, closeness, and package comparators. The AIRS classification approach is based on a formalization of the concepts and is similar to faceted classification. The internal representation of software library is constructed using a frame system with multiple inheritance. AI research has produced knowledge representation languages such as predicate calculus, semantic nets and production rules to capture the data semantics that can not be realized with traditional database systems. Also these systems make use of keywords and semantics associated with them. A limitation of this approach is the lack of semantics associated with keywords and the meaning of the set of keywords can neither be inferred nor can these systems determine whether two different keywords represent the same concept [48, 168].

2.6.6 VRSI - A Virtual Reuse Repository

The VRSI system is a prototype of a Virtual Repository Supporting Semantic Interoperation based on the Improved Relevancy Matching and Ranking method **(**IRMR) that allow the users to retrieve components from multiple repositories via a single representation view [165]. The basic idea used in this repository is to map the queries for one repository to those for the other using the correlation between terms in two different

repositories. Based on the correlation, a query represented by the terms in *A* can be automatically transformed to a query represented by the terms in *B*. The users can use these queries to retrieve components in *B* transparently. The IRMR approach used by this repository system first calculates the correlation between the terms in two different repositories. The correlations are automatically calculated based on the mutual components in both repositories and recorded in a term-to-term association matrix. Once correlation calculation is performed query transformation step occurs. In which query represented by the terms in one repository is transformed into a query represented by the terms in another repository based on the correlation to retrieve in the repository. At last the retrieved results are ranked using the calculated correlation. In VRSI system the query is based on the Boolean model, i.e., a query is composed of index terms linked by three connectives: *not*, *and*, *or*. The users do not need to formulate the intentions as queries for each repository individually in the changed cognitive model. They only need to formulate the intentions as the query represented by the vocabulary in their familiar repository *Ri* instead. The *translator* can automatically translate the query in *Ri* to queries for each repository (R_1 to *Rn*) to retrieve the components. VRSI system consists of three functional modules: the *calculator*, the *translator* and the *sorter*, and the *relevancy matrix repository*. The *calculator* module calculates relevancy matrix M_{ab} and M_{ba} between 2 repositories (suppose A and B) based on the catalog information in the physical repositories and stores them in the *relevancy matrix repository*. In the *translator* module, users' queries represented by the representation method of their familiar repository are automatically translated to the queries represented by the representation method of other repositories. These queries are used to retrieve components in the physical repositories separately. At last, the *sorter* module ranks the retrieved components.

2.6.7 GURU Repository system

GURU system has been applied to construct an organized library of AIX utilities [166]. GURU automatically assembles conceptually structured software libraries from a set of un-indexed and unorganized software components. In the first stage GURU extracts the indices from the natural language documentation associated with the software components to be stored by using new indexing scheme. This indexing scheme is based

on lexical affinities and on their statistical distribution. It identifies set of attributes for each document to represent a functional description of the associated software unit. After that GURU assembles the indexed object into a browsing hierarchy by using a hierarchal clustering technique that draw information exclusively from the indices identified from previous stage.

Thus GURU support both *classical linear retrieval* in which candidate are ranked according to the numerical measure that evaluates how well they answer the query and *cluster based retrieval* in which the browser hierarchy direct search the best candidate. GURU apply a pure information retrieval approach by automatically building free text indices that characterize software components. For more effective retrieval GURU also use a free text method that is richer than a single term indexing. GURU perform a search and retrieval technique using a conventional inverted index file structure. GURU used the hierarchal agglomerative clustering (HAC) method because their search and construction techniques are more efficient. In GURU user can express their queries in natural language and then the indexing component is applied in order to translate the query into attributes understandable by the system. Exactly the same technique is used for extracting LAs from natural language queries as from natural language documentation. In order to retrieve the best candidate for a given query GURU apply the IR method which consists of considering the query as a document and retrieving the components in the repository whose profile are the most similar profile of the query. It is necessary to include conceptual information about functionality in the indices. Unfortunately, conceptual information is very difficult to get. A very few programmers provide conceptual indices with their code. Natural language documentation is very rich source of conceptual information, however this information is contained implicitly in an unstructured way and is not usable as usually available.

2.6.8 LaSSIE- Knowledge based Software Information System

LaSSIE is a Knowledge based system that incorporates a large knowledge base, a semantic retrieval algorithm based on formal inference, and a powerful user interface incorporating a graphical browser and a natural language parser. LaSSIE is intended to help programmers find useful information about a large software system. The LaSSIE

system is an attempt to build a software information system that integrates architectural, conceptual and code views of a large software system into a knowledge base (KB) for use by developers. The KB is built using a classification-based knowledge representation language KANDOR to provide semantic retrieval. Besides serving as a repository of information about the system (i.e. description of reusable object, architectural constraints) KB also serves as an intelligent index for re-usable components. In this system the classification-based query processing done by KANDOR. Most of the LaSSIE KB describes actions. The query from a user describes an operation that defines examples use want to reuse or to understand in detail. Inference is used to answer the programmer's queries about re-usable components. LaSSIE system accrues several key advantages by using a frame system like KANDOR which incorporates classification as its foundation. These advantages include Aggregation of information about individuals, Semantic retrieval, Use of classification and inheritance to support Updates.

Knowledge based systems are usually smarter than information retrieval systems, but in the knowledge based approach the reuse tool aims at understanding the queries and functionality of components before giving any answer. Some of the knowledge-based systems are context sensitive and produce results depending on user expertise. They require some domain analysis and pre encoded semantic information that is usually provided manually. The main problem of applying this approach in the context of software libraries is that many domains can not be easily confined and the domain analysis is very difficult. This makes the construction of such systems very tedious and expensive [104].

2.6.9 Web Composition repository

Web Composition Repository is a key tool for a systematic approach to code reuse. The repository is used for storing, managing, and retrieving large numbers of web composition markup language (WCML) components. Consequently, it facilitates reuse in component-based Web Engineering. The Web Composition approach is based on modeling with components. A component in the Web Composition model is a code abstraction of any target language, e.g. HTML, script code, or Latex.

The Web Composition model is object-oriented, which allows code-reuse when creating new components as a component may be constructed using existing components. This is achieved using specialization (inheritance) and aggregation (has-part relation). The Web Composition Repository works on top of the Web Composition Component Store. The Component Store provides persistent storage to components. The Repository's basic mode of operation is the cooperation of three system entities: the Component Store, a Metadata Store, and a search or browsing tool. These tools find appropriate Metadata Stores using the Repository Broker. The Repository Broker is responsible for the coordination of currently participating and future system entities, thus ensuring an open model. It is used for selecting Metadata Stores by interfaces. The broker maintains a database of interfaces of all available Metadata Stores. The broker can then provide means for the tool to access the interface of a selected Metadata Store directly. The Component Store is seen as a "flat" storage containing no more information than the components. The Metadata Store can be used to store additional indexing, browsing, hierarchy, or any other Meta-information. A typical Metadata Store defines a graph having components as vertices and edges that represent some relation between the components. Moreover this repository also provides viewers and query tools for retrieving components, analyzing components, or offering any other service using the Repository. Moreover it is possible for the Metadata store to manipulate metadata as a reaction to query tool user actions. Techniques like adaptive indexing could be used to rearrange components in order to group them according to previously made requests. In such a way the repository could organize its own structure to fit the users needs [37]

2.6.10 ROSA Repository

ROSA (Repository of Objects with Semantic Access) is a Learning Content Management Systems (LCMS) that provides the creation, storage, reuse, retrieval and management of learning objects (LO) [31]. In ROSA, course structure and content are organized according to Conceptual Maps. The ROSA system has been developed to provide efficient LO retrieval. ROSA model is based on the description of LOs by a standard set of metadata descriptors and semantic associations among LOs. Query processing techniques for the ROSA data model explores both LOs metadata attributes and semantic

associations among LOs. The combination of standard metadata descriptors with semantic associations contextualizes and individualizes LOs, providing more precise answers to user queries. Main data structure of ROSA data model is a LO class whose instances are described according to LOM metadata attributes. The LO class is specialized into Logical and Physical LO. The ROSA algebra was conceived to support query processing on the ROSA data model with emphasis on the processing of LO metadata and relationships. The algebra is complemented with initial set of equivalence rules, opening a path to query transformations and query optimization techniques. The operations in ROSA receive as input data of type *collections of collections of* complex resources. The model distinguishes two types of collections: external and internal. External collection consist of more general collection class giving a structure to complex resources, whereas the internal data structure corresponding to the tuple concept of the Relational model. Thus, an external collection contains one or more internal collections, and each that internal collection contains one or more complex resources. A query execution engine (QEE) developed to support the execution of ROSA algebra operations. The QEE was developed as an instance of the QEEF (Query Execution Engine Framework) software framework.

Due to the complexity and diversity of semantics in human languages, the problem of semantics being diluted or misunderstood right from the very moment semantics is created and throughout the subsequent transmission processes. When the abstract semantics is expressed in words, that are concrete and explicit, the void of a common vocabulary may lead to alterations in the original semantics when expressed in words, since different people may have different choices of words, thus resulting in semantic gaps. As human languages contain homonyms and synonyms, the same vocabulary may reflect different semantics in different contexts. Such nuances are often not recognized by machine and results in the semantic gaps. Human beings interpret semantics based on a mental model that has been constructed with knowledge and experiences accumulated over time, and the same semantics may be perceived differently under different mental models.

These repositories are compared in table 2.2 on the basis of certain factors such as Retrieval technique used, whether repository is web based or not, Component

specifications in repository, Security feature, Update facility and the information stored in the repository. In all explored repositories it has been observed that few are better in one aspect and few in the other. Still research in the area of constructing efficient repositories is wide open.

Table 2.2 Comparative Analysis of Existing Repositories

Repository	Retrieval Technique	Web Based	Component Specification	Security	Update Facility	Organization	Availability	Information stored
ARRS	Keyword, Profile and Signature matching	N	Functional Description (concept) and signatures (constraints)	N	N	Automatic Generation of Query	Public	Java Class Files
SHORE	Facets Approach	Y	Classification or Textual description	N	Y	This stores information extracted from XML documents using a variant of R-trees and B-trees structure. .	Public	Software Artifacts i.e. requirement, Design, Code
CKBR-P	Facets Approach	N	Classification and Coding Scheme (C&C)		Y	Effective C&C scheme for the subsequent retrieval of relevant components is used	Public	Knowledge Based (Structures Identifiers and Semi conductor Descriptors)
ORDB	Similarity based Approach	N	Representations of experience elements (EE) is stored in a set of BLOB Data-Type			It consists of two logical sections called Organization-Wide Section (OWS) and Experiment-Specific Section (ESS).	Private	Software Artifacts
AIRS	Faceted classification	N	Component is described by (feature, term) pairs	Y	N	The internal representation of library is constructed by a frame system with multiple inheritance	Public	Ada Packages and Set of C components
VRSI	Relevancy matching and ranking method (IRMR)	Y		N		It maps the queries for one repository to those for the other using the correlation between them.	Public	Relevancy matrix between Repositories

Table 2.2 Comparative Analysis of Existing Repositories (Contd…)

GURU	Linear, Cluster based and Free Text Method		Set of attributes associated with components		N	Automatically assembles structured libraries from set of un-indexed and unorganized components	Public	Library of AIX utilities
LaSSIE	Semantic retrieval	N	Classification scheme		Y	The KB is built using a classification-based knowledge representation language KANDOR		Knowledge Base i.e. architectural, conceptual and code views
Web composition	Browsing	Y	Tag-based definition of Components, properties and relationships between components			The Web Composition model is object-oriented	Public	WCML components.
ROSA	Retrieval by Analogy		Learning Object are organized as conceptual Maps			It is based on object oriented specification	Private	Learning objects

2.7 Conclusion

Hybridization of two or more retrieval and classification techniques can be a better alternative for software repositories. It may result in improved specification matching. In this chapter, various classification and retrieval techniques have been reviewed critically. Also various existing reusable software repositories are reviewed. Demerits in existing repositories are highlighted. Rest of the thesis describes our work that addresses some of these demerits.

Chapter 3

Formal Specifications Support for Software Reuse

3.1 Introduction

Formal methods refer to the use of mathematical and formal logic techniques in the specification, design and construction of computer systems. Formal methods allow the logical properties of a computer system to be predicted from a mathematical model of the system by means of a logical calculation. Formal methods facilitate to verify mathematically that a certain description of a system is internally consistent, whether certain properties of the system are consequences of proposed requirements or not, also it helps in evaluating whether requirements have been interpreted correctly to derive the design of the system. Formal methods provide developers ways for reducing or removing subjectivity in the stated requirements. They can also be used to establish and maintain strict trace ability to system descriptions across various life cycle development phases. These methods also help in specifying certain properties as independent and isolated system, so that they can be described as an independent module. A specification is formal if it is expressed in a language made up of following three components [5, 19, 22, 29, 38]:

- Rules for determining the grammatical well-structured of sentences (the syntax)
- Rules for interpreting sentences in a precise, meaningful way within the domain under consideration
- Rules for inferring information from the specifications, useful for design and implementation

A formal specification is a concise description of the behavior and properties of a system written in a mathematical based language, specifying what a system is supposed to do as abstract as possible. The formal language should allow the specifications to be organized into units linked through structuring relationships such as specialization, aggregation, and instantiation etc. Each unit generally has a declaration part and an assertion part, in declaration part all variables are declared and in assertion part all properties on the

variables are formed. In software reuse, while developing with reuse developer needs to know the available components that match or nearly match the present requirements. Simple keyword match based on natural language will face the problem of ambiguity. In these type of specifications single term and its all synonyms can be used to match the given context. But due to inherited ambiguity problem simple keyword based search for software component match may not be that effective. Also matching only considering formal specifications, due to strict syntax will lead to only exact match. It will miss all near matches; these near matches are very useful in case of software reuse. We have proposed a matching technique that makes use of both types of specifications: natural keywords and formal specifications in Z.

The proposed work has two major contributions, first various formal specification languages have been explored and compared on the basis of various attributes; their domain of applicability has been also discussed. Then appropriateness of Z language has been discussed for reusability. Secondly a flexible weighted approach using natural and Z specifications is developed and demonstrated. Benefits of both type of specifications, natural keyword based and formal specifications have been exploited to identify most appropriate type of math for current context.

3.2 Formal Specification Languages and Methods

Problem specifications are essential for designing, developing, testing, communicating, reengineering and reusing solutions. Formalism helps in obtaining high quality specifications within these processes and it also provides the basis for their automation. It is very difficult to write correct specifications probably as difficult as writing a program. Semantics of formalism help to overcome many of the problems with natural language. The main characteristics of formal specifications are adequate, internally consistent, unambiguous, complete etc. Formal specification techniques differ by the particular specification paradigm they are based on [22, 38, 76]. Specification languages may be history-based specifications, state-based specifications, transition-based specifications, functional specifications and operational specifications [100]. Computer based tools can support formal specifications and proofs for automation. In this section we have explored

all types of formal languages and tried to compare them on the basis of certain parameters.

3.2.1 RAISE Specification Language

RAISE stands for Rigorous Approach to Industrial Software Engineering. RAISE emphasizes the use of formal mathematical techniques in the development of software, in requirement analysis and formulation, specifications, design and development. RAISE provides the RAISE Specification Language (RSL) and the RAISE method. RAISE method is based on stepwise refinement paradigm. The aim of the RAISE is to enable the construction of more reliable software, software with fewer errors and more easily maintainable software and better-documented software [46].

RSL is a wide spectrum specification language as it may be used to express high-level abstract requirements as well as low-level designs using explicit imperative constructs. Most advantageous of this is that users only have to know one notation in addition to the programming language used for the implementation, since the whole development is carried through in RSL. RAISE is useful for developing any system with inherent complexity and strong reliability requirements. Typical applications could be systems software, embedded systems or safety critical software.

3.2.2 The B Method

The B method is developed by Jean-Raymound Abrial and his team of researchers at BP Sunbury in the later 1980s. This method is based on a wide spectrum pseudo programming notations, Abstract Machine Notation (AMN), which provides a common framework for the construction of specifications, refinements and implementations. AMN provides structuring mechanisms that support modularity and abstraction in object-based style. B draws advances in the formal method that span the last thirty years that include Z notations, pre and post conditions, guarded commands and refinement calculus [Jacques et al. 2006]. It synthesizes them into a unified pragmatic and usable development. A software system conceived with this method is composed of several abstract machines. Each machine contains some data and offers some operations. The data can not be reached directly rather they are always reached through the operations. Each operation of an abstract machine is an atomic action. The initial model of the abstract machine is

finally converted to executable module. This passage from specifications to code is entirely controlled by the method [151].

3.2.3 CSP

Communicating Sequential Processes (CSP) is a formal language to describe parallel systems. CSP is developed by Tony Hoare in early 1980s. CSP describes processes in terms of entities that interact using events. It is an important part of CSP that if you hide the interactions between a group of entities, but not the interactions between the group and the environment, that you end up with another process that is process compose. In CSP processes can be described in terms of the events they may participate in, and groups of processes can be described in terms of the traces of events in which they did participate. CSP has been used as basis for the programming language Occam. CSP has been extended to address tasks such as hardware co design [55].

3.2.4 Larch

The Larch has two components Larch Interface Languages and Larch Shared Language (LSL). Interface languages are used to specify the communication between the system components. Each specification provides the information needed to communicate through a specific interface. A critical part of each interface is how components communicate across the interface. These communication mechanisms differ from one language to other. These differences lie in signaling exceptional conditions, parameter passing and storage allocation mechanisms. Larch interface languages have been designed for a variety of programming languages such as Ada, C, C++, CORBA, Modula-3 and Smalltalk. There is also a generic Larch interface languages that can be specialized for particular programming language or used to design interfaces between programs in different languages [141].

The Larch Shared Language, LSL is used to specify mathematical theories. LSL is an equational algebraic specification language i.e. specifications in LSL mainly consists of first order equations between terms. The semantic of LSL is based on classical, multisorted first order equational logic but LSL do have two second order constructs that allow to specify rules of data type induction and certain type of deduction rules. Unlike

programming language, LSL is purely declarative mathematical formalism. In LSL mathematical operators and their theories are specified.

3.2.5 Vienna Development Method

The Vienna Development Method (VDM) is a formal technique of the model base developed in IBM Vienna laboratory in about the middle of 1970s. VDM tools support precise modeling in structured and well-defined notations. There are the ISO Standard specification language VDM-SL and its object oriented extension VDM++. Use of VDM varies from abstract specifications till development; each step involves Data Reification and then Operation Decomposition. Data reification develops the abstract data types into more concrete data structures [52]. Operation decomposition develops the abstract implicit specifications of operations and functions into algorithms that can be directly implemented in any programming language. VDM-SL consists of a mathematical model built from simple data types like sets, lists and mappings along with operations [70].

3.2.6 Business Object Notation

Business Object Notation (BON) is a method processing as well as graphical language for specifying Object Oriented systems. The fundamental construct in BON is the class. It provides mechanism for specifying classes and objects, their relationships and assertions in first order predicate logic, for specifying the behavior of routines and invariants of classes. BON class diagrams consist of one or more classes organized in clusters drawn as dashed rounded rectangles that may include classes and other clusters. Classes and clusters interact via two general kinds of relationships inheritance and client-supplier. The BON process is not use case driven but it is architecture centric. BON is based on the principles of seamless ness, reversibility and design by contract, making it an ideal basis for industrial strength formal methods development of object-oriented software. It is supported by the EiffelCase tool [76, 120, 157].

3.2.7 OBJ

The specification language OBJ is an example of a language based on object-oriented ideas, which uses equational specifications to define the behavior of objects. "OBJ" refers

to the *language family*, while "OBJ2," "OBJ3", "CafeOBJ," "BOBJ," etc. refers to particular members of the family [71]. The OBJ languages are broad spectrum algebraic programming and specification languages, based on order sorted equational logic, possibly enriched with other logics such as rewriting logic, hidden equational logic, or first order logic and providing the powerful module system of parameterized programming [removed]. All the OBJ languages are rigorously based upon a logical system; more precisely, they are *logical languages*, in the sense that their programs are sets of sentences in some logical system, and their operational semantics is given by deduction in that logical system. All recent OBJ languages use some version of order-sorted algebra, which provides a rigorous basis for user definable sub-types, exception handling, multiple inheritance, overloading, multiple representations, coercions, and more. They also support user definable mixed syntax, user definable execution strategies, rewriting modulo standard equational theories.

3.2.8 Z and Object Z

The language Z is a notation for writing specifications. It is based on typed set theory and first order logic. The language has been developed over a number of years largely through case studies and industrial experience, and during this development a number of conventions have been developed for using the schema calculus to build up specifications. Most of these case studies have used a common paradigm: the system has been viewed as a state machine. The schema calculus has been used to build up a model of the internal state in terms of more or less separate components; operations have been defined in terms of changes to the overall state, and again the schema calculus has been used to specify these operations by merging operations on the components of the state. Z schema consists of two parts: predicates and data invariants. Predicates are the variables of the state and data invariants will hold before and after the operation has been executed. The existence of this calculus is one of the main distinguishing features of Z and it is enormously important in the practical development of large-scale specifications. First, it makes the job of the specifiers easier by allowing them to develop specifications incrementally. Second, it makes large Z specifications relatively easy to read, by structuring them into comprehensible units. Third, it helps with the development of proofs and possibly refinement steps in a structured way [52, 62, 91, 138].

3.2.9 Object Constraint Language

The Object Constraint Language (OCL) is a modeling language with which software models can be constructed. It is defined as a standard add-on to the Unified Modeling Language (UML), the Object Management Group (OMG) standard for object-oriented analysis and design. Some of the most important characteristics of OCL are: it is a declarative, modeling, formal and typed language; it is not a programming language. OCL is an improvement over its competitors in a way that traditional formal languages are usable only to persons with a strong mathematical background, but difficult to use for the average business or system modeler. OCL has been developed to fill this gap. OCL is a formal language, which remains easy to read and write. OCL allows expressions using simple logic operations, first order logic, and also set operations. First order logic allows a wide range of expression allowing universal quantifiers leading to generation of some powerful expressions. The Object Constraint Language is a formal language to express side effect-free constraints [158]. Users of the Unified Modeling Language and other languages can use OCL to specify constraints and other expressions attached to their models. OCL is a formal language where all constructs have a formally defined meaning. The specification of OCL is part of the UML specification, and it is available from IBM and the OMG. The need for a formal language is that in object-oriented modeling, a graphical model, like a class model, is not enough for a precise and unambiguous specification. UML is useful as a way of communicating ideas from one person to another. OCL allows us to help this by making the model more concrete and precise. There is a need to describe additional constraints about the objects in the model and although such constraints are often described in natural language practice has shown that this will always result in ambiguities. To write unambiguous constraints OCL can be used.

3.3 Comparative Analysis

Various specification languages discussed in section 3.2 are compared on the basis of various features. These features include specification style, their mathematical base, concurrency handling, constraints specifications and automation etc. Main strengths and

weaknesses of each specification are also discussed. Detailed comparative analysis of all specification languages is given in table 3.1.

Table 3.1 Comparison of the different Formal Methods

Characteristic	RAISE	Z	B	CSP	OCL	VDM	Larch	OBJ
Specification and Development	It can be used for both specification and development.	It is a specification language, more productive to use another notation for development.	It can be used for both specification and development.	It can be used for specification.	It can be used for specification.	It can be used for both specification and development.	It can be used for specification.	It can be used for specification.
Specification Style	Provides a choice between applicative & imperative styles on the one hand and algebraic and model-based on the other.	Model-based, primarily applicative, style.	AMN is based on a predicate-transformer style of specification	Concurrent specifications are used. Concurrent processes can be defined which communicate with each other via synchronous channels.	Model-oriented specifications	Model-oriented specifications	Algebraic specifications Two-tiered, definitional specification style	Algebraic specifications
Mathematical basis	Abstract, axiomatic styles of description and concrete, operational styles	Set Theory First Order Predicate Calculus	Set Theory First Order logic AMN	Processes, events, channels	Set Theory First Order logic	Set Theory First Order Predicate Calculus	Classical, multisorted first-order equational logic	Order sorted equational logic
Structuring Mechanisms	RSL often combines functions on sub states into one or more larger modules.	Schema Calculus, which allows various schemas to be combined to form new schemas.	AMN has stronger structuring constructs than VDM. It is object-based; information hiding of various kinds is enforced, so that encapsulation of state by operations is the order of the day.		None	None	Several traits can assert different properties of the same operators and then be combined into single trait.(the basic unit of specification is a “trait” in Larch Shared Language)	Structuring mechanisms are similar to views.
Handle Concurrency	Parallel activities can be specified via the RSL concept of a process.	Z is not capable of reasoning about temporal constraints, concurrency and other non-functional requirements.	Handles concurrency	Parallel activities can be specified based on a notion of process algebra.	Does not handle concurrency.	A number of approaches have been worked out for handling concurrency in VDM, none of these having achieved the level of use needed to make it part of the ISO Standard VDM-SL.	Supports concurrency	Supports concurrency

Table 3.1 Comparison of the different Formal Methods (Contd…)

Characteristic	RAISE	Z	B	CSP	OCL	VDM	Larch	OBJ
Exception Handling Support	No exception handling support.	Not explicitly supported in notation. Defined by specifying an operation using structuring mechanisms.	Provides exception handling support.	Facilitates a concurrent exception handling mechanism.	No exception handling support.	Has notation to mark and define error conditions	Provides exception handling support.	Provides a rigorous basis for user definable sub-types, exception handling, multiple inheritance, overloading, coercions, and more.
Preconditions, Invariants and Proof Obligations	Implicit definitions using pre- and post-conditions	Precondition is not explicitly stated; it has to be calculated from the delta schema definitions.	The preconditions are explicit.		Explicit definitions using invariants, pre- and post-conditions	The preconditions are explicit. Invariants play a secondary role, as a proof obligation rather than a specification mechanism.	The foundation of behavioral specification is the use of pre- and postconditions.	Functional Specification comprises of exact procedure or function signatures coupled with defining pre-, and post-conditions.
Model or Property Oriented	Model-Oriented (allows to describe a model of the system to be developed)	Model-Oriented (allows to describe a model of the system to be developed)	Model-Oriented (allows to describe a model of the system to be developed)	Model-Oriented (allows to describe a model of the system to be developed)	Model-Oriented (allows to describe a model of the system to be developed)	Model-Oriented (allows to describe a model of the system to be developed)	Property-Oriented (provides an axiomatization which defines the properties of the system to be developed)	Property-Oriented (provides an axiomatization which defines the properties of the system to be developed)
Strengths	RSL is able to express functional properties of a system at different levels of abstraction from a high-level axiomatic one to low-level algorithmic details that are close to a program.	The strength of Z lies primarily in describing the state of a model, and the operations which can be performed on that state. It is suited to expressing specifications for small and medium size applications.	Good integrated development method. Highly Structured. Could be used without carrying out proofs, i.e. for structuring and code generation.	The strength lies in defining communications between processes.	To write unambiguous constraints, OCL has been developed. It allows us to help this by making the model more concrete and precise.	Specification is comprehensive & precise. System tests derived from specs, so customer could see level of testedness.	The division into two tiers allows the language used for each tier to be more carefully designed and expressive.	OBJ is capable of expressing data-orientated properties and requirements in a simple, easily learnt and readable manner. It is weaker, and requires more effort and user experience, to apply to process-orientated requirements.
Weaknesses	Its major weakness is the low level of automation in the justification tools. RSL does not have built-in constructs for expressing real-time properties.	In its basic form, it is not capable of reasoning about temporal constraints, concurrency and other non-functional requirements.	Specs perhaps a little limited in style, much development effort put in e.g. by French institutions and BP (original effort).	No mechanism for analyzing emergent behavior.	OCL expressions are unnecessarily verbose. They are hard to read.	Can't distinguish between essential and merely desirable functions & can't specify global properties. Cannot specify usability, performance, reliability & safety reqts.	One sometimes finds oneself writing out a very similar specifications in both LSL and a BISL, that is a waste of effort.	It is weaker, and requires more effort and user experience, to apply to process-orientated requirements.

Table 3.1 Comparison of the different Formal Methods (Contd…)

Characteristic	RAISE	Z	B	CSP	OCL	VDM	Larch	OBJ
Tools	Portable type checker, ***rsltc***, for the RAISE Specification Language (RSL).	Noncommercial Public domain tools providing syntax and type checking include: **Zebra** and **ZTC**. Development tools include **ZADOK**, **Forsite**, **IBM Z Tool**, **ZEE**, **ZEN**, **ICL Z** tool.	**Atelier-B** and **B-tool** both commercial products.	*(Failures/ Divergence Refinement)* FDR, ProBE, Deadlock Checker, Casper, ARC	USE, OCL Checker, Oclarity, OCTOPUS	IFAD VDM-SL, Rose VDM++ Link	LCL, Larch/Ada Larch/Small-talk Larch/C++ VSPEC, Generic Concurrent Interface Language, GCIL	**ObjEx** toolset, Gerrard Software Ltd, UK (UNIX and VAX VMS). **OBJ3**, Stanford Research Institute, USA, and **PRG**, Oxford University, UK.
Applications	Main use is in high-reliability applications in aerospace and transportation.	General purpose, Operating System, Components, Databases, Security, Hardware, Graphics	Railway signaling and train control	Software design Hardware design Security protocol analysis	Navigating and querying models and databases. Specifying aspects, specifying model behavior, in Domain Specific Languages	Air Traffic Control Information System (Real Time System). CDIS displays information about incoming and outgoing flights, weather conditions, and equipment status at airports.		Mature in many "data driven" application areas, including GUI, Secure and Safety critical systems, communications

3.4 Benefits of Formal Methods in Software Reuse

Software component libraries have been suggested as a means for facilitating software reuse. Component retrieval systems based on analysis of natural language documentation have been proposed to construct libraries. Unfortunately software components represented by natural languages may hinder the retrieval process due to the problem of ambiguity, inconsistency and incompleteness inherent to natural languages. Using formal specifications to represent software components can minimize these problems. By using formal specifications to represent software component also facilitates the determination of reusability because they more precisely characterize the functionality of the software, and well-defined syntax makes processing amenable to automation. Major benefits of using formal methods in software development or software reuse are:

1. Formal specifications feature a high degree of logical precision that eliminates much of the ambiguity problem found inevitably in informal specifications. This precision translates into a higher likelihood that all requirements writers and readers have a consistent understanding of requirements and a higher likelihood that the requirements will be implemented correctly.
2. The use of formal specifications and proofs is not all or nothing approach. It can be tailored to the level of rigor appropriate to a given budget, schedule and technical needs. It can be scaled to match the needs of the project. It can be used even for near approximate matches not only for exact matches.
3. Computer based tools can support formal specification and proofs. This provides automation for tasks such as consistency checking and the preparation of proofs.
4. Formal specifications complement testing by providing a precise specification from which better test plans can be derived. They go beyond testing because they have the unique capability of theorem proving to show that key properties are satisfied in entire classes.
5. Formal proofs also eliminate subjectivity from requirements analysis by providing a logical and precise argument for the behavior of requirements.
6. Formal specification and proofs can be used in any phase of software life cycle.

3.5 Development of Flexible Weightage Approach using Z Specifications

The use of formal methods to specify the components helps to remove the ambiguity and inconsistency from the specifications. Various formal specification methods have been explored and compared in the previous section. Z specifications have been used in this implementation due to mathematical set theory representation and various tools are available for automation support. In component repository Z specifications are used along with natural keywords to describe components. User enters both keywords as well as Z specifications in form of schema in query to locate a reusable component for present context. Natural language keywords are used to narrow down the search space and Z specifications are used to exact match the components with query. A large database of components specified with natural language keywords and formal specifications using $L^{A}T_{E}X$ markups is constructed. Each component has 32 attribute keywords representing its functionality and general characteristics as explained below in table 3.2. In [114] authors have used similar type of attributes for rough fuzzy clustering.

Table 3.2 Attribute Vector

Sr. No.	Attribute	Description
1.	Component Name	For selecting candidate components with keyword based search component name is used.
2.	Alternate Names	Helps in keyword based search, like Addition, sum, total, add etc. To identify candidate components in first step, keywords are used.
3.	Language Used	In which language the component in question is developed C, C++, Java
4.	Development Platform	To check its compatibility, whether it is used in Windows, Unix, and Linux. It can be used in network-based system or not.
5.	Arguments	These are parameters in case of functions or methods. Number of parameters, types of parameters play important role in deciding its reusability.

6.	Size	It is number of instruction lines i.e. LOC
7.	Cyclomatic Complexity (CC)	MaCabbe's Cyclomatic Complexity, it is measure of number of independent paths. If two components are available for said functionality, component with lower Cyclomatic Complexity can be selected for saving testing effort.
8.	Domain	Component domain will help to locate appropriate functionality from the repository. Component functionalities are tailored to suit specific domains. Sometimes same component has been used in multiple domains in past.
9.	Designed for reuse	Component is developed as general purpose or specific. It requires additional effort to develop a component with keeping in mind its reusability. General component is easy to reuse.
10.	Specifications	It is formal specifications, such as invariants and operations in component. For present context we are considering only Z language.
11.	Test Cases	Component has been already tested for mentioned test cases, after reuse it need to be tested with some new test cases to find errors for present context, also to avoid redundant test cases.
12.	Test Plans	As per IEEE 1012 Standard for software verification and validation
13.	Certification	Component is certified or not. It is to assess quality of the component.
14.	Developed by	It is author of the component. It may be individual or the team.
15.	Price	It is in terms of LOC or Function points equivalence etc. it will help for cost estimation.
16.	Component History	It is to trace all type of adaptations/changes made in

		parent component. It will also include its reuse history in various instances.
17.	Performance	It is in terms concurrency, stress and throughput. Static data performance and dynamic data performance.
18.	Pre-condition	What pre requisite state is required to use current component.
19.	Post-condition	In what state this component will exit.
20.	Algorithmic Complexity	In order of O(n), to select efficient algorithm in terms of time.
21.	Effort	Effort required in developing this component.
22.	Data Structures	To select component depending upon features, Whether component is developed using arrays or linked list.
23.	Resource Utilization	Time-space trade off, it may also include other computing resources also.
24.	Bugs (Known)	Listing these bugs will help in deciding whether these types of bugs are serious or ignorable for present context.
25.	Limitations	Scope, deficiencies, constraints detected. Impact of these detected deficiencies on overall performance.
26.	Flexibility and Interoperability	With which type of systems (hardware/software) it can work and with the others it can not work.
27.	Quality	Depending on ISO 9126 software quality standard every component is assigned a quantitative number.
28.	Criticality	These are few critical success factors of the components. Also it will include constraints related to volume, peak load etc.
29.	Adaptability	General guidelines to extend or tailor it for some specific environment/system.
30.	Distribution/Availabi	Architecture of the component. It is a two-tier/three-

	lity	tier or n-tier type of architecture.
31.	Keywords	Few keywords that correlate its functionality with other types of components. These will be other than the words at 1 and 2.
32.	Remarks	Recommendations regarding deficiencies, performance in various loads and specific suggestions for use.

In second type of specification i.e. formal specifications consists of declaration predicates, schema definitions and pre/post conditions for each method. With keyword based matching we shall get list of candidate components. One component may have many methods so formal specification matching will be done for all methods of a particular component.

3.5.1 Why Specification Language Z is used

The Z notation is a model-oriented formal specification language developed by the programming research group at Oxford University Computing Laboratory in the early 80s. Since then Z has been used to specify a wide variety of software systems including database systems, transaction systems, distributed computing systems and operating systems. The most notable success of Z is the specification of CICS Application Programming Interface (API) by IBM United Kingdom Laboratories [61]. Approximately 37000 lines of code were produced from Z specifications and designs, and it was reported that the code has approximately 2.5 times fewer problems than the code that was not specified in Z. Z is a non-executable but strongly typed specification language. ZTC is a type checker for Z, which determines if there exist syntactical and typing errors in Z specifications. It has been explored that Z due to mathematical support in form of set theory is bit easier to learn and use. The schema calculus has been used to build up a model of the internal state in terms of more or less separate components; operations have been defined in terms of changes to the overall state, and again the schema calculus has been used to specify these operations by merging operations on the components of the state. The existence of this calculus is one of the main distinguishing features of Z and it is enormously important in the practical development of large-scale specifications. First,

it makes the job of the specifier easier by allowing them to develop specifications incrementally. Second, it makes large Z specifications relatively easy to read, by structuring them into comprehensible units [138, 151]. Third, it helps with the development of proofs and possibly refinement steps in a structured way. Due to all these reasons for present implementation Z has been selected.

3.5.2 Matching Steps

1. Find the components whose keywords and their synonyms match with keywords in the input query.
2. Calculate following similarities for all components found in step 1:
   2.1 Find similarity on the basis of number of keyword match; calculate the percentage match on the basis of 32 component attributes.
   2.2 Formal similarity is calculated considering declarations, predicates, state schemas, operation schema methods and pre/post conditions.
   2.3 Find similarity of formal specification of input schema with the specifications of every method schema of this component
   2.4 Method/Component with highest similarity with input schema is considered and is added to overall similarity calculated on other attributes.
3. Display user relevant components with percentage match in decreasing order.

List of all components is sorted on the basis of overall percentage similarity, User can select appropriate component from the list. Above steps are shown in figures 3.1(a) and 3.1(b).

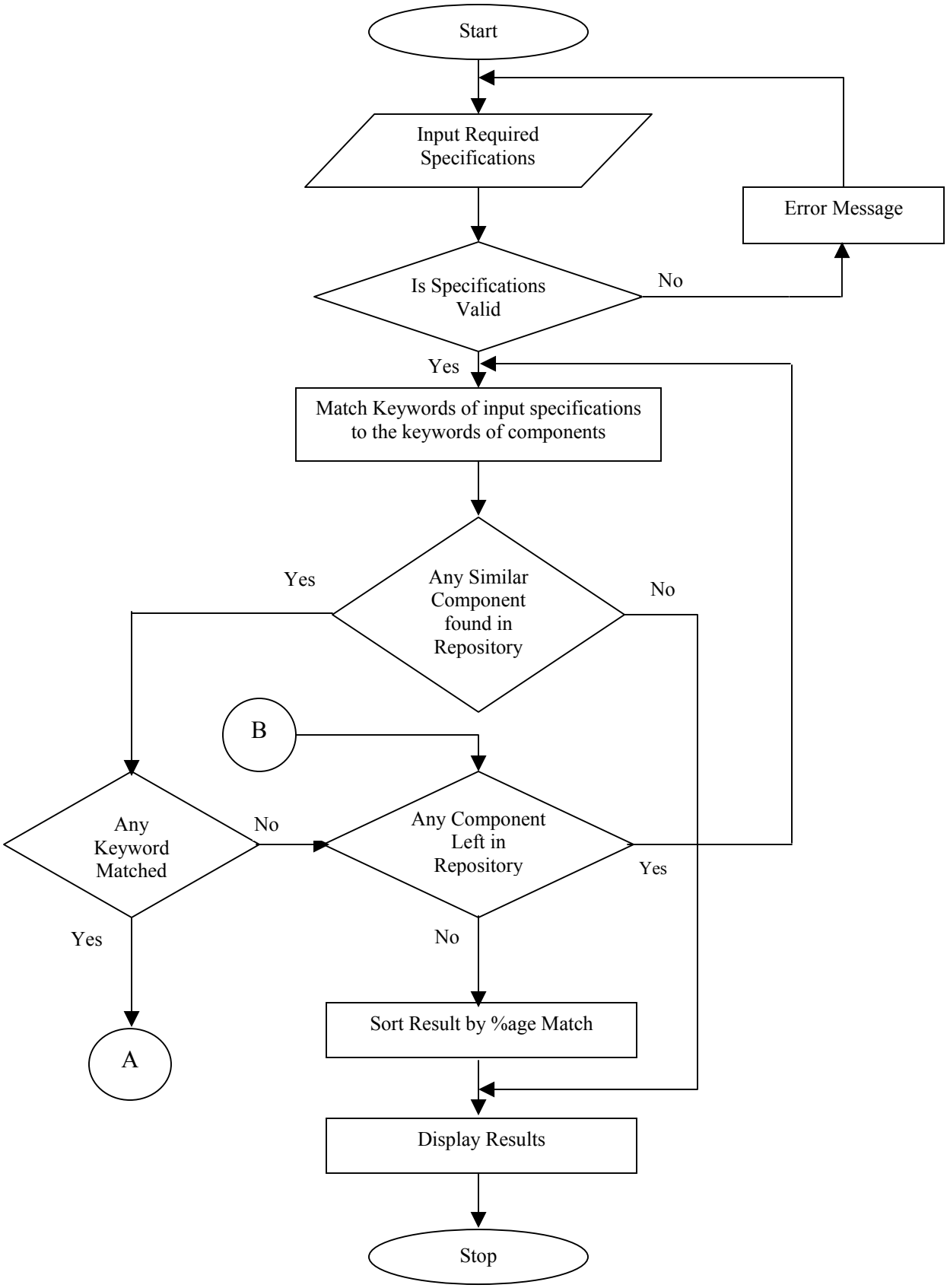


Figure 3.1(a): Step 1 Keyword Matching

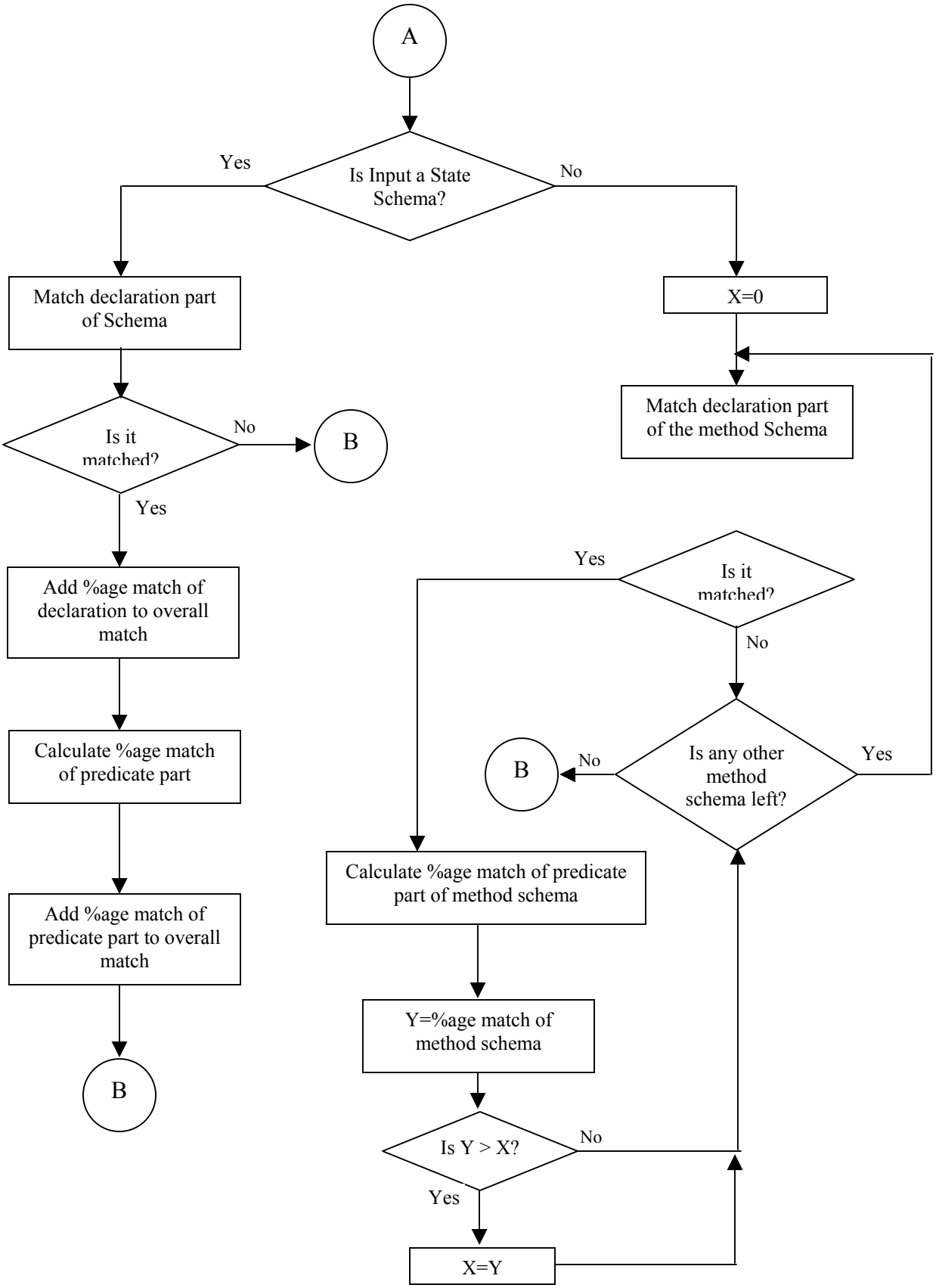


Figure 3.1(b): Step 2 Specification Matching

Table 3.3 Example Input Specifications

Input Specifications	Repository Specification
Natural Language Based: Insert, Delete, Traverse, Linked List, Node Formal Language Based: Declaration Part: \Delta LinkedList data?: Data link?: Next name?: Name Predicate Part: used'=used \ head LinkedList free'=free + head LinkedList LinkedList=tail LinkedList	Natural Language Based: Add, Insert, Delete, Linked List, Update, View Declaration Part: \LinkedList data?:Data link?:Next name?:Name head?:Node current?:Node Predicate Part: used'=used \ head LinkedList free'=free + head LinkedList used $\cap$ free= $\varnothing$ used $\cup$ free= All

$$\text{Percentage Match} = (n_{m1}/N_1 * W_1) + (n_{m2}/N_1 * W_2) + m * (t_{imax})$$

Where

$$t_i = (n_{iz}/N_{pq} * W_3) + (n_{iz}/N_{pd} * W_4) + (n_{iz}/N_{cq} * W_5) + (n_{iz}/N_{cd} * W_6) + (n_{iz}/N_{sq} * W_7) + (n_{iz}/N_{sd} * W_8)$$

m=1 if keyword match exist, otherwise 0

n_{m1}- Number of keywords matched in the component concept descriptions

N_1- Number of keywords input by user in query

n_{m2}- Number of keywords matched in 32 component attributes

n_{iz}- Number of Z notation markups matched

N_{pq}- Number of Z notation markups matched that are entered by user in declaration and predicate part

N_{pd}- Number of Z notation markups present in database in declaration and predicate part of the component

N_{cq}- Number of pre/post conditions specified in user query that are matched

N_{cd}- Number of pre/post conditions of the component in repository that are matched

N_{sq}- Number of markups specified by user for component schema definition that are matched

N_{sd}- Number of markups used in schema definition of the component that are matched in the component repository

All weights assigned to various specification attributes are:

W_1-Keywords entered in query matched with keywords specified in repository

W_2-Match on the basis of 32 Attributes of each component

W_3-Number of markups matched with declaration and predicate parts of Z specifications from user query

W_4-Number of markups matched with declaration and predicate parts and are present in specifications in repository

W_5-Number of markups matched with pre/post conditions of Z specifications from user query

W_6-Number of markups matched with pre/post conditions of Z specifications present in component specifications in repository

W_7-Number of Z markups matched with schema definition specified by user

W_8-Number of Z markups matched with schema definition specified in repository

3.6 Experimental Results

Consider a user query in form keywords and Z markups; values of various attributes of the query are as under:

N1	13	Npd	7
nm1	3	Ncq	3
nm2	8	Ncd	6
niz	6	Nsq	6
Npq	4	Nsd	10

We have considered five various cases of weights assigned to formal and natural specifications. Weights assignment is flexible in our case. As per user requirements weights assigned to a particular type of specifications can be changed. Depending upon weight assigned to natural part of query specifications and formal part of specifications percentage match will vary i.e. the correctness and precision of component match will vary. To show results here, weights assigned to formal and natural specifications are Very High Very Low, High Low, Medium Medium, Low High and Very Low Very High respectively, as shown in Table 3.4. Depending on the degree of rigor required in matching the weights assigned to formal specifications can be increased or decreased.

CASE 1: Consider Weight assigned to formal specifications is 80% i.e. W3=W4=W5=W6=W7=W8=13.33% and weight assigned to natural specifications component is 20% i.e. W1=W2=10%. With this combination of weights the best component match will come out to be 100% means exact match.

CASE 2: Consider Weight assigned to formal specifications is 60% i.e. W3=W4=W5=W6=W7=W8=10% and weight assigned to natural specifications component is 40% i.e. W1=W2=20%. With this combination of weights the best component match will come out to be 86.494%.

CASE 3: Consider Weight assigned to formal specifications is 50% i.e. W3=W4=W5=W6=W7=W8=8.33% and weight assigned to natural specifications component is 50% i.e. W1=W2=25%. With this combination of weights the best component match will come out to be 79.106%. As we decrease the weightage of formal component of specifications the percentage match decreases.

CASE 4: Consider Weight assigned to formal specifications is 40% i.e. W3=W4=W5=W6=W7=W8=6.66% and weight assigned to natural specifications component is 60% i.e. W1=W2=30%. With this combination of weights the best component match will come out to be 71.719%.

CASE 5: Consider Weight assigned to formal specifications is 20% i.e. W3=W4=W5=W6=W7=W8=3.33% and weight assigned to natural specifications component is 80% i.e. W1=W2=40%. With this combination of weights the best component match will come out to be 57.013%.

Table 3.4 Experimental Results

CASES	Percentage Weight Assigned to Formal Specifications	Percentage Weight Assigned to Natural Specifications	Percentage Match
CASE 1	80	20	100.00
CASE 2	60	40	86.495
CASE 3	50	50	79.107
CASE 4	40	60	71.719
CASE 5	20	80	57.013

3.7 Conclusions

In this chapter various formal specifications and methods from software reuse point of view have been explored. These formal specification methods are compared on the basis of various characteristics. A flexible weighted approach using natural specifications and Z specifications has been proposed. In this technique user can select appropriate weights based on current context. This technique helps in finding appropriate component for present situation entered by user in form of query. It is has been observed that weights assigned to various specifications play significant role in matching the components. It has been also observed that if weight assigned to formal specifications is reduced it resulted in reduced percentage match of the component from the repository. It shows formal specifications help in matching appropriate functionality from the repository. But if there is no keyword match then there will be no further matching based on formal specifications. Basically we are pruning the search with first keywords and then from the keyword matches we are comparing functionality with the help of formal specifications. It does not mean that with only keyword match percentage match will be always lesser, but the confidence in match will be lesser with only keyword match due to ambiguity. It is concluded that when project is in early phases of development life cycle and only outlined requirements in form of narrative descriptions are available then user can increase the weight assigned to natural language specifications for component matching. At this stage even if the user is able to locate near matches then also these can be useful guidelines. At later stage in logical design or physical design, user will have some concrete description of the system to be developed then user can increase the weight assigned to formal specifications.

Chapter 4

Genetic Algorithm Based Reusable Software Components Repository

4.1 Introduction

Genetic Algorithms (GAs) are search algorithms based on the mechanism of natural selection and natural genetics [42]. They combine survival of the fittest among string structures with the innovative flair of human search. In every generation a new set of artificial creatures i.e. strings is created using bits and pieces of the fittest of the old. They efficiently exploit historical information to speculate on new search points with expected improved performance. GAs are used to optimize a function or a process. What are we trying to accomplish when we optimize? The conventional view of optimization theory is as presented by Beightler, Phillips, and Wilde according to this "Man's longing for perfection finds expression in the theory of optimization. It studies how to describe and attain what is best, once one knows how to measure and alter what is Good or Bad … *Optimization theory* encompasses the quantitative study of optima and methods for finding them". The optimization seeks to improve performance of any function toward some optimal point or points. This definition has two parts: first improvement to approach some second part i.e. optimal point [89]. There is a clear distinction between the process of improvement and the destination goal. In judging optimization process commonly focus remains on convergence and interim performance is ignored completely. This type of emphasis is in calculus generally; it is not the case in natural optimization. In natural optimization process we are concerned about doing better as compared to others in the category [66, 73, 88]. 74 removed

GAs are different from traditional ways in number of ways such as GAs work with coding of parameters rather than parameters themselves [150]. GAs start with a population of points called initial population, not with a single point. GAs use objective function and probabilistic transition rules rather than deterministic rules. In software development reuse of previously developed artifacts can help in producing timely quality software products. Reuse of existing artifacts cut short the major component of cost. For

implementing software reuse primary requirement is a component repository. Repository should contain large number of components belonging to various development stages. Constructing reusable component and then storing it in repository is *development for reuse* and locating an appropriate component, retrieving it and then integrating it for present context is *development with reuse*. Both types of reuse are equally important for implementing software reuse successfully [64, 65]. Developing reusable software components require extra effort but this can be compensated by reusing this component in various contexts. Development with reuse requires appropriate component selection from the repository. Identifying best-fit component from large repository is a very difficult problem. Identification of appropriate component is only possible if components are described properly in repository. For searching software components from repository, only keywords are not sufficient. It is very difficult to describe components and their functionality with only keywords and synonyms due to large number. Natural language based search also suffers from inherited ambiguity. Describing software components using only formal specifications is also not very good alternative being strictly syntax sensitive [77, 100, 101, 105]. For matching formal methods make use of theorem proving and misses all nearly matching components. It is very rare to find exact match for reusable component and in software reuse near match components can be really advantageous [167, 168, 169].

Motivation to Use GAs

Objective: To optimize the combination of attribute values and to locate the most appropriate component for given context. In given problem 32 component attributes and their relative weights are taken into account while selecting appropriate component. To find the optimal combination of these attribute values is the primary objective.

Constraints:

1. Average fitness value is the indicator of the similarity of the population with the user query.
2. Components with high fitness value will survive longer and components with small fitness value will be eliminated in next evolving generations.

We hereby propose to store every component with Attribute Vector and associated Weight Vector in repository. For retrieving the component a two-steps technique is proposed. In first step, using key word based search we find all candidate components. Further to prune the search and find the most suitable component keyword based search will not be feasible. With simple keyword based search very large number of comparisons will be required to match the attribute values and also it will suffer from ambiguity problem. In second, genetic based step Weight Vector is optimized. To locate exact match or very near appropriate match, 32 attributes are used as Attribute Vector to describe the component in repository. Genetic Algorithm based step is used, when candidate components are very large in number. Simple key word based search in this type of case will require large number of comparisons. Also optimal combination of various attributes can not be located with simple keyword based search. Genetic algorithm based step will help in selecting the optimal combination of various attribute weights for given context. Optimal combination of weights of various attributes will lead to optimal fitness value.

In this chapter the development of a GA based approach to retrieve reusable components from repository has been explained and demonstrated by considering various situations. Components have been described in repository consisting of two constituents: Textual description has keywords and their synonyms reflecting the functionality of the software component, Genetic description has various attributes related to its reusability and weights showing their relative importance. Retrieval technique first makes use of keyword-based search to get better initial population. Then second step is evolving in nature it improves population with each generation and converges into most appropriate set of components. From final population component with maximum fitness value can be selected. The major strength of this technique is its recall and coverage ratio; it is able to identify the most appropriate component for given context. We demonstrated the results by considering simple, moderate and complex cases of input. Limitation of this technique is that sometimes it may take large number of iterations to converge so takes long time to identify appropriate component but in software reuse precision is more important as compared to time taken to identify component. At present only program code

components are considered in software repository, all other program artifacts can also be considered.

4.2 GAs In Reusable Components Repository

Software development with reuse requires a large repository of reusable components. To encourage software reuse certain practices should be inculcated in the development process. To create a reuse environment we need: A component database called repository for storing quality reusable components along with component descriptions, A retrieval system to retrieve the reusable component for reuse and a tool for integrating selected components for a new context. In present work a repository of about twenty five thousand components is constructed with synthetic data, while storing components in repository two types of descriptions are stored along with the components: keyword-based description and genetic description. Keyword-based description contains component name, keywords about the function performed by the component, their synonyms, return type of the function, number of arguments and type of arguments. Genetic description contains various attributes and their weights, for present explanation thirty-two attributes effecting reusability of components are considered, detailed description of these attributes is given in section 3.5, Table 3.2.

Attribute weight is on the basis of its relative importance in the functionality of the component, its value vary from 0.1 to 0.9. Genetic description is used in evolutionary retrieval. With keyword-based search we try to find all similar components in the repository, these components will be candidate components. Initial population is selected from these candidate components on the basis of fitness value. Fitness value of each chromosome is calculated using equation 1. This initial population will evolve into much better population by applying genetic operators crossover, mutation and selection. Genetic operators make use of relative weights associated with genetic description for optimizing generations. Components with high fitness value will survive longer and components with small fitness value will be eliminated in next evolving generations. Average fitness value of each generation is calculated using equation 2. Average fitness value is the indicator of the similarity of the population with the user query. Difference between the fitness values of two consecutive generations is used to decide termination of

further generations. If the difference between the average fitness of two consecutive generations is less than or equal to 0.01 then further evolutionary generations will stop. The component with maximum fitness value in last population generated will be the identified component to be reused for present user context. Appropriate component description stored in repository plays vital role in retrieving precise component [126, 135].

$$Fitness = \sum_{i=1}^{32} A_i * W_i \quad (1)$$

A_i – Attributes of component's genetic description in repository, it is in binary form either 0 or 1

W_i – Weight assigned to indicate the importance of attribute Ai, its value ranges from 0 to 0.9 any real value

$$AverageFitness = \sum_{i=1}^{popsize} \sum_{j=1}^{32} (A_j * W_j) / popsize \quad (2)$$

A_j, W_j – Component attribute and corresponding weight assigned

popsize – Population size considered for each generation

$$\text{Termination Criteria} = (\text{Average Fitness})_i - (\text{Average Fitness})_{i-1} \leq |0.01| \quad (3)$$

Following genetic operators are used to optimize the initial population.

1. *Selection and Reproduction*: All individuals in the previous generation were made available for reproduction in the next generation. The roulette wheel reproduction process was used to select individuals for reproduction [42, 60].

2. *Crossover*: A single-site crossover is followed. In this cross-site is selected randomly along the length of the mated strings and bits next to the cross-sites are exchanged. Since the knowledge of the appropriate site is not known so it is selected randomly [17, 103].

3. *Mutation*: After crossover, the strings are subjected to mutation. Mutation of a bit involves flipping, changing 0 to 1 and vice versa with small mutation probability P_m. Mutation Probability is the probability of mutation, which is used to calculate number of

bits to be mutated. The mutation operator preserves the diversity among the population, which is also very important for the search [159].

*Number of bits to be changed = Mutation Probability * pop size * string length*

Population size considered here in each generation is 18 and string length is number of attributes considered.

4.3 Experimental Results

A component repository of about twenty five thousand components from various domains is chosen for verification. It is a sample repository with synthetic data. We extracted keywords from each component of the repository on the basis of its functionality; these keywords along with their synonyms and other attributes like return type, number of arguments, type of arguments etc are stored as textual description along with the component in the repository. Thirty-two reusability factors are selected and are assigned relative importance in form of weight; these attributes and their weight are stored as genetic description of the component. On the basis of relative importance a weight in the range of 0.1 to 0.9 is assigned to each attribute. There is an option of selecting these weights automatically also depending on the type of component. We have considered three categories of reusable components as in Boehm's Constructive Cost Model (COCOMO) Organic, Semidetached and Embedded. A component may lie in any of these three categories and accordingly the weights for all attributes are selected relatively. User has always flexibility of assigning weights manually on the basis of relative importance of each attribute. We have considered three cases simple, moderate and complex user queries to demonstrate the working of the system.

Case I: Consider user query in which user wants to find a function that compares two given integer numbers and returns the positive difference, it is a very specific type of query as mentioned below:

```
int     compare(int, int)
```

First keyword-based search considering all synonyms will search for all candidate components. With keyword-based search from centralized repository following components were found:

int	add(int, int)	int	big(int, int)
int	multiply(int, int)	int	smallest(int, int)
int	subtract(int, int)	int	swap(int, int)
int	difference(int, int)	int	compl(int, int)
int	divide(int, int)		
int	largest(int, int)…….		

Similar components with more than two arguments and different return type may also be selected as candidate component. But still it is a case of simple search as total number of candidate components selected is 136. For this number applying genetic algorithm based second step is not feasible. So we can either improve search by using searching within these 136 candidate components or can perform manual selection.

Case II: User needs a functionality that can sum up two numbers, one number is integer and the other is of float type. Result returned by this function should be float. This user query can be described as mentioned below:

float add(int, float)

Keyword 'add' is a very general term; add, sum, total, insert, plus etc are included so in first step keyword-based search selects 2600 total candidate components. For simplicity and on the basis of fitness value we select 18 components and apply evolutionary steps. The entire process of evolution covering three basic operations; crossover, mutation and selection are described through Table 4.1 to Table 4.5, of first iteration.

Table 4.1 shows initial population after applying keyword based search. For simplicity and on the basis of fitness value only 18 components per population are considered.

Table 4.2 shows generation 1 population after reproduction step. Roulette Wheel Selection is used for selection. In initial population, first component C5 has fitness value 7. Its fitness value will be compared with the fitness value of C6 i.e. second component.

As fitness value of C5 is greater than fitness value of C6. So it will replace the fitness value of C6 with the fitness value of C5. Then fitness value of C5 is compared with CC5. Again, it will replace fitness value of CC5 with C5. This process will continue until fitness value of C5 becomes less than AD5 i.e. seventh component. Now fitness value of AD5 will be compared with next component and steps will be repeated as above until it becomes less than next fitness value i.e. of component AF5.

Table 4.3 gives population after crossover step. Crossover operator proceeds in following three steps.

1. Crossover point is selected at random along the string length of weight vector. Here crossover point is selected 23.
2. Now mating is to be done randomly. Select mate for each string. Here second-string component C6 has randomly selected mate AD6, which is 8^{th} component.
3. Now we have two-mate 2^{nd} and 8^{th} string and crossover point is 23. After crossover point 23, all position values of 2^{nd} string replaced with 8^{th} string value.

Table 4.4 is Mutation Table, it shows the bit positions for mutation step. These bit positions are selected randomly.

Table 4.5 gives population after mutation. The next step is to perform mutation on strings in the population.

Number of bits to be changed =Mutate Probability * pop size * string length

Number of bits to be changed=0.02 * 18 * 32 = 12

(Mutation rate is the probability of mutation, which is used to calculate number of bits to be mutated. Mutation rate used is 0.02)

Here values of 12 bits are altered with new value. E.g., 0.9 is considered as 1 bit. First 12 strings are selected and then 1 bit is selected from each of these strings to be mutated. In first string, at bit position 11, old number is 0.9 and it is replaced with new number 0.4. The bit position and new number is selected randomly.

Table 4.1 Initial Population after Keyword Search

No	Comp Name:	AV	WV	Fitness Value
1	c5	00001010001111110101010101100001	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7
2	c6	01110110000000000000010010000010	0.80.20.10.10.40.90.80.10.50.50.20.50.60.80.10.50.80.20.20.40.10.80.20.20.50.10.50.20.10.50.10.8	3.5
3	cc5	00000010111001000000001011110010	0.60.20.10.10.60.60.60.10.10.10.50.10.90.60.10.10.60.60.20.60.10.60.70.90.10.80.60.20.10.10.10.6	4.4
4	cc6	00010101000110001011100100011100	0.10.20.30.40.10.10.10.30.50.30.60.50.40.10.30.20.10.10.20.10.30.10.80.40.40.30.20.20.40.20.30.1	3.6
5	a5	00111000001001001100101100011101	0.50.30.30.30.70.50.50.30.10.40.40.40.40.50.30.40.50.40.30.80.30.50.30.40.40.30.40.40.30.40.30.4	5.6
6	a6	00001010001111110101010101100001	0.60.40.20.20.90.10.60.20.40.40.40.60.40.60.20.40.90.50.40.20.20.60.40.40.40.20.40.40.20.40.20.6	7
7	ad5	11110010011001110100101010000010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
8	ad6	10000001100111011111101010000001	0.80.20.10.10.40.90.80.10.50.50.20.50.60.80.10.50.80.20.20.40.10.80.20.20.50.10.50.20.10.50.10.8	7
9	add5	00111101111000000000010110111001	0.30.40.90.70.10.80.30.90.50.50.80.10.10.30.90.10.30.80.40.10.90.30.50.60.10.90.10.40.90.10.70.3	7.9
10	add6	01000010010010100100101111100000	0.20.50.70.70.20.20.20.70.70.40.20.70.40.20.70.40.20.20.50.20.70.20.60.40.40.70.40.50.70.40.70.2	5.6
11	aa5	10001001111100011100101010111000	0.30.10.50.50.70.60.30.50.30.30.30.30.40.30.50.30.30.30.10.70.50.30.10.30.30.50.30.10.50.30.50.3	5.4
12	aa6	10011011100010101011010001000101	0.20.20.30.30.20.20.20.30.10.10.10.10.40.20.30.10.20.10.20.20.30.20.20.10.10.30.10.20.30.10.30.2	3.4
13	as5	11101010110110000010100001110010	0.20.50.70.70.20.20.20.70.70.40.20.70.40.20.70.40.20.20.50.20.70.20.60.40.40.70.40.50.70.40.70.2	7.5
14	as6	11010001011001111111110111001111	0.50.10.10.10.50.50.50.10.20.40.50.10.10.50.10.10.50.50.10.50.10.50.10.10.10.10.10.10.10.10.10.5	5.7
15	aaa5	10010110011000011000010010010011	0.60.20.10.10.60.60.60.10.10.10.50.10.90.60.10.10.60.60.20.60.10.60.70.90.10.80.60.20.10.10.10.6	4.8
16	aaa6	01111001010101101111100101101100	0.10.20.30.40.10.10.10.30.50.30.60.50.40.10.30.20.10.10.20.10.30.10.80.40.40.30.20.20.40.20.30.1	4.8
17	af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1
18	af6	10011011100010101011010001000101	0.30.30.30.30.30.10.20.30.50.60.30.50.30.30.30.50.30.10.30.30.30.30.30.50.30.50.30.30.50.30.3	4.8

Average Fitness::5.89

Generation: 1-------->>

Table 4.2 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7
c6	01110110000000000000010010000010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7
cc5	00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7
cc6	00010101000110001011100100011100	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7
a6	00001010001111110101010101100001	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7
ad5	11110010011001110100101010000010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
aa5	10001001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
as5	11101010110110000010100001110010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1

Table 4.3 Population after Crossover

Crossover Point:: 23

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7	7 1 5
c6	01110110000000000000010010000010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	7	8.9 2 8
cc5	00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	7	8.9 3 16
cc6	00010101000110001011100100011100	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7	7 4 6
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	7	8.9 5 11
a6	00001010001111110101010101100001	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	7	9.1 6 17
ad5	11110010011001110100101010000010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9	8.9 7 10
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9	8.9 8 15
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.40.20.20.20.90.20.20.20.4	8.9	7 9 3
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9	8.9 10 15
aa5	10001001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.40.20.20.20.90.20.20.20.4	8.9	7 11 4
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.40.20.20.20.90.20.20.20.4	8.9	7 12 5
as5	11101010110110000010100001110010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9	8.9 13 10
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9	8.9 14 10
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.40.20.20.20.90.20.20.20.4	8.9	7 15 2
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9	8.9 16 16
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1	8.9 17 7
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1	9.1 18 18

Table 4.4 Mutation Table

Bit Position	Old Number	New Number
11	0.9	0.4
6	0.9	0.3
15	0.2	0.1
24	0.4	0.3
25	0.6	0.6
2	0.8	0.9
31	0.6	0.4
25	0.6	0.5
15	0.6	0.7
27	0.7	0.2
13	0.5	0.7
12	0.7	0.6

Table 4.5 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.80.20.20.90.90.40.20.20.20.40.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	6.5
c6	01110110000000000000010010000010	0.40.80.20.20.90.30.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	3.5
cc5	00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.10.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	6.3
cc6	00010101000110001011100100011100	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.30.20.20.20.90.20.20.20.4	5.8
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
a6	00001010001111110101010101100001	0.40.90.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
ad5	11110010011001110100101010000010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	8.7
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.50.60.70.90.60.70.60.4	8.2
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.70.70.40.10.90.40.60.40.90.40.20.20.20.90.20.20.20.4	7.7
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.20.90.60.70.60.4	6.6
aa5	10001001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.70.70.60.70.40.10.90.40.60.40.90.40.20.20.20.90.20.20.20.4	8.2
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.60.50.70.60.70.40.10.90.40.60.40.90.40.20.20.20.90.20.20.20.4	6.8
as5	11101010110110000010100001110010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	9.3
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.40.20.20.20.90.20.20.20.4	6.4
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	11.3
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.7

Average Fitness::7.81

==========

Best Component	Max Fitness
as6	12.5
aaa6	11.3
as5	9.3

Average Fitness: 7.81

****

It took 28 iterations to converge and find most appropriate match. All other iteration tables showing various generations are given in Appendix A. Let us consider technical assessment criteria for this query

Precision = Relevant Retrieved Assets / No. of Retrieved Assets

= 28 iterations × 18 Components / 2600 = 0.19

Recall = Relevant Retrieved Assets /Total Relevant Assets in Repository = 1, as no relevant component is missed.

Coverage Ratio= Average No. of Visited Assets / Total Size = 2600/25000=0.104

0.19 precision value shows that initially keyword search selects too many components but out of those components algorithm successfully selected appropriate component. Coverage ratio of 0.104 indicates that only about 10% components need to be visited to locate component with recall of 1.

Case III: User enters a query to locate following get( ) function

float get(int, float)

Keyword 'get' is ambiguous in nature, so keyword based search in first step will include all feasible candidate components. It selects 4250 components as candidate components. It is an example of a complex case for searching. Initial population selected on the basis of keyword based match. Genetic algorithm based operations reproduction, crossover and mutation on initial population in iteration I are shown in Tables 4.6 to Table 4.10. Final population is shown in Table 4.11. All genetic operators will work in the same way as explained in case II.

Table 4.6 Initial Population after Keyword Search

No	Comp Name:	AV	WV	Fitness Value
1	gg12	10000001100111011111101010000001	0.30.60.40.40.60.30.30.40.40.20.40.40.30.30.40.40.30.60.60.60.40.30.60.30.40.40.40.60.40.40.40.3	6.3
2	gg13	01111001010101101111100101101100	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7.7
3	g12	00100101100010000011001100001010	0.10.10.20.20.40.30.10.20.90.10.30.90.10.10.20.10.10.30.10.40.20.10.10.10.10.20.10.10.20.40.20.1	2.8
4	g13	00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	4.6
5	ge12	01110110000000000000010010000010	0.50.10.20.20.80.80.50.20.90.90.50.90.50.50.30.90.50.50.10.40.20.50.10.50.90.20.90.10.20.90.20.5	3.4
6	ge13	11010001011001111111110111001111	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
7	gt12	00111101111000000000010110111001	0.80.20.10.10.40.90.80.10.50.50.20.50.60.80.10.50.80.20.20.40.10.80.20.20.50.10.50.20.10.50.10.8	5.9
8	gt13	01000010010010100100101111100000	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	4.9
9	ggg12	10010110011000011000010010010011	0.80.20.10.10.40.90.80.10.50.50.20.50.60.80.10.50.80.20.20.40.10.80.20.20.50.10.50.20.10.50.10.8	7
10	ggg13	01111001010101101111100101101100	0.20.60.50.50.50.40.20.50.20.20.50.20.40.20.50.20.20.50.60.50.50.20.60.40.20.50.20.60.50.20.50.2	7.8
11	gh12	10001101110100000010001110110111	0.10.20.30.40.10.10.10.30.50.30.60.50.40.10.30.20.10.10.20.10.30.10.80.40.40.30.20.20.40.20.30.1	4.7
12	gh13	10100000001101111111010001000010	0.50.30.30.30.70.50.50.30.10.40.40.40.40.50.30.40.50.40.30.80.30.50.30.40.40.30.40.40.30.40.30.4	5.9
13	gge12	00111000001001001100101100011101	0.50.10.50.50.50.50.50.50.20.40.50.40.50.50.50.40.50.50.10.50.50.50.10.50.40.50.40.10.50.40.50.6	6.2
14	gge13	00001010001111110101010101100001	0.20.10.40.40.20.20.20.40.90.60.80.90.80.20.40.60.20.20.10.20.40.20.10.80.60.40.10.10.40.60.40.2	6.2
15	gtt12	10010110011000011000010010010011	0.20.60.50.50.50.40.20.50.20.20.50.20.40.20.50.20.20.50.60.50.50.20.60.40.20.50.20.60.50.20.50.2	4.1
16	gtt13	01100111110010011101110111010110	0.30.60.40.40.60.30.30.40.40.20.40.40.30.30.40.40.30.60.60.60.40.30.60.30.40.40.40.60.40.40.40.3	8
17	ggt12	00011001001010000000100000111001	0.20.70.60.60.20.60.20.60.40.40.90.40.90.20.60.40.20.20.70.20.60.20.70.90.40.60.40.70.10.40.60.2	5.2
18	ggt13	11011011111101011111011110000100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.2

Average Fitness::6.17

Generation: 1-------->>

Table 4.7 Population after Reproduction

No	Comp Name:	AV	WV	Fitness Value
gg12		10000001100111011111101010000001	0.30.60.40.40.60.30.30.40.40.20.40.40.30.30.40.40.30.60.60.60.40.30.60.30.40.40.40.60.40.40.40.3	6.3
gg13		01111001010101101111100101101100	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7.7
g12		00100101100010000011001100001010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7.7
g13		00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7.7
ge12		01110110000000000000010010000010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7.7
ge13		11010001011001111111110111001111	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
gt12		00111101111000000000010110111001	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
gt13		01000010010010100100101111100000	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
ggg12		10010110011000011000010010010011	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
ggg13		01111001010101101111100101101100	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
gh12		10001101110100000010001110110111	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
gh13		10100000001101111111010001000010	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
gge12		00111000001001001100101100011101	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
gge13		00001010001111110101010101100001	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
gtt12		10010110011000011000010010010011	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
gtt13		01100111110010011101110111010110	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
ggt12		00011001001010000000100000111001	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1
ggt13		11011011111101011111011110000100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.2

Table 4.8 Population after Crossover

Crossover Point:: 8

No	Comp Name:	AV	WV	Fitness Value
gg12		10000001100111011111101010000001	0.30.60.40.40.60.30.30.40.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	6.3 8.1 1 12
gg13		01111001010101101111100101101100	0.40.80.20.20.90.90.40.20.40.20.40.40.30.30.40.40.30.60.60.60.40.30.60.30.40.40.40.60.40.40.40.3	7.7 6.3 2 1
g12		00100101100010000011001100001010	0.40.80.20.20.90.90.40.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	7.7 8.1 3 15
g13		00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	7.7 8.1 4 13
ge12		01110110000000000000010010000010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	7.7 7.7 5 5
ge13		11010001011001111111110111001111	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 6 17
gt12		00111101111000000000010110111001	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 7 13
gt13		01000010010010100100101111100000	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 8 10
ggg12		10010110011000011000010010010011	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 9 17
ggg13		01111001010101101111100101101100	0.20.10.20.20.80.70.20.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	8.1 7.7 10 4
gh12		10001101110100000010001110110111	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 11 17
gh13		10100000001101111111010001000010	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 12 7
gge12		00111000001001001100101100011101	0.20.10.20.20.80.70.20.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	8.1 7.7 13 5
gge13		00001010001111110101010101100001	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 14 9
gtt12		10010110011000011000010010010011	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 15 16
gtt13		01100111110010011101110111010110	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 16 10
ggt12		00011001001010000000100000111001	0.20.10.20.20.80.70.20.20.50.50.70.50.70. 90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	8.1 8.1 17 15
ggt13		11011011111101011111011110000100	0.10.90.60.60.40.60.40.60.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	12.2 8.1 18 8

Table 4.9 Mutation Table

Bit Position	Old Number	New Number
1	0.3	0.8
19	0.6	0.3
22	0.2	0.9
21	0.2	0.2
12	0.2	0.3
27	0.1	0.8
5	0.8	0.1
2	0.1	0.2
17	0.2	0.3
8	0.2	0.2
1	0.2	0.3
13	0.7	0.7

Table 4.10 Population after Mutation

No	Comp Name:	AV	WV	Fitness Value
gg12		10000001100111011111101010000001	0.80.60.40.40.60.30.30.40.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	7
gg13		01111001010101101111100101101100	0.40.80.20.20.90.90.40.20.40.20.40.40.30.30.40.40.30.60.30.60.40.30.60.30.40.40.40.60.40.40.40.3	7.7
g12		00100101100010000011001100001010	0.40.80.20.20.90.90.40.20.50.50.70.50.70.90.20.50.20.70.10.70.20.90.10.70.50.20.10.10.20.50.20.2	4.5
g13		00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	4.2
ge12		01110110000000000000010010000010	0.40.80.20.20.90.90.40.20.20.20.90.30.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	3.3
ge13		11010001011001111111110111001111	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.80.10.20.50.20.2	8.1
gt12		00111101111000000000010110111001	0.20.10.20.20.10.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	5.1
gt13		01000010010010100100101111100000	0.20.20.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	4.3
ggg12		10010110011000011000010010010011	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.30.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	4.5
ggg13		01111001010101101111100101101100	0.20.10.20.20.80.70.20.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	6.9
gh12		10001101110100000010001110110111	0.30.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	6
gh13		10100000001101111111010001000010	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	5.5
gge12		00111000001001001100101100011101	0.20.10.20.20.80.70.20.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	6.9
gge13		00001010001111110101010101100001	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	7.3
gtt12		10010110011000011000010010010011	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	4.4
gtt13		01100111110010011101110111010110	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	7.8
ggt12		00011001001010000000100000111001	0.20.10.20.20.80.70.20.20.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	3.4
ggt13		11011011111101011111011110000100	0.10.90.60.60.40.60.40.60.50.50.70.50.70.90.20.50.20.70.10.70.20.20.10.70.50.20.10.10.20.50.20.2	10.3

Average Fitness::5.96

==========

Best Component	Max Fitness
ggt13	10.3
ge13	8.1
gtt13	7.8

Average Fitness: 5.96

****

Generation: 42 -------->>

Table 4.11 Final Iteration

Comp Name	AV	WV	New Fitness
gg12	10000001100111011111101010000001	0.50.50.40.10.20.90.40.80.80.50.70.50.10.90.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	8.2
gg13	01111001010101101111100101101100	0.40.30.40.20.20.90.40.80.80.50.40.50.90.90.90.30.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	10.4
g12	00100101100010000011001100001010	0.40.30.40.20.20.90.40.80.80.50.40.50.90.90.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.60.90.7	7
g13	00000010111001000000001011110010	0.40.30.40.20.20.90.40.80.80.50.10.50.90.90.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	7
ge12	01110110000000000000010010000010	0.40.30.40.20.20.90.40.80.80.50.40.50.90.90.90.80.40.70.10.20.30.90.50.70.30.90.80.90.80.20.90.7	4.3
ge13	11010001011001111111110111001111	0.40.30.80.20.50.90.40.40.70.30.70.50.90.90.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	12.6
gt12	00111101111000000000010110111001	0.40.30.80.20.70.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.50.90.80.90.80.20.90.7	9.7
gt13	01000010010010100100101111100000	0.40.30.80.20.50.90.40.40.70.30.40.50.90.30.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	7.6
ggg12	10010110011000011000010010010011	0.40.30.80.20.50.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.50.90.80.90.20.20.90.7	7.7
ggg13	01111001010101101111100101101100	0.40.30.80.20.50.90.40.40.70.30.40.50.90.90.90.40.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	10.5
gh12	10001101110100000010001110110111	0.40.30.80.20.50.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.50.90.80.90.80.20.90.7	9
gh13	10100000001101111111010001000010	0.40.40.80.20.50.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	8.8
gge12	00111000001001001100101100011101	0.40.30.80.20.50.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.50.90.80.90.80.20.90.7	8.6
gge13	00001010001111110101010101100001	0.40.30.80.20.50.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	10.2
gtt12	10010110011000011000010010010011	0.40.30.80.20.50.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	7.5
gtt13	01100111110010011101110111010110	0.40.30.80.20.50.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	12.5
ggt12	00011001001010000000100000111001	0.40.30.80.20.50.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.50.90.80.90.80.20.90.7	6.5
ggt13	11011011111101011111011110000100	0.40.30.80.20.50.90.40.40.70.30.40.50.90.90.90.80.40.70.10.20.90.90.50.70.30.90.80.90.80.20.90.7	9.8

Average Fitness::8.77

==========

Best Component	Max Fitness
ge13	12.6
gtt13	12.5
ggg13	10.5

Average Fitness: 8.77

****

This query took 42 iterations to converge. Now consider technical assessment criteria for this query

Precision = (42 × 18) / 4250 = 0.178

Recall = 1, No relevant component is forgotten while matching

Coverage Ratio = 4250/25000 = 0.17

It shows only 17% components need to be visited in matching this query.

Average fitness of both the queries for all generations of Case II and Case III are given in Table 4.12 and the same is plotted in figure 4.1

Table 4.12 Average Fitness

Generation Number	CASE II	CASE III	Generation Number	CASE II	CASE III
0	5.89	6.17	**22**	8.57	7.89
1	7.81	5.96	**23**	8.62	7.83
2	8.05	6.29	**24**	8.68	7.92
3	8.08	6.44	**25**	8.73	8.04
4	8.11	7.47	**26**	8.59	8.11
5	8.16	7.54	**27**	8.62	8.02
6	8.41	7.52	**28**	8.66	8.15
7	8.26	7.84	**29**	8.66	8.37
8	8.13	7.87	**30**		8.28
9	8.29	8.11	**31**		8.21
10	8.2	7.94	**32**		8.24
11	8.22	7.97	**33**		8.2
12	8.34	7.79	**34**		8.22
13	8.56	7.66	**35**		8.07
14	8.74	7.78	**36**		8.09
15	8.67	7.83	**37**		8.37
16	8.59	8.12	**38**		8.22
17	8.53	8.17	**39**		8.44
18	8.34	8.19	**40**		8.51
19	8.51	8.15	**41**		8.77
20	8.64	8.06	**42**		8.77
21	8.73	8.02			

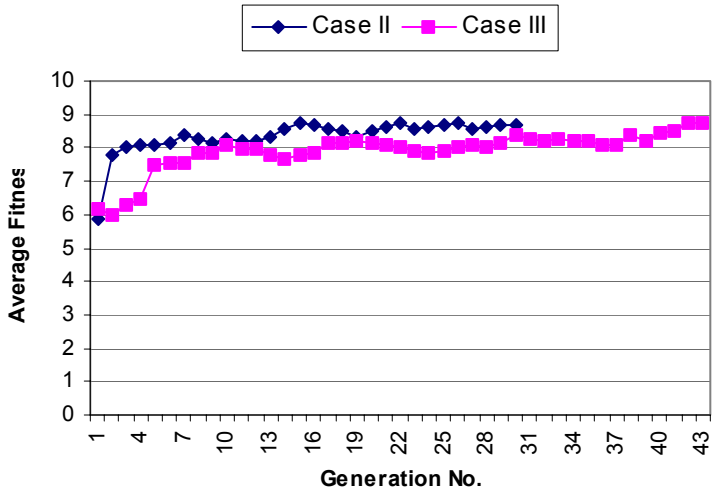


Figure 4.1 Comparative Average Fitness For Moderate and Complex queries

4.4 Comparative Analysis

The results reported by Mili et. al.[97] Mareek et. al.[167] regarding technical assessment of software repositories are on the basis of technical, managerial and human factors using various parameters. Here only three parameters from technical assessment are considered for comparative analysis. Precision, Recall of VH-Very High means 1, most suitable, then H-High, M-Medium and L-Low, 0 is lowest. In case of Coverage Ratio, it is reverse VH=1 means exhaustive coverage and VL-Very Low means selective coverage. U-Unknown, means this is not relevant.

Table 4.13 Technical Assesment

Method	Precision	Recall	Coverage Ratio
Information Retrieval Methods	M	H	L
Descriptive Methods	H	H	VH
Operational Semantics Methods	VH	H	H
Denotational Semantics Methods	VH	H	H
Topological Methods	U	U	VH
Structural Methods	VH	VH	VH
Proposed Genetic Algorithm Based Method	**L**	**VH**	**VL**

In proposed method Recall VH(1) is achieved with Coverage Ratio of VL(0.10) is an achievement as compared to other existing methods. The precision L(0.18 appox.) is lower side due to initial population. The improvement in initial population will result in better precision. With proposed method to locate any component from repository, on average 10% components need to be visited without missing any relevant component, Recall 1.

4.5 Conclusions

In this chapter, a novel way of storing the component descriptions in the software repository has been described for evolutionary type of retrieval. This work is the first

attempt to describe reusable components in this way and then applying genetic algorithms to the challenging problem of identifying precise reusable component for the required context. Our work make use of attributes and their weights to identify appropriate component evolutionarily. The major contributions of our work are; first, as shown in experimental results it has been able to identify appropriate components even if we start with a much wider population. Secondly, the results produced show-encouraging trends and are similar as in for document retrieval, but in this work genetic algorithms are used for more complex task of identifying software component. In future component description by including design, testing and other software artifacts related terminology can be enhanced to produce better results. Also user experience related terms in form of collaborative tagging would also be useful in identifying precise components. The results produced here are better than simple keyword based search.

Chapter 5

Ant Colony Based Reusable Software Components Retrieval

5.1 Introduction

We usually see natural ants creating paths from food to home; these paths are usually shortest possible paths from the food source to their home. Ants leave a chemical substance pheromone, on the path they traverse. Following ants to select the appropriate path use this pheromone, more number of ants traverse the same path, more amount of pheromone will get deposited and chances of selecting this path by the following ants will increase. Ant colony optimization algorithms are based on the idea that "Each path followed by an ant is associated with a candidate solution for a given problem. When an ant follows a path, the amount of pheromone deposited on that path is proportional to the quality of the corresponding candidate solution for the target problem. When an ant has to chose two or more paths, the paths with a larger amount of pheromone have a greater probability of being chosen by the ant" [18, 24, 69, 90]. As a result, the ants eventually converge to a short path, hopefully optimum or a near optimum solution for the target problem. Similarly, retrieval of appropriate reusable software component for a given context is also an optimization problem [44, 45, 78, 81].

Motivation to Use Ant Colony Optimization

In ant colony optimization, while ants move in search of food they communicate with each other by sensing the pheromone deposited on the path followed by leading ants. By following each other they reach the final destination by traversing the shortest path in general. Ant colony optimization has been used earlier in web pages clustering, heterogeneous data clustering and population clustering. In current work it has been proposed to use ant colony optimization by using attribute interactions to classify reusable components. Attribute interactions means how attributes of one component are related with other attributes of components. These attribute interactions are very close to semantic relation of these attributes, so it results in appropriate type of matching.

Software repositories consist of large number of components of various domains and are very huge in size. Software reuse consists of two types of tasks, development for reuse and development with reuse. Reuse involves four steps in general selection, analysis, customization and instantiation [14, 92]. Component classification is very vital step and it is very difficult to identify appropriate component from repository for given context. Because the search space is very large, an ant colony optimization based component classification technique is proposed in this work.

In this task major goal is to generate rules for classifying components. For a particular context, various rules are generated and then their quality is checked. Various terms in form of ants are selected for a given context, and then their interrelationships are derived [21, 58, 80]. To generate rules deposited pheromone on various terms is checked. Pheromone is deposited on a particular path if terms on that path and their relationships are contributing for current context otherwise pheromone is evaporated and these terms will not play role in rule generation. Quality of a rule is calculated to decide the convergence of a rule. If quality of two consecutive rules is same or nearly same it means it converged and is included in the list of various rules applicable to present context.

In proposed work 150 terms i.e. ants, their attributes, Characteristics means attribute values and their initial pheromone is considered. These 150 ants may represent a large software repository cluster, as one term may represent similar components or more terms may be required to represent single component. One database to represent various domains of components is created. In this database representative terms of the domain are stored. 10 ant-based terms are considered to form a rule. We have considered “Data Structure” as a representative class. It took three iterations to generate one rule. A set of rules can be generated to satisfy one query and a particular path means a component is selected on the basis of pheromone deposited. The major goal is to discover a good set of classification rules to classify reusable software component from repository for given context. To the best our knowledge, the use of Ant Colony algorithm for discovering classification rules for retrieving reusable software components from repository, is a research area still unexplored. Actually ant algorithms developed for data mining and clustering are different from the classification task addressed in our work [60, 67]. A

component clustering and rule generation technique has been proposed in this work for component repository.

We need to know and compare attributes indicating the semantic meaning of any component with given user context to return most appropriate component. A component repository containing about one lakh fifty thousand synthetic reusable software components have been used for taking results of current algorithm. To represent one cluster of components a database of 150 terms like linked list, tree, update etc, is created, Attributes of these terms are considered and their values are assigned. Initially pheromone value of one is assigned to each term. In this repository components belonging to various 22 domains like banking, systems, hospital, finance etc are stored. To represent each domain, a set of attributes on the basis of their functionality is created. First user enters a query; from his query keywords are identified. Various terms to represent these keywords and then their relations are identified. These terms are compared with database of 150 terms and 10 best terms from the available on the basis of similarity match for given context are selected. These 10 terms are compared with the domain attributes for the said context. On the basis of these comparisons, pheromone for each term is calculated. Then quality of these ten terms to represent the current context is calculated. Pheromone for each term will get updated. In next iteration only terms with more amount of pheromone will get selected and irrelevant terms means terms with less amount of pheromone will be ignored. These iterations are repeated till we get the rule with desired quality or the quality of two consecutive rules is same. This rule will be stored in domain rules database and then next rule will be generated for the same domain.

5.2 Components Clustering

Components need to be arranged in repository in some particular fashion to facilitate efficient search [34, 40]. Components clustering mean grouping them with respect to some characteristics. Components or the terms that are used to represent the components should be stored in some order. This will later help user to locate particular component or a cluster of components. Here in our implementation component interactions are used for clustering. Component interaction is the relation between two components, this relation can be common attribute, list of attributes or domain to which these components belong

to. Each component is represented by key feature means attribute or list of attributes. These attribute terms can be short listed by using tool like WordNet. WordNet is a semantic lexicon for the English language [32]. It groups English words into sets of synonyms called synsets, provides short, general definitions, and records the various semantic relations between these synonym sets. WordNet was created and is being maintained at the Cognitive Science Laboratory of Princeton University under the direction of psychology Professor George A. Miller. 2006 version is used for present implementation, which contains about 150,000 words organized in over 115,000 synsets for a total of 207,000 word-sense pairs; in compressed form, it is about 12 megabytes in size. This short-listing of attributes help to represent components with least number or unique attribute. These attributes uniquely identify the components and are indicators of functionality of components. Same type of attribute identification is applied to query at the time of component search to resolve the ambiguity. This will help to direct the search on only selected clusters. So component clustering helps to reduce the search space and results in reduced search time [6, 9, 49, 50, 51].

A clustering method is proposed in this work on the basis of common attribute(s). Components are clustered on the basis of component interactions. It is assumed that there are N components with M attributes to represent these components, each component has at least one attribute, always attempt should be made to keep less number of attributes to represent more number of components (M>N). This restriction of less number of attributes is to define more and more number of components uniquely and less number of attributes needs to be searched to locate components. All components are scattered on the Cartesian plain randomly for clustering around the origin. Each component has one corresponding ant. Each ant will be allowed to interact with all other ants. These ants will determine interactions between components. First they will try to find common attributes i.e. for their each attribute, is there any attribute that matches with some attribute of any component. If two ants find that they have at least one attribute matching with each other. It means they have something in common; they will be placed in one cluster. By finding component interactions, totally different components will be scattered away [95, 163, 164]. These components will be compared with the query first at the time of search. Here clustering is putting the similar components in one domain. But one attribute may belong

to more than one domain. There may be situation that an attribute may represent more than one component and a component is defined by a list of attributes. Components are arranged with attributes in a hierarchical relation as shown in figure 5.1.

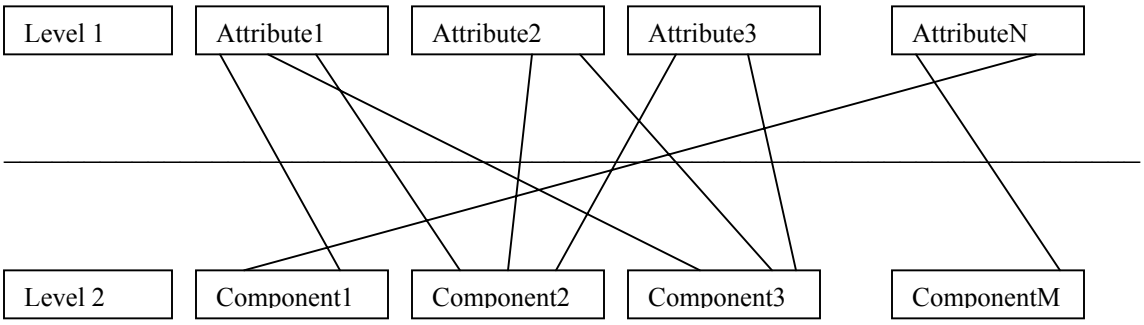


Figure 5.1 Arrangement of Components and Attributes

To avoid redundancy minimum number or unique attributes are used to define a component. The repeating attributes are deleted and the original unique attribute is linked to the component e.g. consider Attribute2 and Attribute3 have common meaning, Attribute2 represents component2, Attribute3 represents component3, then attribute3 is deleted and attribute2 now will be used to represent component2 and component3. Few component clusters on the basis of attributes are formed as shown in figure 5.2 to figure 5.6.

CASE I: Account from various domains

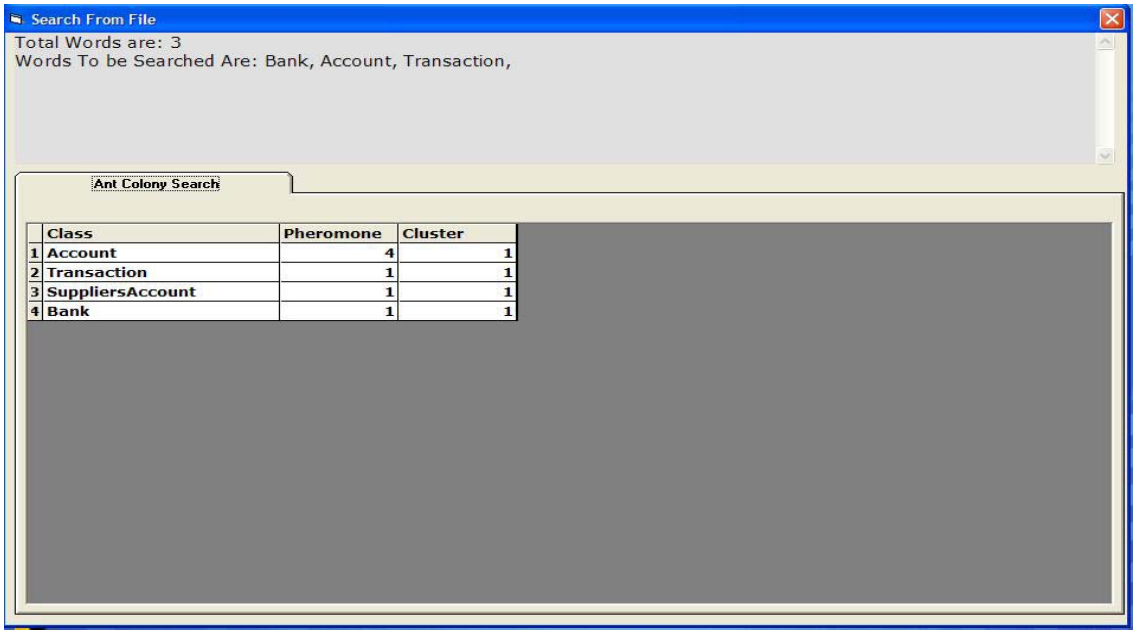


Figure 5.2: Simple domain search case

Here it is seen that all the given three terms belong to cluster number 1. Term Account belongs to class Account with pheromone weight of 4. All other terms belong to various other classes of cluster 1. So the probability of component class Account becomes more. For queries pertaining to only term Account will be searched in only cluster 1.

For our Ant colony based component repository Recall value will be 1, As the total number of relevant assets in the repository will be in form of total clusters to be visited and in each case these clusters are to be visited at least once if any query term belongs to this cluster. So total number of relevant assets in repository and total number of retrieved assets for current query will be same. It means no relevant components are forgotten. But for each query Precision and Coverage Ratio (CR) will differ depending on number of clusters to be visited.

Precision = Relevant Retrieved Assets / Total Number of Retrieved Assets

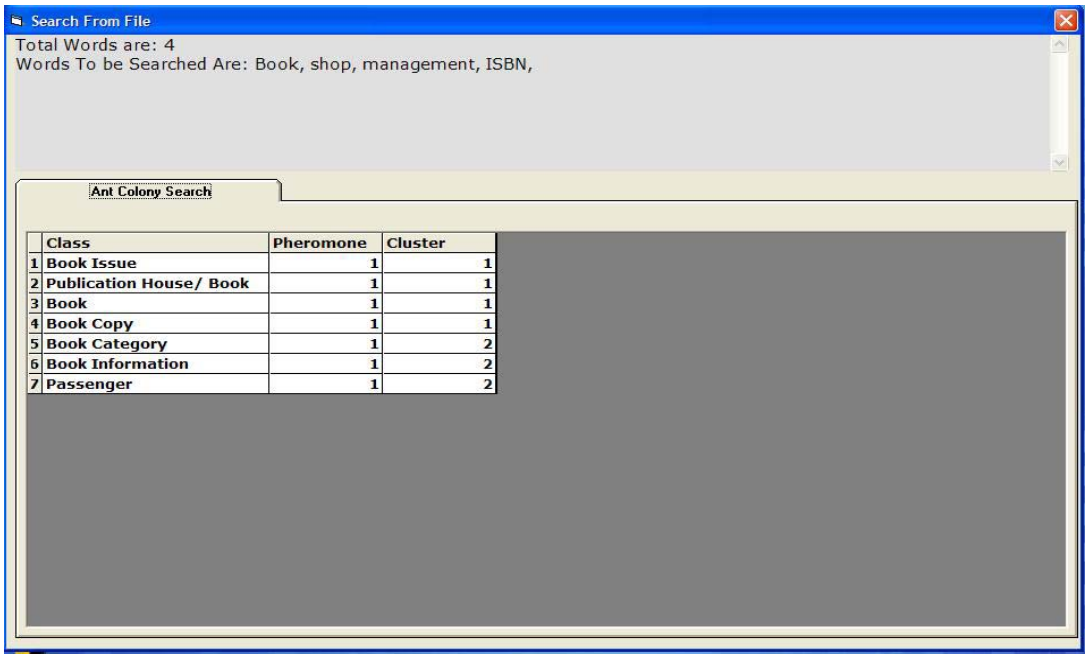
As all terms belong to same cluster 1 and search will also suggest 10 terms or less from the same cluster.

Precision = 1 (ClusterNumber 1) × / Number of Clusters to be visited =1/1=1

$$CR = \frac{\text{No. of Clusters to be visited} \times \text{Components in each cluster} \times \text{Total Terms}}{\text{Total Number of Components in Repository}}$$

CR = (1×100×150)/150000 = 0.1

CASE II: Book



	Class	Pheromone	Cluster
1	Book Issue	1	1
2	Publication House/ Book	1	1
3	Book	1	1
4	Book Copy	1	1
5	Book Category	1	2
6	Book Information	1	2
7	Passenger	1	2

Figure 5.3: Search case for Book cluster single keyword in various domains

In this example case Book Issue, Publication House, Book and Book Copy are selected from cluster 1 and Book Category, Book Info and Passenger are selected from cluster 2. So for searching any query containing these types of terms these two clusters need to be searched.

Precision = 1 (ClusterNumber 1 OR 2)×/2(Number of Clusters to be visited) =1/2=0.5

$$CR = \frac{\text{No. of Clusters to be visited} \times \text{Components in each cluster} \times \text{Total Terms}}{\text{Total Number of Components in Repository}}$$

CR = (2×100×150)/150000 = 0.2

CASE III: Hospital

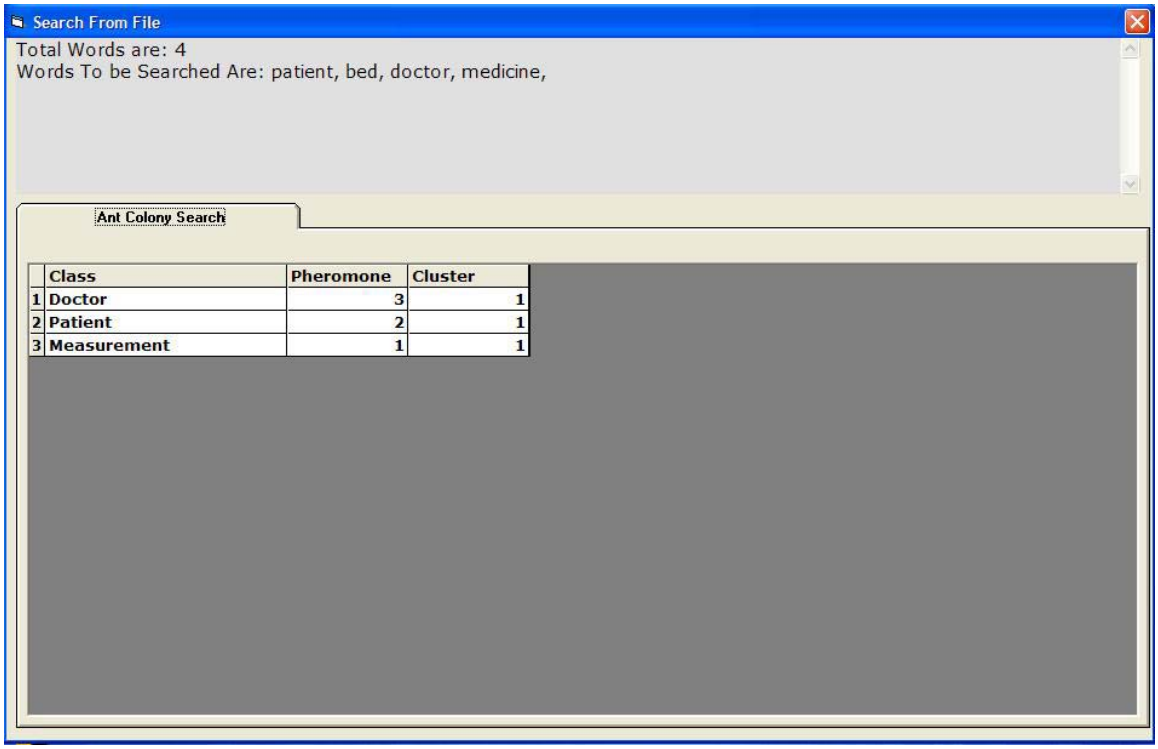


Figure 5.4: Search case for single cluster same domain

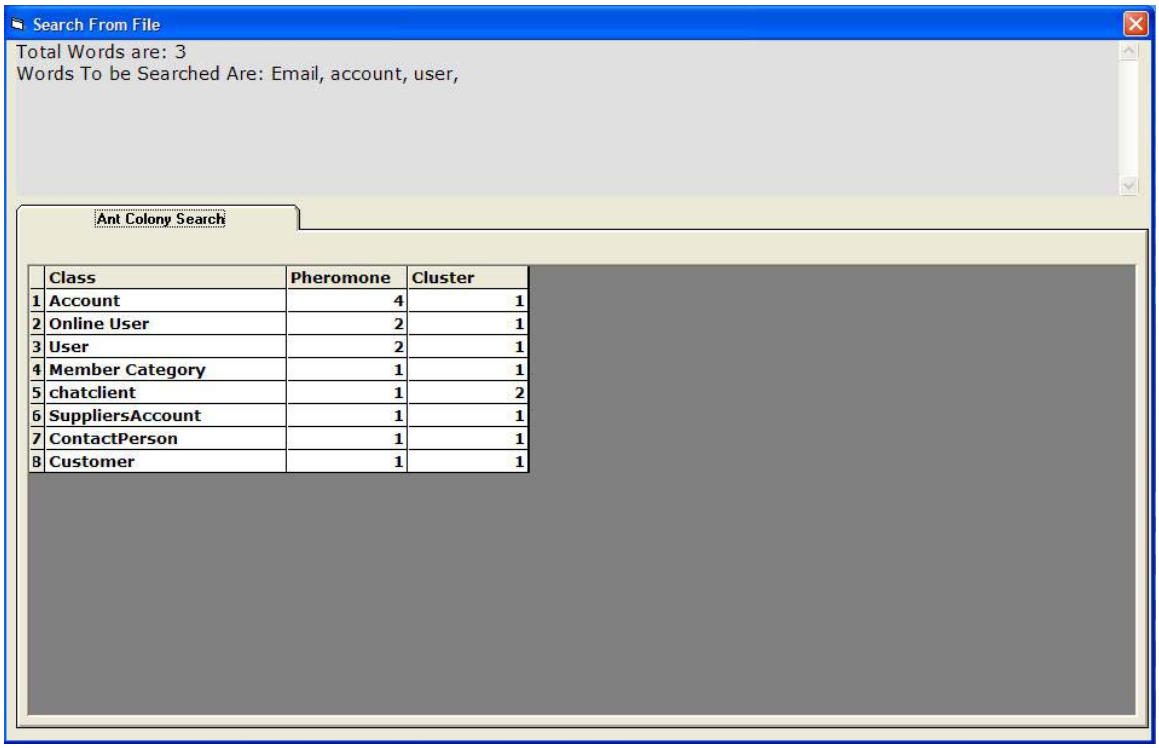
Here all terms belong to only one cluster, cluster 1.

Precision = 1 (ClusterNumber 1) × / Number of Clusters to be visited =1/1=1

$$CR = \frac{\text{No. of Clusters to be visited} \times \text{Components in each cluster} \times \text{Total Terms}}{\text{Total Number of Components in Repository}}$$

CR = (1×100×150)/150000 = 0.1

CASE IV: email: In this case all three terms Email, Account and user belong to two clusters Cluster 1 and Cluster 2. Account component class with maximum pheromone of 4 from cluster 1 has higher probability.



	Class	Pheromone	Cluster
1	Account	4	1
2	Online User	2	1
3	User	2	1
4	Member Category	1	1
5	chatclient	1	2
6	SuppliersAccount	1	1
7	ContactPerson	1	1
8	Customer	1	1

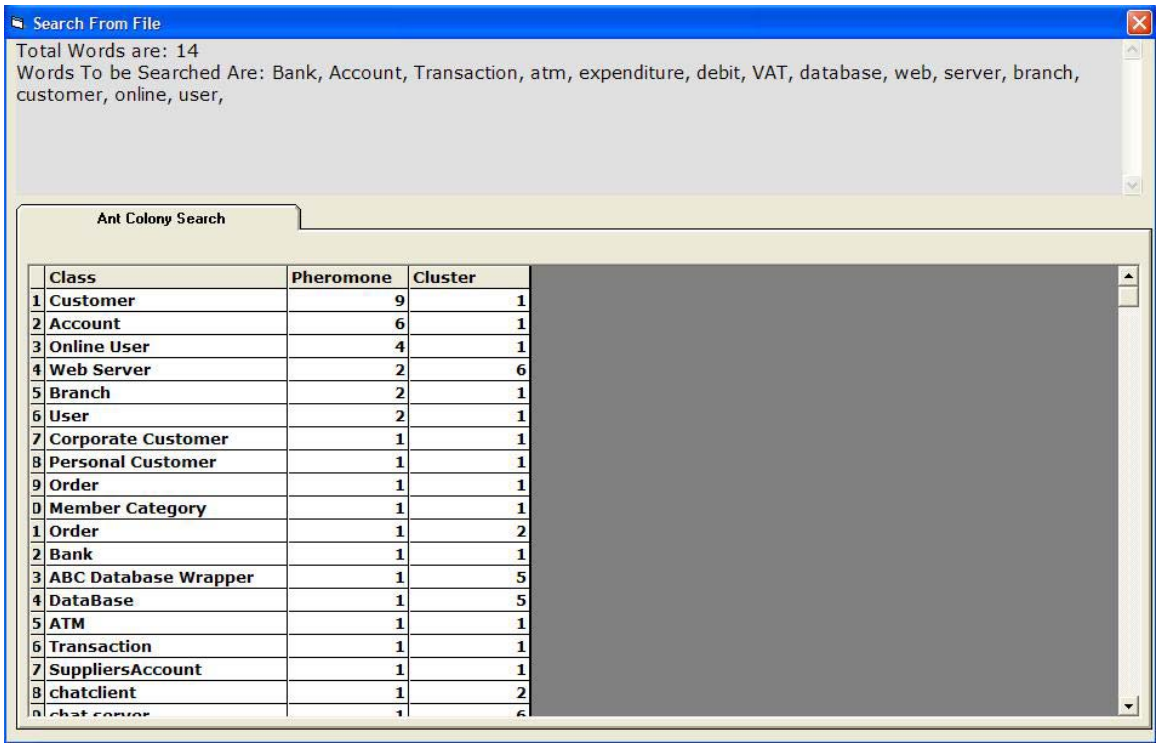
Figure 5.5: Search case for email from various domains

Precision = 1 (ClusterNumber 1 OR 2) × / 2 (Number of Clusters to be visited) =1/2=0.5

$$CR = \frac{\text{No. of Clusters to be visited} \times \text{Components in each cluster} \times \text{Total Terms}}{\text{Total Number of Components in Repository}}$$

$CR = (2 \times 100 \times 150)/150000 = 0.2$

CASE V: Complex Mixed Domain: In this example case various terms from different domains are given. These terms are found to be from 6 clusters. Customer, Account and Online User has higher pheromone deposited, so their chances of being selected are more. For searching component related to this type of query we need to visit 6 clusters



	Class	Pheromone	Cluster
1	Customer	9	1
2	Account	6	1
3	Online User	4	1
4	Web Server	2	6
5	Branch	2	1
6	User	2	1
7	Corporate Customer	1	1
8	Personal Customer	1	1
9	Order	1	1
0	Member Category	1	1
1	Order	1	2
2	Bank	1	1
3	ABC Database Wrapper	1	5
4	DataBase	1	5
5	ATM	1	1
6	Transaction	1	1
7	SuppliersAccount	1	1
8	chatclient	1	2

Figure 5.6: Keywords from multiple domains

Precision = 1 (ClusterNumber 1 to 6) × / 6 (Number of Clusters to be visited) =1/6=0.16

$$CR = \frac{\text{No. of Clusters to be visited} \times \text{Components in each cluster} \times \text{Total Terms}}{\text{Total Number of Components in Repository}}$$

$$CR = (6 \times 100 \times 150)/150000 = 0.6$$

5.3 Component Search

Probability based search technique is proposed to locate an appropriate component in an ant based repository. In searching component, first the cluster is identified on the basis of attribute relationships [143, 145]. This cluster is considered to represent the component domain. Keywords are identified from the user query for ambiguity resolution. Before actually moving to search following probabilities are calculated that will guide the search in appropriate direction.

First calculate the *probability of component to be in the cluster* (P_c)

$$P_c = \frac{\text{Number of attributes define the component}}{\text{Total Number of Attributes in the cluster}}$$

Since more the number of attributes, less the uniquely identified attributes that define the component and thus more the interaction between the components so the probability of selection of the component becomes less.

The *average attribute factor* of the component in the cluster (P_a)

$$P_a = \frac{Total\_NumberofAttributes\_in\_cluster}{Total\_Numberof\_Components}$$

P_a will lie between 0 and 1 in case all the components have only one attribute and some can be common. It can be 1 in case some of the components have more than one attribute but some are common so the average ratio is one. If the number of attributes that define the component are greater than one and there are some common attributes also, then the value of P_a can be greater than 1. If the value of P_a is much higher than 1 then the probability of finding the good or appropriate component in that cluster is very less. So for better search we can sequence our search in the increasing order of P_a.

Now the probability of each component to be selected in the cluster by the query is defined as:

$$P_q = \frac{Number\_of\_Matched\_Attributes\_of\_Component}{Total\_Number\_of\_Attributes\_in\_Query}$$

P_q guides the efficiency of the query i.e. of the component to be searched,

$$Efficiency = \left(\frac{P_q}{p}\right) * \left(\frac{P_c}{P_a * K}\right)$$

Where K is constant and is derived from the WordNet tool depending upon the attribute value of the component to that of search query.

p is the number of clusters traversed to find the resultant component. If p is increasing it means efficiency is decreasing.

So this search technique is dependent on the overall cluster and the clustering defined by the ants. More the number of attributes less accurate are the results. Thus the query has to be very specific. Thus the density of the attributes in the cluster do affect the query output as more dense the clusters lesser the value of P_a and thus greater the efficiency of the output.

After selecting the similar clusters, appropriate component is searched starting with first cluster. Here clusters represent domains of components. On the basis of user query appropriate terms are selected using WordNet to represent user query. Based upon query terms candidate clusters are identified from the repository. These clusters are searched considering value of P_a. Here a set of 150 terms to represent reusable components in one cluster of the repository has been considered. Various clusters to represent various domains have been considered. On the basis of cluster terms rules are generated. These rules will help to locate an appropriate component in the repository also these rules may help to store and represent any component in repository. Initial pheromone of 1 is updated on the basis of term's existence in the rule. If any term continues to be in the next iteration then pheromone associated with the term in next generation will increase this means more pheromone will get deposited otherwise it will get evaporated.

5.3.1 Pheromone Update

The pheromone of each term occurring in a rule is updated in each iteration. If a term occurs in the rule it means ant visited this term in generating this rule correspondingly its pheromone deposited increases. But if any term does not occur in current rule it means ant has not visited this term and correspondingly pheromone is evaporated. For next iteration these updated pheromone values in database are used to generate next rule. Iterations are repeated till a rule with quality 0.9 or two consecutive rules with same or nearly same quality value are generated. As current ant completes the construction of rule, the rule pruning takes place. The strategy for rule pruning is same as used in other approaches but rule quality criteria in this approach is different [53, 106, 107, 111, 144].

The basic idea here is to iteratively improve the quality of the constructed rule. Initially terms in first rule are selected on the basis of keyword, its synonyms and their interrelationships. These terms are then compared with the domain class terms stored in a separate domain database. The terms, which are present in constructed rule and are also there in the domain class, will play active rule in rule generation. Other terms are iteratively removed one by one. In each iteration the terms, which are matched with domain their pheromone is increased according to relation given in equation (1) and equation (2). The quality of the current rule is calculated according to equation (3). The

terms that does not occur in the rule their pheromone has to be decreased to show evaporation. Pheromone evaporation for unused terms is calculated by normalization of the pheromone of each term i.e. by dividing each term's pheromone with sum of pheromone value of all participating terms.

$$\tau_i = \frac{1}{\sum_{i=1}^{a} b_i} \tag{1}$$

Where τ_i is $term_i$, the amount of pheromone associated with each term found by the ant is increased in proportion to the quality of the rule.

a – Total number of attributes in $term_i$, b_i is the number of possible values that attributes can have.

$$\tau_i(t+1) = \tau_i(t) + \tau_i(t) * Q \forall i \in R \tag{2}$$

Where R is the set of terms occurring in the current rule constructed by the ant in iteration t.

$$Q = \frac{TP}{TP + FN} * \frac{TN}{FP + TN} \tag{3}$$

Where Q is the quality of a rule.

TP- True Positives, is the number of terms covered by the rule that have the domain predicted by the rule.

FP- False Positives, is the number of terms covered by the rule but that have the different domain predicted by the rule.

FN- False Negatives, is the number of terms that are not covered by the rule but have the domain predicted by the rule

TN- True Negatives, is the number of terms that are not covered by the rule and do not from the predicted domain

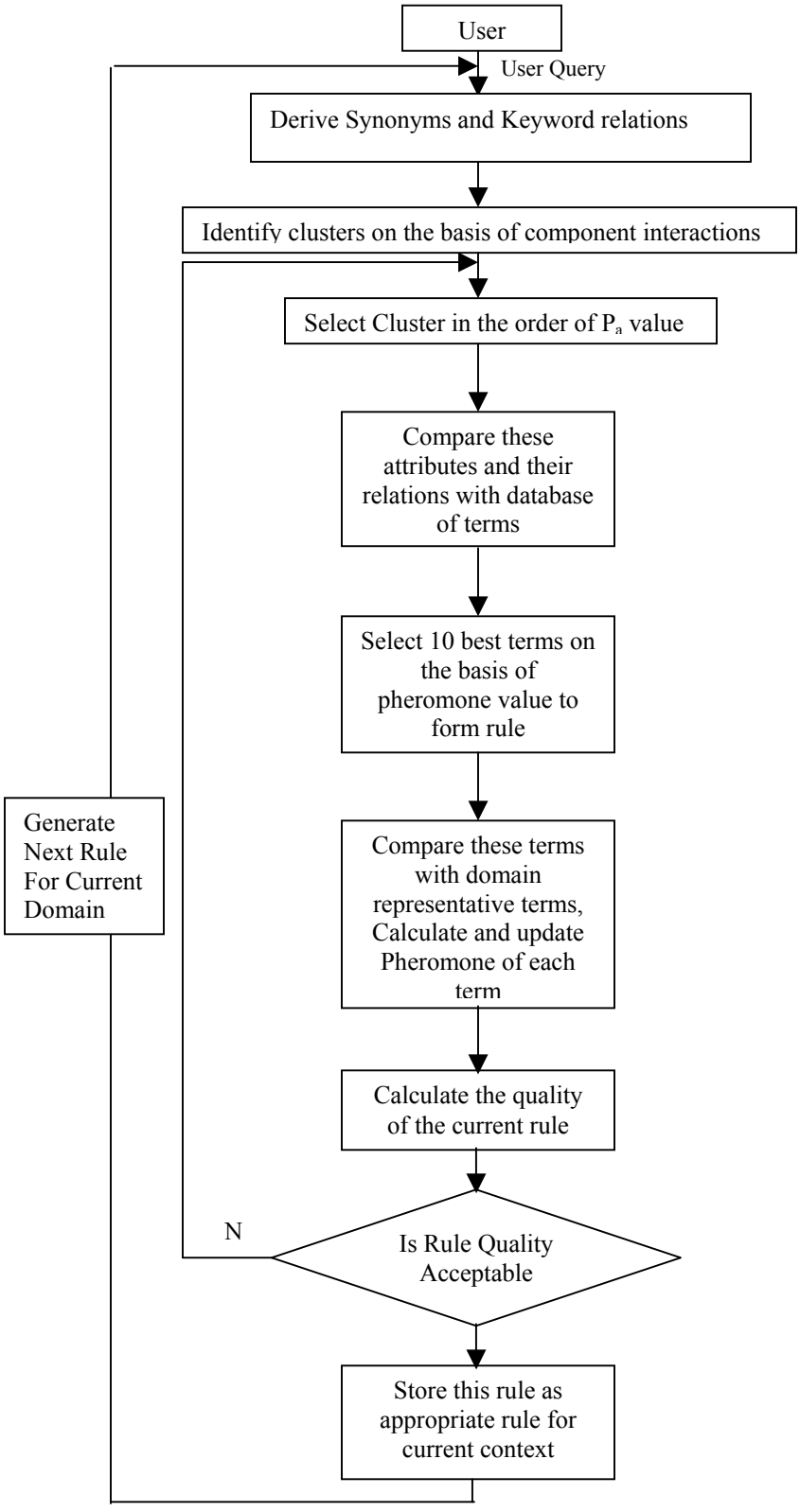


Figure 5.7: Rule Generation Steps

5.3.2 Experimental Results

Consider a database of 150 terms to represent the reusable software components. Initially terms are selected on the basis of keywords, their synonyms and relationships among these terms. Initial pheromone value of 1 is assigned to all terms in the database. A user query of 'Linked List' is considered from data structure domain. In domain database there are 11 terms to represent this type of components in the software repository. On searching our Representative Terms database given in Table 5.1 on keywords, 10 terms are selected to form Rule 1 given in Table 5.2.

Table 5.1 Representative Terms database containing 150 terms

Terms	Attributes	Values	Initial Pheromone
T1	A1 A2 A3 A4	V1 V2 V3 V1 V2 V1 V1 V2 V3	1
T2	A1 A2 A3	V1 V2 V1 V2 V3 V1	1
T3	A1 A2 A3 A4 A5	V1 V2 V3 V1 V2 V1 V2 V3 V4 V1 V1 V2	1
T4	A1 A2	V1 V2	1
T5	A1 A2 A3	V1 V2 V3 V1 V2 V1	1
T6	A1 A2 A3	V1 V2 V3 V1 V1	1
T7	A1 A2 A3	V1 V1 V2 V1	1
T8	A1 A2 A3 A4	V1 V1 V2 V1 V1 V2	1
T9	A1 A2	V1 V1 V2 V3 V4	1
T10	A1 A2	V1 V1	1
T11	A1 A2 A3	V1 V1 V1 V2 V3	1
T12	A1 A2 A3	V1 V2 V1 V2 V1 V2	1
T13	A1	V1 V2	1

	A2	V1 V2	
T14	A1 A2	V1 V1	1
T15	A1 A2 A3	V1 V1 V1	1
T16	A1 A2 A3	V1 V1 V1 V2	1
T17	A1 A2	V1 V2 V1 V2	1
T18	A1 A2 A3 A4	V1 V2 V1 V2 V1 V2 V1	1
T19	A1 A2 A3	V1 V2 V1 V2 V3 V1 V2 V3 V4	1
T20	A1 A2	V1 V2 V3 V1	1
…….			
T150	A1	V1	1

Select 10 terms from Table 5.1 on the basis of similarity of attributes and their inter relationships, these terms will form the first rule

Table 5.2 Rule 1

1	T1
2	T3
3	T5
4	T6
5	T8
6	T11
7	T13
8	T16
9	T18
10	T19

Now compare terms in Table 5.2 with domain terms given in Table 5.3

Table 5.3 Data Structure domain terms

1	T1
2	T3
3	T5
4	T6
5	T8
6	T9
7	T11
8	T13
9	T16
10	T100
11	T101

TP=8 (T1, T3, T5, T6, T8, T11, T13, T16), FP=2 (T18, T19)

FN=3 (T9, T100, T101), TN=5 (T18, T19, T9, T100, T101)

Now find the quality of Rule1 using equation (3), Q=0.52. Next step is to update the pheromone of each term occurring or not occurring in the rule using equation (1) and equation (2)

For updating pheromone deposited, put values of a and b in equation 1 from database containing 150 terms for all participating terms, we shall get:

T1=1/9, T3=1/12, T5=1/6, T8=1/6, T11=1/5, T13=1/4, T16=1/4

Using equation (2), the amount of pheromone increment is to be calculated as given below

T1=1/9+1/9*0.52=0.17
T3=1/12 + 1/12 *0.52=0.12
T5=1/6 + 1/6 * .052=0.26
T6=1/5 + 1/5 * .052=0.30
T8=1/6 + 1/6 * .052=0.26
T11=1/5 + 1/5 * .052=0.30
T13=1/4 + 1/4 * .052=0.38
T16=1/4 + 1/4 * .052=0.38

To decrease the pheromone of two not occurring terms, T18 and T19, it will be computed by dividing the current pheromone value (not modified) of term by the total of pheromone values for all participating terms (modified).

Total=Increased pheromone values (T1+T3+T5+T6+T8+T11+T13+T16)=10.17

So the pheromone value of T18 and T19 is to be decreased by 1/10.17 i.e. by 0.1.

Table 5.4 RepresentativeTerms Database with Updated pheromone values after Rule1

Terms	Attributes	Values	New Pheromone
T1	A1 A2 A3 A4	V1 V2 V3 V1 V2 V1 V1 V2 V3	1.17
T2	A1 A2 A3	V1 V2 V1 V2 V3 V1	1
T3	A1 A2 A3 A4 A5	V1 V2 V3 V1 V2 V1 V2 V3 V4 V1 V1 V2	1.12
T4	A1 A2	V1 V2	1
T5	A1 A2 A3	V1 V2 V3 V1 V2 V1	1.26
T6	A1 A2 A3	V1 V2 V3 V1 V1	1.30
T7	A1 A2 A3	V1 V1 V2 V1	1
T8	A1 A2 A3 A4	V1 V1 V2 V1 V1 V2	1.26
T9	A1 A2	V1 V1 V2 V3 V4	1
T10	A1 A2	V1 V1	1
T11	A1 A2 A3	V1 V1 V1 V2 V3	1.30
T12	A1 A2 A3	V1 V2 V1 V2 V1 V2	1
T13	A1 A2	V1 V2 V1 V2	1.38
T14	A1 A2	V1 V1	1
T15	A1 A2 A3	V1 V1 V1	1
T16	A1 A2 A3	V1 V1 V1 V2	1.38
T17	A1 A2	V1 V2 V1 V2	1
T18	A1 A2 A3 A4	V1 V2 V1 V2 V1 V2 V1	0.9
T19	A1 A2 A3	V1 V2 V1 V2 V3 V1 V2 V3 V4	0.9

T20	A1 A2	V1 V2 V3 V1	1
……..			
T150	A1	V1	1

Now terms are selected to form Rule2 on the basis of their pheromone value.

Table 5.5 Rule2

1	T1
2	T3
3	T5
4	T6
5	T8
6	T9
7	T11
8	T13
9	T16
10	T17

Again compare it with domain terms in Table 5.3 to select occurring and not occurring terms in Rule2.

TP=9, FP=1, FN=2, TN=3, Quality of the Rule2 is Q=0.61.

Now we shall calculate pheromone increment and decrement for participating and not participating terms respectively by using equations 1 & 2. Pheromone increment for participating terms, T1, T3, T5, T6, T8, T9, T11, T13, T16, is calculated. Pheromone decrement for not participating terms, only T17, is calculated.

Table 5.6 RepresentativeTerms Database with Updated pheromone values after Rule1

Terms	Attributes	Values	New Pheromone
T1	A1 A2 A3 A4	V1 V2 V3 V1 V2 V1 V1 V2 V3	1.35
T2	A1 A2 A3	V1 V2 V1 V2 V3 V1	1
T3	A1 A2 A3 A4 A5	V1 V2 V3 V1 V2 V1 V2 V3 V4 V1 V1 V2	1.25
T4	A1 A2	V1 V2	1
T5	A1 A2 A3	V1 V2 V3 V1 V2 V1	1.53
T6	A1 A2	V1 V2 V3 V1	1.62

	A3	V1	
T7	A1 A2 A3	V1 V1 V2 V1	1
T8	A1 A2 A3 A4	V1 V1 V2 V1 V1 V2	1.53
T9	A1 A2	V1 V1 V2 V3 V4	1.32
T10	A1 A2	V1 V1	1
T11	A1 A2 A3	V1 V1 V1 V2 V3	1.62
T12	A1 A2 A3	V1 V2 V1 V2 V1 V2	1
T13	A1 A2	V1 V2 V1 V2	1.78
T14	A1 A2	V1 V1	1
T15	A1 A2 A3	V1 V1 V1	1
T16	A1 A2 A3	V1 V1 V1 V2	1.78
T17	A1 A2	V1 V2 V1 V2	0.93
T18	A1 A2 A3 A4	V1 V2 V1 V2 V1 V2 V1	0.9
T19	A1 A2 A3	V1 V2 V1 V2 V3 V1 V2 V3 V4	0.9
T20	A1 A2	V1 V2 V3 V1	1
…….			
T150	A1	V1	1

Select terms to form Rule3

Table 5.7 Rule3

1	T1
2	T2
3	T3
4	T5
5	T6
6	T8
7	T9
8	T11
9	T13
10	T16

Repeat all above steps to calculate quality and pheromone increment/decrement to form new rule. It is observed that

TP=9, FP=1, FN=2, TN=3, Q=0.61, it is same as of Rule2. So it converges here. Now this rule can be selected to represent component Linked List in the repository. This rule will help any user in future to retrieve the Linked List component from the software repository.

5.4 Empirical Results

The approach described in previous section has been fully developed in form a tool in Visual Basic and Oracle with 1,50,000 total components of synthetic data from various 22 domains. We have tested our system on 1,50,000 components and used 10 terms to describe one rule. Implementation has been tested on Intel M 1.73 GHz processor with 1 GB of RAM.

In order to evaluate the performance we use the following criteria [82, 128]:

- **User Effort**: This consists of all the effort user must expend in order to use the repository system. It is very difficult to formally measure the user effort, however we believe that the efforts that must be invested for using our tool are minimum. User query can be formed in natural language keywords hence no need to learn any formal language. Keywords can be selected to represent the functionality and domain of application.

- **Efficiency**: This refers to the average interval between the time a user query is given and the time an answer is returned. Efficiency becomes an issue if retrieval takes too much time and user starts complaining. This has been observed while working with current system that the response time is reasonable. System takes time for initial clustering. Once all components are clustered in various domains then user query does not take much time. Execution of the user queries has shown that time taken to perform the query was dominated by the clustering time. Here time and space is not big issue, as appropriate reusable component will save future effort in development.

- **Maintenance effort**: This effort consists of all the effort required to keep the system working and up to date. These efforts particularly include clustering time. The

clustering is done automatically. Each time new component is inserted in the repository, it requires clustering. Because this component will go to appropriate cluster for later classification and retrieval. It took about 58 seconds for initial clustering of 1,50,000 components. Once clustered then user query does not take much time. As user query in case IV with 3 keywords regarding email took about 2.5 seconds on Pentium IV 1.73 GHz processor with 1GB RAM.

- **Retrieval effectiveness**: This is the most important performance criterion; it refers to the ability of the system to provide results as needed by user. We have considered here recall, precision and coverage factor to assess the retrieval effectiveness. The example cases considered in section 5.2 shows that 60% of the components needed to be visited for mixed query of 14 keywords to achieve minimum precision of 0.16 without missing any relevant component i.e. recall of 1. In average case a query of 3 keywords require 20% of the components to be visited to achieve precision of 0.5 with recall of 1.

5.5 Comparative Analysis

Conventional retrieval techniques are not suitable for component clustering. If we consider keyword based i.e. natural language based retrieval techniques for clustering, they will suffer from ambiguity problem means keyword relations to form clusters will not be appropriate. Only domain based clusters can be generated. Also trying clusters with formal specifications will also not be feasible, as it will not work with approximate matches. Some sophisticated technique may help in clustering the software component to facilitate better retrieval [Mili et. al. 97]. Assessment criteria used in [49, 50, 51] is compactness and separation of any cluster. It is calculated on the basis of distance in various members of a cluster. These types of criteria may not be applicable to software components as same functionality may be useful in many different contexts. Assessment criteria used in section 4.4 in table 4.13, is used here to compare the quality of retrieval.

Table 5.8 Comparative Analysis

Method	Cases [Section 5.2]	Precision	Recall	Coverage Ratio
Ant Colony Based Retrieval	CASE I	VH	VH	VL
	CASE II	M	VH	VL
	CASE III	VH	VH	VL
	CASE IV	M	VH	VL
	CASE V	L	VH	M

It is observed from above cases that there is considerable improvement in precision, recall and coverage ratio. Very high (=1) recall is achieved, with on average high or medium precision and very low coverage ratio, which is a major advantage over conventional retrieval techniques.

5.6 Conclusions

We have presented an ant based clustering technique for arranging components in various domains in software repository. Various cases considered have shown the cluster formation and then keywords representation for various domains. Also an ant colony algorithm based technique for rule discovery has been proposed to identify appropriate component from the candidate clusters. This ant based search technique first selects clusters on the basis of average attribute factor. The major goal is to discover rules to represent various domains of reusable components. Further these rules help in retrieving appropriate component from the reusable components repository. We have considered here one sub domain ‘Data Structures’ in set of components and then ‘Linked List’. It has been found that only three iterations were required to discover a rule to represent or select ‘Linked List’. Rule convergence criteria considered here is either attaining quality value of 0.9 or two consecutive rules with same quality. Experimental results shows that terms successfully can generate rules for component repository. Ant colony based component clustering also helped to put similar components in same domains. Clustering of components before applying search helps to reduce the search space. As the maximum

value of Coverage Ratio is 0.6 in complex query case, it shows that only 60% components are visited to locate the most relevant match. Also Recall value of 1 indicates that no relevant component is missed. Precision value ranges from 0.5 to 1; it indicates that in average case 50% relevant components are compared to reach the optimal match. With the help of clustering no relevant match has been missed.

Chapter 6

Conclusions and Scope for Future Work

This chapter concludes the thesis and discusses future scope of the work. The main objective of the thesis is to design and develop a reusable software components repository. Reusable software component may include any algorithm, data flow diagram, source code or executable code, test case and test plans etc. Reusable software component means that the said artifact can be used for required functionality rather than developing it from scratch. Software components that are developed keeping in mind their future reusability require additional efforts to make them general purpose. Not only creating the reusable software component is difficult but also specifying these components appropriately to facilitate efficient and convenient retrieval also requires lot of efforts. An attempt has been made in this thesis to specify components in repository in a way that helps in retrieving the components. In third chapter of this thesis a natural and formal specifications based hybridized technique is proposed for component retrieval. Then in fourth chapter a Genetic Algorithm based technique for component retrieval has been demonstrated with suitable cases. Finally, in chapter five an ant colony optimization based rule generation technique for component retrieval is discussed.

6.1 Conclusions

The major contributions of the reported work are outlined as follows:

1. A formal specifications based reusable component repository has been developed. The repository makes use of natural keyword based and formal specifications of the component for classification and retrieval. Natural keywords include Name of the Component, Number of Parameters, Type of Parameters, Return Type and Synonyms. Formal specification in Z markups considers Declaration and Predicate part, Pre conditions and Post conditions, and schema definitions. This system considers user input query in form of mixed form of natural description and formal schema. The proposed matching technique first short-lists the candidate components on the basis of keywords and their synonyms. After this step it applies formal matching criteria on

candidate components to resolve ambiguity. It is a flexible weighted method to calculate percentage match of the user query with candidate components to retrieve the best possible match for reuse. We have been able to locate a component with even 100% match by assigning weight of 80% to formal and 20% to natural specifications. This has been observed if we further increase the percentage weight assigned to formal specifications then it will be able to locate only exact matches but it will miss all approximate matches. But in software reuse sometimes, approximate matches after minor adaptations are very useful.

2. Use of flexible weighted scheme helps user to locate most appropriate match. Depending upon software development phase in which user is working the weights can be varied. During early development phases like in requirement analysis, user requirements are in the form of natural descriptions. User may assign higher weight to natural descriptions as compared to formal descriptions to locate proper match from repository. However, in later development phases more concrete forms of requirements are available with user, higher weight can be assigned to formal descriptions as compared to natural descriptions. Thus the selection of weights assigned to natural and formal descriptions depends on which development phase user is working in. During earlier development phases, more weight should be assigned to natural specifications and during later development activities higher weight may be assigned to formal specifications.
3. Genetic Algorithm based repository makes use of natural keywords and genetic description for classification and retrieval of components. Natural keywords and their synonyms help to prune the initial population for genetic algorithm based second step. The major achievement in this case is coverage ratio. In section 4.3 case II, it has been shown that for given user query the precision and coverage ratio are 0.19 and 0.104 respectively. It is observed that in genetic algorithm based step only 10% of the total components needed to be visited without missing any relevant component to match most appropriate component. In case III more complex match, only 17% component are visited to match the appropriate component with precision of 0.178 and without missing any relevant component.

4. In the ant colony based classification and retrieval, the components are clustered before applying search. Component clustering helps to reduce the search space. To select the cluster average attribute factor of cluster has been used for ant colony based rule generation. We have considered a database of 150 terms from 22 domains to represent the reusable components. A set of 10 terms has been considered to represent a rule and the rule quality has been considered as termination criteria. It is observed that after initial clustering it takes only three iterations to converge the rule.
5. In ant colony based rule generation in worst case 60% of the components are required to visit to locate the component with precision of 0.16 and without missing any relevant component. In average case a precision of 0.4 has been achieved with recall of 1 and coverage ratio of 0.25. In best case precision of 1 is achieved with recall of 1 and only 10% of the total components are visited to converge.
6. Very high (=1) recall, with on average high or medium precision and very low coverage ratio, is a major advantage over conventional retrieval techniques.

6.2 Scope for Future Research

This work opens up a number of avenues for future research. The research can be extended in several directions and some of them are summarized below:

1. In proposed flexible weighted approach, we have considered only code components in the repository. Further research can be made to include all types of components such as Software Requirements Specification (SRS) documents, Data Flow Diagrams, UML diagrams, Test Cases etc. The weights assignment for such cases will be different for natural and formal specifications. Also our implementation considers static features of the component while searching. Work can be further extended to include dynamic behavior of the component while searching.
2. We have used Z based formal descriptions for similarity matching. In deciding percentage match declaration and predicates, pre/post conditions and schema definitions are considered. As a future work, constraint similarity using Object Constraint Language (OCL) can be used for similarity matching. OCL constraints can be used in conjunction with UML.

3. The genetic description of the component in our implementation uses relative weights between 0 and 1 for the attributes. Some automatic techniques can be thought of to decide weights as per relevance. Also mutation probability here is considered independently, Selection of mutation probability can be made dependent on certain factors related to reusability.
4. User experience based collaborative tagging can also be included in component attributes in repository. Collaborative tagging means adding terms on the basis of user experience with the component. It may help further to improve the retrieval of appropriate component. Each time a user retrieves a component from the repository the user will add few terms based on his/her experience regarding the reusability of the said component.

References

1. Amy Moormann Zaremski and Jeannette M. Wing, "Signature Matching: A Key to Reuse," First ACM SIGSOFT Symposium on the Foundations of Software Engineering, Software Engineering Notes, vol. 18, no. 5, pp 182-190, 1993.
2. Andreou A. S., Vogiatzis D. G., Papadopoulos G. A., "Intelligent Classification and Retrieval of Software Components", Proceedings of the 30th Annual IEEE International Computer Software and Applications Conference (COMPSAC'06), 2006.
3. Andy Podgurski and Lori A. Clarke, "A Formal Model of Program Dependences and Its Implications for Software Testing, Debugging, and Maintenance," IEEE Transactions on Software Engineering, vol. 16 no. 9, pp 965-979, September 1990.
4. Ansari A. Q. and Moinuddin, "Towards Rough Set Based Information System", Proc. International Conference on Knowledge Based Computer Systems (KBCS'98), Mumbai, India, pp 273-286, Dec.1998.
5. Axel van Lamsweerde, "Formal Specifications: A Roadmap" Future of Software Engineering, Limerick, Ireland, ACM Press, June 2000.
6. Ayad H. and Kamel M., "Topic Discovery from Text Using Aggregation of Different Clustering Methods", in procedings of Advances in Artificial Intelligence: 15th Conference of the Canadian Society for Computational Studies of Intelligence, Calgary, Canada, pp 161-175, 2002.
7. Barry W. Boehm, "Improving Software Productivity", IEEE Software, pp 43-57, September 1987
8. Bersini H., Oury C., Dorigo M., "Hybridization of Genetic Algorithms", Technical Report No. IRIDIA 95-22, IRIDIA, Universite Libre de Bruxelles, Belgium, 1995.
9. Boley D., Gini M., Gross R., Han E., Hasting K., Karypis G., Kumar V, Mobasher B. and Moore J., "Partitioning-Based Clustering for Web Document Caregorization", Journal of Decision Support Systems, no. 27, pp 329-341, 1999.

10. Bollacker K.D., Ghosh J., "A Supra-Classifier Architecture for Scalable Knowledge Reuse", in proceedings of International Conference on Machine Learning (ICML 98), pp 64-72, 1998.
11. Brad J. Cox. "Planning the Software Revolution." IEEE Software , vol. 7 no. 6. pp 25-35, November 1990.
12. Charles Rich and Richard Waters, "Automatic Programming: Myths and Prospects," IEEE Computer, pp 40-51 August 1988.
13. Charles Rich and Richard Waters, "The Programmer's Apprentice: A Research Overview," IEEE Computer, pp 11-25, November 1988.
14. Charles W. Krueger, "Software Reuse" ACM Computing surveys, vol. 24 no. 2, pp 131-183, June 1992.
15. Crnkovic I. and Larsson M., "A Case study: Demands on Component-based Development", Proceedings of ICSE, 22nd International Conference on Software Engineering, Ireland, pp 22-30,2002.
16. Cutler M., Deng H., Maniccam S.S. and Meng W., "A New Study Using HTML Structures to Improve Retrieval", in proceedings of 11th IEEE International Conference on Tools with AI, pp 406-409, IEEE 1999.
17. Dana Vrajitoru, "Crossover Improvement for the Genetic Algorithm in Information Retrieval", Information Processing & management, 1998, vol. 34, no. 4, pp 405-415.
18. Daniel A., "Ant Colony Optimization: From Biological Inspiration to an Algorithmic Framework", Technical Report TR 013, Centre for Intelligent Systems and Complex Processes, Swinburne University of Technology, Melbourne, Australia, 2006.
19. David Carrington, David Duke, Ian Hayes and Jim Welsh, "Deriving Modular Designs from Formal Specifications", Software Engineering notes, vol. 18 no. 5, pp 89-98, Los Angeles, California, USA
20. David M. Balda and David A. Gustafson, "Cost-estimation models for the reuse and prototype software development lifecycles", ACM SIGSOFT, pp 42-50, July 1990
21. Dhillon I. S. and Modha D. S., "Concept Decompositions for Large Sparse Text Data using Clustering", in Journal of Machine Learning, vol. 42, no. 1, pp 143-175, 2001.

22. Dillon L. K. and Sankar S., ``Introduction to the Special Issue on Formal Methods in Software Practice", IEEE Transactions Software Engineering, vol. 23 no. 5, May 1997, pp 265-266.
23. Donald H. Kraft, Frederick E. Petry, Bill P. Buckles, Thyagarajan Sadasivan, "The use of genetic programming to build queries for information retrieval", pp 468-473, IEEE Software, 1994.
24. Dorigo M. and Thomas S. , "Ant Colony Optimization", Prentice-Hall of India Publication, New Delhi, pp 223-244, 2005.
25. Ellis Horowitz and John B. Munson, "An Expansive View of Reusable Software," IEEE Transactions on Software Engineering, vol. 10 no. 5, pp 477-487, 1984.
26. Emmrich W. and Kaveh N., "Component Technologies: Javabeans, COM, CORBA, RMI and CORBA component model", ICSE'02 International Conference on Software Engineering, Orlendo, Florida, 2002.
27. Enoch Wang and Betty Cheng, ``A Rigorous Object-Oriented Design Process", Technical Report, MSUCPS: TR97-146, Department of Computer Science, Michigan State University, East Lansing, Michigan, 1997.
28. Enoch Wang and Betty Cheng, ``Integrating the Functional Model into Object-Oriented Design", Technical Report, MSUCPS: TR97-120, Department of Computer Science, Michigan State University, East Lansing, Michigan, 1997.
29. Enoch Y. Wang, Heather A. Richter, Betty C. Cheng, ``Formalizing and Integrating the Dynamic Model within OMT", Proc. of IEEE International Conference on Software Engineering (ICSE97), pp 45-55, Boston, Massachusetts, May 1997.
30. Evered M., Menger G., Schmolitzky A., Keedy J. L., "Software Reuse in an Object Oriented Framework: Distinguishing types from Implementation and Objects from Attributes", 6th International Conference on Software Reuse, pp 420-435, Vienna, 2000
31. Fábio A. M. Porto, Ana M. C. Moura, Adriana P. Fernandez, Abílio Fernandes, Fábio J. C. Silva, Gilda H. B. Campos, Laura Coutinho, "ROSA: A Data Model and Query Language for e-Learning Objects" IPGL DB Research Conference, PUC-Rio, Abril 2003.

32. Fellbaum C.(Ed), "WordNet-An Electronic Lexical Database", MIT Press, 1998.
33. Fichman R. G.and Kemerer C. F., "Incentive Compatibility and Systematic Software Reuse", Journal of Systems and Software, vol. 57 no. 1, pp 45-60, 2002.
34. Fisher D., "Iterative Optimization and Simplification of Hierarchical Clusterings", Journal of Artificial Intelligence Research, no. 4, pp 147-180, 1996.
35. Gail E. Kaiser and David Garlan, "Melding Software Systems from Reusable Building Blocks," IEEE Software, pp 17-24, July 1987.
36. Gamma E. et al., "Design Patterns: Elements of Reusable Object-Oriented Software", Addison Wesley Professional, 1997.
37. Gellersen W., Wicke R., Gaedke M., "Web Compostion: An Object-Oriented Support System for the Web Engineering Lifecycle", Computer Networks and ISDN Systems Special Issue on the 6th International World-Wide Web Conference, Santa Clara, CA, USA, 1997.
38. Gerald C. Gannod and Betty C. Cheng, ``A Formal Automated Approach for Reverse Engineering Programs with Pointers", in proceedings of IEEE Automated Software Engineering, November 1997.
39. Gerhard Fischer, "Cognitive View of Reuse and Design." IEEE Software, pp 60-72, July, 1987.
40. Ghosh J., Strehl A., and Merugu S., "A Consensus Framework for Integrating Distributed Clusterings under Limited Knowledge Sharing", in proceedings of NSF Workshop on Next Generation Data Mining, Baltimore, pp 99-108, 2002.
41. Gianluigi Caldiera and Victor R. Basili, "Identifying and Qualifying Reusable Software Components", IEEE Computer vol. 24 no. 2, pp 61-70, February 1991.
42. Goldberg, D.E, "Genetic Algorithms in Search, Optimization and Machine Learning", Reading, MA: Addison-Wesley Publishing Co., 1989.
43. Gordon, M, "Probabilistic and Genetic Algorithms for Document Retrieval", Communications of ACM, vol. 31 no. 2, pp 152-169, 1988.
44. Gupta S. K., Bhatnagar Vasudha and Wasan SK, "Mining of Data", IETE Journal of Research, Special issue on Data and Knowledge Engineering, vol. 47 no.1, pp 5-18, Jan/Feb 2001.

45. Gupta S. K., Bhatnagar Vasudha, Wasan S. K., "Architecture for Knowledge Discovery and Knowledge Management", Journal of Knowledge Information Systems, vol 7, no 3, pp 310-336, 2005.
46. H. Dang Van, C. George, T. Janowski, and R. Moore, editors. Specification Case Studies in RAISE. Springer, 2002.
47. Hafedh Mili, Abdul Errahman El Wahidi, and Yoav Intrator, "Building a Graphical Interface for An Object-Oriented Tool for Software Reuse," in proceedings of TOOLS USA '92, ed. Bertrand Meyer, pp 81-96, Prentice Hall, Santa Barbara, CA, August, 1992.
48. Hafedh Mili, Odile Marcotte, and Anas Kabbaj, "Intelligent Component Retrieval for Software Reuse," in proceedings of the Third Maghrebian Conference on Artificial Intelligence and Software Engineering, Rabat. Morocco, April, 1994.
49. Halkidi M., Batistakis Y. and Vazirgiannis M.,"On Clustering Validation Techniques", Journal of Intelligent Information Systems", vol. 17, no. 2, pp 107-145, 2001.
50. Halkidi M., Vazirgiannis M. and Batistakis Y., "Quality Scheme Assessment in the Clustering Process", in proceedings of the Fourth European Conference on Principles and Practices of Knowledge Discovery in Databases (PKDD), pp 165-276, 2000.
51. Handl J. and Meyer B., "Improved Ant-Based Clustering and Sorting in a Document Retrieval Interface, in Parallel Problem Solving from Nature-PPSN VII, Seventh International Conference, Granada, Spain, 2002.
52. Hayes I. J., Jones C.B. and Nicholls J.E., "Understanding the Differences Between VDM and Z", ACM SIGSOFT Software Engineering Notes, vol. 19, no. 3, pp 75-81,ACM Press, New York, ISSN: 0163-5948, 1994
53. He J., Tan A., Tan C. and Sung S., "On Quantitative Evaluation of Clustering Systems", in Information Retrieval and Clustering, Wu W., Xiong H. and Shekhar S. (Eds), Kluwer, Boston, MA, 2003.
54. Henderson P. and Walters R., "Behavioural Analysis of Component-based Systems", Journal of Information and Software Technology, vol. 43 no.1, pp 161-169, 2001.
55. Hinchey Michael G., Stephen A., "Concurrent Systems: Formal Development In CSP", McGraw-Hill International Series in Software Engineering, 1995.

56. Hisham Haddad and Walter Fronter, "A Framework for domain-Specific Reusable Components", in proceedings of International Conference on Software Engineering Research and Practices, pp 384-390, June 2003.
57. Hisham M. Haddad and Erol Biberoglu, "Component-Based Software engineering: Issues and Concerns", Proceedings of International Conference on Software Engineering Research and Practices, pp 391-397, June 2003.
58. Hoe K. M., Lai W. K., Tai T. S. Y. , "Homogeneous Ants for Web Document Similarity Modeling and Categorization", in Ant Algorithms, LNCS 2463, pp 256-261, Springer, 2002.
59. Holden N. and Freitas A.A., "Web Page Classification with an Ant Colony Algorithm", in Parallel Problem Solving from Nature-PPSN VIII, LNCS 3242, 1092-1102, Springer Verlag, September 2004.
60. Holger Billhardt, Daniel Borrajo, Victor Maojo, "Using Genetic Algorithms to Find Suboptimal Retrieval Expert Combinations", SAC 2002, Madrid, Spain, pp 657-662, 2002.
61. Houston I. and King S., "CICS Project: Experiences and Results from the use Z in IBM", Proceedings VDM 91-Formal Software Development Methods, LNCS 552, pp 588-596, 1991.
62. Information Technology – Z Formal Specification Notation – Syntax, Type System and Semantics, ISO/IEC 13568:2002.
63. James M. Conrad, Mark Baldwin, Sean Curran, and Larry Martin, "Using a New Software Product Development Process for a Code Reuse Project," in proceedings of the 1999 Engineering of Computer-Based Systems Conference, Nashville, TN, pp 34-40, March 1999.
64. Janus Laski and Wojciech Szermer, "Regression Analysis of Reusable Program Components," in Advances in Software Reuse, IEEE Computer Society Press, Lucca, Italy, pp 134-141, March 1993.
65. Jean-Marc Morel and Jean Faget, "The REBOOT Environment," in Advances in Software Reuse, IEEE Computer Society Press, Lucca, Italy, pp 80-88, March 1993.
66. Jeong Ah Kim, "Component Adaptation Through Adaptation Components", Proceedings of International Conference on Software Engineering Research and Practices, pp 379-383, June 2003.

67. Jiang W., Xu Y., and Xu Y., "A Novel Data Mining Method Based on Ant Colony Algorithm", in Advance Data Mining and Applications, LNAI(LNCS) 3584, pp 284-291, Springer-Verlag Berlin Heidelberg 2005.
68. John E. Gaffney and R. D. Cruickshank, "A General Economics Model of Software Reuse," in proceedings of the 14th International Conference on Software Engineering, ACM Press, Melbourne, Australia, pp 327-337, May 1992.
69. Johnson E. and Kargupta H., "Collective, Hierarchical Clustering from Distributed, Hetrogeneous data", in Large Scale Parallel KDD Systems, vol. 1759, LNCS, pp 221-244, Springer-Verlag, 1999.
70. Jones C., "Systematic Software Development using VDM, Jones, C.B. Prentice Hall International Series in Computer Science", 1986.
71. Jones G., "Object Store 6.0, White Paper", October 1999, Butler Group, (http://www.objectdesign.com/whitepapers/objectstore.ht).
72. Jones T. C., "Reusability in Programming: A Survey of the State of the Art," IEEE Transactions on Software Engineering, vol. 10 no. 5, pp 488-494, September 1984.
73. Jorng-Tzong Horng, Ching-Chang Yeh, "Applying genetic algorithms to query optimization in document retrieval", Information Processing and Management, no. 36, pp 737-759, 2000.
74. Jose R. Herrero and Juan J. Navarro, "Building Software Via Shared Knowledge", Proceedings of International Conference on Software Engineering Research and Practices, pp 861-870, June 2003.
75. Joseph A. Goguen, "Reusing and Interconnecting Software Components," IEEE Computer, pp 16-28, February 1986.
76. Joseph Goguen and Grant Malcolm, "An Executable Course in the Algebraic Semantics of Imperative Programs", in Teaching and Learning Formal Methods, edited by Michael Hinchey and C. Nevill Dean, Academic Press, pages 161-179, 1996.
77. Jun-jang Jeng and Betty Cheng, ``Reusing Analogous Components", in IEEE Transactions on Knowledge and Data Engineering, vol. 9, no. 2, March-April, 1997.

78. Kang B. Y., Kim D. W. and Lee S. J., "Exploiting concept clusters for content-based information retrieval" in International Journal oF Information Sciences—Informatics and Computer Science, vol. 170, no. 2-4, 2005.
79. Kantardzic M., Filipovic A., Glavic H., "Algorithm for Matching Linguistically Imprecise Data Considering the Flat R-L Fuzzy Numbers in a Knowledge Base", in Artificial Intelligence and Information-Control Systems, edited by I. Plander, North-Holland, Amsterdam, pp 141-149, 1989.
80. Kantardzic M., Zurada J., New Generation of Data Mining Applications, IEEE Press and John Wiley, February 2005.
81. Kargupta H., Huang W., Krishnamoorthy and Johnson E., "Distributed Clustering using Collective Principal Component Analysis", Knowledge and Information Systems Journal Special Issue on Distributed and Parallel Knowledge Discovery, no. 3, pp 422-448, 2001.
82. Kaur Navdeep, Singh Rajwinder, Sarje Anil Kumar and Misra Manoj, "Performance Evaluation of Secure Concurrency Control Algorithm for Multilevel Secure Distributed Database Systems", ITCC vol. 1, IEEE Computer Society, pp 249-254, 2005.
83. Kotonya G., and Rashid A., "A Strategy for Managing Risk in Component-Based Systems", in proceedings of IEEE 26th Euromicro Conference, pp 12-21, Warsaw, Poland, 2001.
84. Kotonya G., Hutchinson J., Onyino W. and Sawyer P., "COTS-Based System Development: Processes and Problems", in proceedings of 4th International Information Society Conference: Development and Reengineering of Information Systems, Ljubljana, Slovenia, October 2001.
85. Kotonya G., Sommerville I., Hall S., "Towards a Classification Model for Component-Based Software Engineering Research", in proceedings of the IEEE 29th Euromicro Conference New Waves in System Architecture", 2003.
86. Kruchten P., Schonberg E., and Schwartz J., "Software prototyping using the SETL programming language," IEEE Software, vol. 1, no. 4, pp. 66-75, October 1984.
87. Kumar Sandeep, Agrawal D.P., Iyer P., "An improved Type-Inference Algorithm to expose parallelism in OO Programs", in the book Languages, Compilers and Run-Time Systems for Scalable Computers, Boleslaw K.

Szymanski and Balaram Sinharoy (Editors), Kluwer Academic Publishers, Boston, MA, Chapter 22, pp. 283–286, 1995.

88. Lee B., Moon B. and Wu C., "Optimization of Multi-way Clustering and Retrieval using Genetic Algorithms in Reusable Class Library", in proceedings of Fifth Asia Pacific Software Engineering Conference (APSEC '98), pp 4-11, 1998.
89. Lin F. T., Kao C. Y., Hsu C.C., "Applying the Genetic Approach to Simulated Annealing in Solving Some NP-hard Problems", IEEE Transaction on Systems, Man and Cybernetics, no. 23, pp 1752-1767, 1993.
90. Lumer E. and Faieta B., "Diversity and Adaptation in Population of Clustering Ants", in proceedings of the Third International Conference on Simulation of Adaptive Behaviour, vol 3, pp 499-508, MIT Press, Cambridge, 1994.
91. Mauco M. V., Leonardi M. C., Riesco D., Montejano G., Debnath N., "Formalising a Derivation Strategy for Formal Specifications from Natural Language Requirements Models," in proceedings of 2005 IEEE International Symposium on Signal Processing and Information Technology pp 646-651, 2005.
92. McCain R., "A Software Development Methodology for Reusable Components," in Proceedings of the 18th Hawaii Conference on Systems Sciences, Hawaii, January 1985.
93. Mehmet Aksit and Lodewijk Bergmans, "Obstacles in Object-Oriented Software Development", in Proceedings of OOPSLA'92, ACM Press, Canada, pp 341-358, October, 1992.
94. Michael A. Cusumano, "The Software Factory: A Historical Interpretation." IEEE Software, pp 23-30, March 1989.
95. Michels R. and Middendorf M., "An Ant System for the Shortest Common Supersequence Problem", in New Ideas in Optimization, New York, McGraw Hill, pp 51-61, 1999.
96. Mili H., Mili A., Yacoub S., and Addy, "Reuse Based Software Engineering", Wiley-Interscience Publication, USA, 2002.
97. Mili, R. Mili and R.T. Mittermier, "A Survey of Software Reuse Libraries", Annals of Software Engineering no. 5, pp 349-414, 1998.

98. Modha D. S., Spangler W. S., "Clustering Hypertext with Applications to Web Searching", in proceedings of the ACM Hypertext Conference, San Antonio, TX, May-June 2000.
99. Mohagheghi P. and Couradi R., "Experiences with Certification of Reusable Components in the GSN project in Ericsson, Norway", 4th ICSE workshop on Component-Based Software Engineering, Tronto, Canada, May, 2001.
100. Moineau T. and Gaudel M. C., "Software Reusability through formal Specifications," in proceedings of the First International Workshop on Software Reusability, Universitaet Dortmund, 1991.
101. Neighbors J. M., "The DRACO Approach to Constructing Software from Reusable Components," IEEE Transactions on Software Engineering, pp 564-574, September 1984.
102. Neil A. Maiden and Alistair G. Sutcliffe, "People-Oriented Software Reuse: the Very Thought," in proceedings of the Second International Workshop on Software Reuse, ed. William B. Frakes, pp. 176-185, IEEE Computer Press Lucca, Italy, March 24-26, 1993.
103. Oliver I., Smith D. and Holland J.R., "A Study of Permutation Crossover Operators on the Travelling Salesman Problem", in proceedings of the Second International Conference on Genetic Algorithms, New Jersey, pp 224-230, 1987.
104. P. Devambu et al. "LASSIE: A Knowledge-Based Software Information System", Communications of the ACM, vol. 34, no. 5, pp 36-49, 1991.
105. Pamela Zave and William Schell, "Salient features of an executable specification language and its environment," IEEE Transactions on Software Engineering, vol. 12, no. 2, pp 312-325, February 1986.
106. Parepinelli R. S., Lopes H.S., and Freitas A., "An Ant Colony Algorithm for Classification Rule Discovery in Data Mining: A Heuristic Approach", Idea Group Publishing London, pp 191-208, 2002.
107. Parpinelli R. S., Lopes H. S., and Freitas A. A., "Data Mining with an Ant Colony Optimization Algorithm", IEEE Transactions on Evolutionary Computing, vol. 6, no. 4, pp 321-332, 2002.
108. Patrick Schicheng Chen, Rolf Hennicker, and Matthias Jarke, "On The Retrieval of reusable Components," in Advances in Software Reuse, IEEE Computer Society Press, Lucca, Italy, pp. 99-108, March 1993.

109. Peter Wegner, "Varieties of Reusability," in Tutorial: Software Reusability, ed. Peter Freeman, pp 24-38, 1987.
110. Praveen Pathak, Michael Gordon, Weiguo Fan, "Effective Information Retrieval Using genetic Algorithms based matching functions adaptation", in proceedings of the 33rd Hawaii International Conference on system sciences, pp 1-8, 2000.
111. Prodromidis A., Chan P. and Stolfo S., "Meta-learning in Distributed Data Mining Systems: Issues and Approaches", in Advances in Distributed and Parallel Knowledge Discovery, AAAI/MIT Press, Cambridge, MA, 2000.
112. Qian Y. and Suen C. Y., "Clustering Combination Methods", in proceedings of IEEE International Conference on Pattern Recognition (ICPR '00), Barcelona, Spain, vol. 2, pp 732-735, September 2000.
113. Qian Y., Suen C.Y., Tang Y., "Sequential Combination Methods for Data Clustering Analysis", Journal of Computer Science and Technology, vol. 17 no. 2, pp 118-128, 2002.
114. Rao Vijay D., Sarma V.V.S., "A Rough-Fuzzy Approach for Retrieval of Candidate Components for Software Reuse", Pattern recognition letters 24, pp 875-886, 2003.
115. Ravichandran T. and Marcus A. Rothenberger, "Software Reuse Strategies and Component Markets", ACM Transactions on Software Engineering, vol. 46 no. 8, pp 109-114, August 2003.
116. Reitz M. "Software Evolvability by Component-Orientation A Loosely Coupled Component Model Inspired by Biology", in proceedings of Second International IEEE Workshop on Software Evolvability (SE'06), 2006.
117. Reitz M. and Nogel U., "Components: A Valuable Investment for Financial Engineering", in proceedings of the International Conference on Principles and Practices of Programming in Java, 2006.
118. Reitz M. and Stenzel C., "Minerva: A Component-Based Framework for Active Documents", in proceedings of the Software Composition Workshop (SC'04), no. 114 in ENTCS, Elsevier, 2005.
119. Reitz M., "Active Documents", in proceedings of the Workshop on Component Oriented Programming (WCOP), 2006.
120. Richard F. Paige and Jonathan S. Ostro , “A Comparison of the Business Object Notation and the Unified Modeling Language”, Technical Report CS-

1999-04, Department of Computer Science 4700 Keele Street North York, Ontario M3J 1P3 Canada, May 1999.

121. Robert G. Lanergan and Charles A. Grasso, "Software Engineering with Reusable Designs and Code," IEEE Transactions on Software Engineering, vol. 10 no. 5, pp 498-501, September 1984.
122. Robert J. Hall, "Generalized Behavior-based Retrieval," in Proceedings of the 15th International Conference on Software Engineering, ACM Press, Baltimore, Maryland, pp 371-380, May, 1993.
123. Ruben Prieto-Diaz and Peter Freeman, "Classifying Software for Reusability," IEEE Software, pp. 6-16, January. 1987.
124. Rubén Prieto-Díaz, "Software Reuse: Issues and Experiences", American Programmer vol.6, no.8, pp 10-18, April 1993.
125. Ruben Prieto-Diaz, "Domain Analysis for Reusability," in Proceedings of COMPSAC'87, pp. 23-29, IEEE Press, 1987.
126. Salton G. Buckley C., "Term-weighting approaches in automatic text retrieval", International Journal of Information Processing and Management, vol. 24, no. 5, pp 513-523, 1988.
127. Salton G., "Automatic Text Processing: The Transformation, Analysis, and Retrieval of Information by Computer", Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 1989.
128. Salton G., McGill M J, "Introduction to Modern Information Retrieval", McGraw-Hill, Inc. New York, NY, USA, 1986.
129. Scott Henninger, "An Evolutionary Approach to constructing Effective Software Reuse Repositories", ACM Transaction on Software Engineering and Methodology, vol. 6, no. 2, pp 111-140, April 1997.
130. Scott N. Woodfield, David W. Embley, and Del T. Scott, "Can Programmers Reuse Software," IEEE Software, pp 152-59, July 1987.
131. Sharma A. K., Moinuddin and Gupta J. P., "A Novel Representation of Rule Base for Fuzzy Expert System", Journal of Computer Science and Informations, CSI (India) Bombay, vol. 27, no. 4, pp 125-129, Dec.1997.
132. Sharma P., Awasthi Lalit K., Sharma R. K., and Govil M. C., “Region Growing Approach for Image Segmentation”, International Journal of Information and Computing Science, vol.8, no.2, pp 1-7, December 2005.

133. Shayoying Liu and Rolf Adams, "Limitations of Formal Methods and an Approach to Improvement", Invited Lecture in Department of Computer Science, Faculty of Information Sciences, Hiroshima City University, Hiroshima, Japan, 1995.
134. Shetty T. N. R., Riccio P. M., Quinqueton J., "Hybrid method for knowledge processing, integration and representation", Proceedings of the 2006 IEEE International Conference on Information Reuse and Integration, IRI - 2006, Waikoloa, Hawaii, USA, pp 45-50, 2006.
135. Shukla S. B., Ramakrishnan V. C., and Agrawal D. P., “A Real-Time Component Labeling Algorithm and its Architectural Mapping”, invited paper, Frontiers in Parallel Computing, Bhatkar et al, editors, Narosa Publishing House, pp 263-269, 1991.
136. Singh Harpreet, Anneberg L., Kaushal R., and Chamas N.,"Determination of software reliability," 21st Pittsburgh conference on modeling and simulation, May, 1990.
137. Smaragdakis Y. and Batory D., "An Object-Oriented Implementation Technologies for Refinement and Collaboration-Based Designs", ACM Transactions on Software Engineering and Methodolgy, vol. 11 no. 2, pp 215-255, April 2002.
138. Spivey J. M., ”The Z Notation: A Reference Manual”, Prentice Hall, International Series in Computer Science, 2001.
139. Steinbach M. and Kumar V., "Generalizing the Notion of Confidence" , Knowledge and Information Systems, Published online, October 3, 2006.
140. Steinbach M. Karypis G. and Kumar V., "A Comparison of Document Clustering Techniques", KDD-2000 Workshop on Text Mining, August 2000.
141. Stephen J. Garland, John V. Guttag, and James J. Horning, ``An overview of Larch: Functional Programming, Concurrency, Simulation and Automated Reasoning", Peter E. Lauer (editor), Springer-Verlag Lecture Notes in Computer Science, vol. 693, pp 329-348, July 1993.
142. Strehl A. and Ghosh J., "Cluster Ensembles - A Knowledge Reuse Framework for Combining Multiple Partitions", in Journal of Machine Learning Research, no. 3, pp 583-617, 2002.

143. Strehl A., Ghosh J. and Mooney R., "Impact of Similarity Measure on Web-Page Clustering", in proceedings AAAI Workshop on AI for Web Search, Austin, pp 58-64, 2000.
144. Stutzle T., and Dorigo M., "A short convergence proof for a class of ACO algorithms", IEEE Transactions on Evolutionary Computation, vol. 6, no.4, pp 358-365, 2002.
145. Stutzle T., and Dorigo M., "ACO Algorithm for Travelling Salesman Problem", in Evolutionary Algorithms in Engineering and Computer Science, Wiley New York, pp 163-183, 1991.
146. Susan P. Arnold and Stephen L. Stepoway, "The Reuse System: Cataloguing and Retrieval of Reusable Software", in Proceedings of COMPCONS'87, pp 376-379, IEEE Computer Society Press, 1987.
147. Tangsripairoj S. and Samadzadeh M. H., "A Survey of Data Mining Technology Applied to Software Reuse", Proceedings of International Conference on Software Engineering Research and Practices, pp 847-853, June 2003.
148. Tapaswi Shashikala, Joshi R.C., "An Evolutionary Computing Approach for Mining of Bio-medical Images", in proceedings of Advances in Multimedia Information Processing - PCM 2004. 5th Pacific Rim Conference on Multimedia, Tokyo, Japan, pp 523-532, 2004.
149. Ted Davis. "The Reuse Capability Model: A Basis for Improving an Organization's Reuse Capability," in Advances in Software Reuse, pp.126-133, IEEE Computer Society Press Lucca, Italy, March 24-26, 1993.
150. Tomasz G. Smolinski, Grzegorz M. Boratyn, Mariofanna G. Milanova, Roger Buchanan, Astrid A. Prinz "Hybridization of Independent Component Analysis, Rough Sets, and Multi-Objective Evolutionary Algorithms for Classificatory Decomposition of Cortical Evoked Potentials", Pattern Recognition in Bioinformatics (PRIB 2006), International Workshop, Hong Kong, China, Lecture Notes in Computer Science 4146, pp 174-183, Springer 2006.
151. Treharne H., King S., Henson M. C., Steve Schneider, editors. ZB 2005, "Formal Specification and Development in Z and B: Fourth International Conference of B and Z Users", Guildford, UK, Springer 2005.

152. Varadharajan Vijay, "A Formal Approach to System Design and Refinement", In COMPEURO'90, proceedings of the IEEE International Conference on Computer Systems and Software Engineering, 1990, Tel-Aviv, Israel, Los Alamitos, CA, USA: IEEE Computer Society Press, pp 544-545, 1990.
153. Varadharajan Vijay, "A Mathematical Model for System Design and Refinement", in International Journal of Computational Mathematics, vol. 34, no. 1, pp 13-31, 1990.
154. Victor R. Basili and John D. Musa, "The Future Engineering of Software: A Management Perspective", IEEE Computer, vol. 24, no. 9, pp 90-96, September 1991.
155. Vitharana P., Zahedi F. M., and Jain H., "Knowledge-Based Repository Scheme for Storing and Retrieving Business Components", IEEE Transactions on Software Engineering, vol. 29, pp 649 – 664, July 2003.
156. Voas J. M., "The Challenges of Using COTS Software in Component-Based Development", IEEE Computer, vol. 31, no. 6, pp 44, 1998.
157. Waldén Kim, Enea Data, "Business Object Notation (BON)", Published as chapter 10 in "Handbook of Object Technology", CRC Press 1998
158. Warmer J. and Clark T., "Object Modeling with the OCL the Rationale Behind the Object Constraint Language", LNCS 2263, Springer, 2002.
159. Weiguo Fan, Michael D. Gordon, Praveen Pathak, "A Generic Ranking Function Discovery Framework by Genetic Programming for Information Retrieval", Information Processing and Management, pp 587-602, 2004.
160. William B. Frakes and B.A. Nejmeh, "An Information System for Software Reuse", in Software Reuse: Emerging Technology, IEEE Computer Society Press, pp 142-151, 1990.
161. William B. Frakes and Thomas P. Pole, "An Empirical Study of Representation Methods for Reusable Software Components", IEEE Transactions on Software Engineering, vol.20, no.8, pp.617-630, 1994.
162. Wolfgang M., Ritter N., "The ORDB-based SFB-501-Reuse-Repository", in proceedings of 8th International Conference on Extending Database Technology (EDBT'2002), pp 745-748, 2002.
163. Yang Y. and Kamel S. M., "An Aggravated Clustering Approach Using Multi-Ant Colonies Algorithms", in The Journal of Pattern Recognition, Elsevier, no. 39, pp 1278-1289, 2006.

164. Yang Y., Kamel S. M. and Fin J., "Topic Discovery from Document using Ant-Based Clustering Combination", APWeb 2005, 7th Asia-Pacific Web Conference, Shanghai, China, LNCS vol. 3399, pp 100-108, Springer, UK, 2005.
165. Ying P., Lei W., Zhang L., Xie B., Yang F., "Relevancy Based Semantic Interoperation of Reuse Repositories", SIGSOFT FSE, pp 211-220, 2004.
166. Yoelle S. Maarek, Daniel M. Berry, and Gail E. Kaiser, "GURU: Information Retrieval for Reuse, Landmark Contributions in Software Reuse and Reverse Engineering" edited by P. Hall, Unicom Seminars Ltd., 1994.
167. Yoelle S. Maarek, Daniel M. Berry, and Gail E. Kaiser, "An Information Retrieval Approach for Automatically Constructing Software Libraries," IEEE Transactions on Software Engineering, vol. 17, no. 8, pp 800-813, August 1991.
168. Yonghao Chen and Betty Cheng, “Formalizing and Automating Component Reuse", in proceedings of IEEE International Conference on Tools with Artificial Intelligence, pp 94-101, November 1997.
169. Yonghao Chen and Betty Cheng, “Formally Specifying and Analyzing Architectural and Functional Properties of Components for Reuse", in proceedings of Workshop for Software Reuse, March 1997.
170. Zündorf B. Schulz H. Mayr K., "SHORE – A Hypertext Repository in the XML World", SD&M Corporation, Southfield, MI 48075, USA, 2000.

Appendix A
Population Generations of Genetic Algorithm Based Step

In this appendix all iterations of Case II from chapter 4 are given for reference. From Generation 2 to Generation 29 all populations are given in this appendix.

Generation: 2-------->>

Table A.1 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.40.80.20.20.90.90.40.20.20.20.40.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	6.5
c6	01110110000000000000010010000010	0.40.80.20.20.90.90.40.20.20.20.40.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	6.5
cc5	00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.20.20.40.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	6.5
cc6	00010101000110001011100100011100	0.40.80.20.20.90.90.40.20.20.20.40.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	6.5
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
a6	00001010001111110101010101100001	0.40.90.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
ad5	11110010011001110100101010000010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	8.7
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	8.7
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	8.7
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	8.7
aa5	10001001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	8.7
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	8.7
as5	11101010110110000010100001110010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	9.3
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5

Table A.2 Population after Crossover

Crossover Point:: 12

Comp Name	AV	WV	Mate1 Mate2
c5	00001010001111110101010101100001	0.40.80.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	6.5 12.5 1 17

c6	01110110000000000000010010000010	0.40.80.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	6.5 12.5 2 17
cc5	00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.20.20.40.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	6.5 6.5 3 2
cc6	00010101000110001011100100011100	0.40.80.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	6.5 12.5 4 14
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8 8 5 5
a6	00001010001111110101010101100001	0.40.90.20.20.90.90.40.20.20.20.90.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8 12.5 6 14
ad5	11110010011001110100101010000010	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8.7 8 7 6
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.7 12.5 8 16
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8.7 8 9 6
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8.7 8 10 6
aa5	10001001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.7 12.5 11 17
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.7 12.5 12 14
as5	11101010110110000010100001110010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	9.3 8.7 13 9
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5 9.3 14 13
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	12.5 6.5 15 4
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	12.5 8.7 16 11
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	12.5 8.7 17 11
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5 12.5 18 14

Table A.3 Mutation Table

Bit Position	Old Number	New Number
2	0.8	0.1
9	0.2	0.8
11	0.4	0.6
13	0.5	0.9
16	0.2	0.7
20	0.4	0.9
6	0.6	0.3
18	0.1	0.7
20	0.9	0.3
27	0.7	0.5
21	0.6	0.2
18	0.1	0.3

Table A.4 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.5
c6	01110110000000000000010010000010	0.40.80.20.20.90.90.40.20.80.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	4.1
cc5	00000010111001000000001011110010	0.40.80.20.20.90.90.40.20.20.20.60.20.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	4.3
cc6	00010101000110001011100100011100	0.40.80.20.20.90.90.40.20.20.20.40.20.90.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.4

a5	0011100000100100110010110001 1101	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
a6	0000101000111111010101010110 0001	0.40.90.20.20.90.90.40.20.20.20.90.20.50.70.60.70.40.10.90.90.60.40.90.50.60.60.70.90.60.70.60.4	8.5
ad5	1111001001100111010010101000 0010	0.10.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
ad6	1000000110011101111110101000 0001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
add5	0011110111100000000001011011 1001	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.30.20.40.80.50.60.60.70.90.60.70.60.4	9.1
add6	0100001001001010010010111110 0000	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.90.20.40.80.50.60.60.50.90.60.70.60.4	6.7
aa5	1000100111110001110010101011 1000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.20.40.90.50.60.60.70.90.60.70.60.4	9.1
aa6	1001101110001010101101000 1000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.30.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.7
as5	1110101011011000001010000 1110010	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	9.1
as6	1101000101100111111111011 1001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5
aaa5	1001011001100001100001001 0010011	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.90.20.40.80.40.20.20.20.90.20.20.20.4	5.9
aaa6	0111100101010110111110010 1101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	11.3
af5	1000000111110001110010101 0111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.40.4	9.1
af6	1001101110001010101101000 1000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.7

Average Fitness::8.05

==========

Best Component	Max Fitness
as6	12.5
aaa6	11.3
as5	9.1

Average Fitness: 8.05

****

Generation: 3-------->>

Table A.5 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	0000101000111111010101010110 0001	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.5
c6	0111011000000000000010010000 0010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.5
cc5	0000001011100100000001011110 0010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.5
cc6	0001010100011000101110010 0011100	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.5
a5	0011100000100100110010110 0011101	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
a6	0000101000111111010101010110 0001	0.40.90.20.20.90.90.40.20.20.20.90.20.50.70.60.70.40.10.90.90.60.40.90.50.60.60.70.90.60.70.60.4	8.5
ad5	1111001001100111010010101000 0010	0.40.90.20.20.90.90.40.20.20.20.90.20.50.70.60.70.40.10.90.90.60.40.90.50.60.60.70.90.60.70.60.4	8.5
ad6	1000000110011101111110101000 0001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.90.40.60.40.90.50.60.60.70.90.60.70.60.4	8.9
add5	0011110111100000000001011011 1001	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.30.20.40.80.50.60.60.70.90.60.70.60.4	9.1
add6	0100001001001010010010111110 0000	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.30.20.40.80.50.60.60.70.90.60.70.60.4	9.1
aa5	1000100111110001110010101011 1000	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.30.20.40.80.50.60.60.70.90.60.70.60.4	9.1
aa6	1001101110001010101101000 1000101	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.30.20.40.80.50.60.60.70.90.60.70.60.4	9.1
as5	1110101011011000001010000 1110010	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.30.20.40.80.50.60.60.70.90.60.70.60.4	9.1
as6	1101000101100111111111011 1001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5
aaa5	1001011001100001100001001 0010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5

aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5

Table A.6 Population after Crossover

Crossover Point:: 19

Comp Name	AV	WV	Mate1 Mate2
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.5 9.1 1 9
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.5 8.9 2 8
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.5 12.5 3 14
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.5 9.1 4 10
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8 8 5 5
a6	00001010001111110101010101100001	0.40.90.20.20.90.90.40.20.20.20.90.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	8.5 9.1 6 13
ad5	11110010011001110100101010000010	0.40.90.20.20.90.90.40.20.20.20.90.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	8.5 9.1 7 10
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.90.90.60.40.90.50.60.60.70.90.60.70.60.4	8.9 8.5 8 6
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.30.20.40.80.50.60.60.70.90.60.70.60.4	9.1 9.1 9 9
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1 8.9 10 8
aa5	10001001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1 8.9 11 8
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1 12.5 12 17
as5	11101010110110000010100001110010	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1 7.5 13 2
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12.5 9.1 14 9
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5 12.5 15 15
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5 7.5 16 3
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12.5 9.1 17 13
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	12.5 8.9 18 8

Table A.7 Mutation Table

Bit Position	Old Number	New Number
21	0.2	0.2
32	0.4	0.1
11	0.4	0.9
15	0.6	0.3
8	0.2	0.3
15	0.6	0.6
14	0.7	0.3
19	0.9	0.8
12	0.7	0.3
30	0.7	0.2

6	0.6	0.3
7	0.4	0.2

Table A.8 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.4
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.1	3.4
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.90.20.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	6.7
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.30.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	6.5
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
a6	00001010001111110101010101100001	0.40.90.20.20.90.90.40.20.20.20.90.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.9
ad5	11110010011001110100101010000010	0.40.90.20.20.90.90.40.20.20.20.90.20.50.30.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.1
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.90.60.40.90.50.60.60.70.90.60.70.60.4	9.3
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.30.40.40.20.20.40.90.80.30.20.40.80.50.60.60.70.90.60.70.60.4	9.1
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.20.60.4	7.4
aa5	10001001111100011100101010111000	0.10.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.8
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.20.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	6.9
as5	11101010110110000010100001110010	0.10.90.60.60.40.60.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	11.3
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	8.6
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.40.60.40.90.50.60.60.70.90.60.70.60.4	7.7

Average Fitness::8.08

==========

Best Component	Max Fitness
as6	12
aaa6	11.3
aa5	9.8

Average Fitness: 8.08

****

Generation: 4-------->>

Table A.9 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.4

c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.4
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.4
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.4
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
a6	00001010001111110101010101100001	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
ad5	11110010011001110100101010000010	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.90.60.40.90.50.60.60.70.90.60.70.60.4	9.3
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.90.60.40.90.50.60.60.70.90.60.70.60.4	9.3
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.90.60.40.90.50.60.60.70.90.60.70.60.4	9.3
aa5	10001001111100011100101010111000	0.10.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.8
aa6	10011011100010101011010001000101	0.10.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.8
as5	11101010110110000010100001110010	0.10.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.8
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12

Table A.10 Population after Crossover

Crossover Point:: 32

Comp Name	AV	WV	Mate1 Mate2
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.4 12 1 15
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.4 8 2 5
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.4 7.4 3 3
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.4 7.4 4 3
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8 9.3 5 9
a6	00001010001111110101010101100001	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8 12 6 15
ad5	11110010011001110100101010000010	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	8 9.3 7 10
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.90.60.40.90.50.60.60.70.90.60.70.60.4	9.3 7.4 8 1
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.90.60.40.90.50.60.60.70.90.60.70.60.4	9.3 12 9 14
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.90.60.40.90.50.60.60.70.90.60.70.60.4	9.3 8 10 6
aa5	10001001111100011100101010111000	0.10.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.8 7.4 11 4
aa6	10011011100010101011010001000101	0.10.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.8 12 12 16
as5	11101010110110000010100001110010	0.10.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.8 12 13 15
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12 8 14 5
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12 7.4 15 4
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12 12 16 15
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12 9.3 17 10
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12 12 18 15

Table A.11 Mutation Table

Bit Position	Old Number	New Number
12	0.2	0.6
19	0.9	0.8
1	0.4	0.6
26	0.6	0.6
28	0.9	0.1
25	0.6	0.2
8	0.3	0.2
20	0.9	0.8
19	0.8	0.2
5	0.4	0.2
1	0.1	0.7
4	0.6	0.7

Table A.12 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.8
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.80.30.20.40.80.50.60.60.70.90.60.70.60.4	3.4
cc5	00000010111001000000001011110010	0.60.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	6.1
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.20.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	6.5
a5	00111000001001001100101100011101	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.10.60.70.60.4	7.2
a6	00001010001111110101010101100001	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.20.60.70.90.60.70.60.4	8.5
ad5	11110010011001110100101010000010	0.40.80.20.20.90.90.40.20.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.60.60.70.90.60.70.60.4	7.5
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.20.90.60.40.90.50.60.60.70.90.60.70.60.4	9.1
add6	01000010010010100100101111100000	0.10.90.60.60.20.60.40.60.70.70.80.70.50.70.60.70.40.70.80.90.60.40.90.50.60.60.70.90.60.70.60.4	7.7
aa5	10001001111100011100101010111000	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	10.4
aa6	10011011100010101011010001000101	0.10.90.60.70.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	7.2
as5	11101010110110000010100001110010	0.10.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.1
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	10.8
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	8.6
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.6

Average Fitness::8.11

Best Component Max Fitness
as6 12
aaa6 10.8
aa5 10.4
Average Fitness: 8.11
****

Generation: 5-------->>

Table A.13 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.8
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.8
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.8
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.8
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.8
a6	00001010001111110101010101100001	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.20.60.70.90.60.70.60.4	8.5
ad5	11110010011001110100101010000010	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.90.20.40.80.50.20.60.70.90.60.70.60.4	8.5
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
aa5	10001001111100011100101010111000	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	10.4
aa6	10011011100010101011010001000101	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	10.4
as5	11101010110110000010100001110010	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	10.4
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12

Table A.14 Population after Crossover

Crossover Point:: 16

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.8	7.8 1 1
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	7.8	10.4 2 11
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	7.8	10.4 3 12
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	7.8	10.4 4 11
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.8	9.2 5 9

a6	00001010001111111010101010100001	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	8.5 7.8 6 3
ad5	11110010011001110100101010000010	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	8.5 10.4 7 11
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2 9.2 8 8
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	9.2 7.8 9 5
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	9.2 7.8 10 3
aa5	10001001111100011100101010111000	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	10.4 7.8 11 3
aa6	10011011100010101011010001000101	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	10.4 12 12 15
as5	11101010110110000010100001110010	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	10.4 10.4 13 11
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	12 10.4 14 12
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	12 10.4 15 13
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12 12 16 14
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12 12 17 15
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	12 7.8 18 4

Table A.15 Mutation Table

Bit Position	Old Number	New Number
27	0.7	0.5
12	0.6	0.4
11	0.4	0.2
18	0.9	0.1
9	0.2	0.9
20	0.3	0.9
31	0.6	0.6
21	0.6	0.6
3	0.6	0.5
5	0.4	0.6
15	0.2	0.9
18	0.1	0.9

Table A.16 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.50.90.60.70.60.4	7.6
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.40.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	3.4
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.20.60.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	6
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.80.40.60.40.90.50.60.60.70.90.60.70.60.4	7.3
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.1
a6	00001010001111111010101010100001	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.10.90.90.20.40.80.50.60.60.70.90.60.70.60.4	7.7
ad5	11110010011001110100101010000010	0.40.80.20.20.90.90.40.30.20.20.90.20.40.40.20.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	8

ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
add5	00111101111000000000010110111001	0.10.90.50.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	9
add6	01000010010010100100101111100000	0.10.90.60.60.60.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	6.6
aa5	10001001111100011100101010111000	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.90.20.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	9.1
aa6	10011011100010101011010001000101	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.7
as5	11101010110110000010100001110010	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	7.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	10.8
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	8.6
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.10.90.30.20.40.80.50.60.60.70.90.60.70.60.4	7.6

Average Fitness::8.16

==========

Best Component	Max Fitness
as6	13.2
aaa6	10.8
as5	9.7

Average Fitness: 8.16

****

Generation: 6-------->>

Table A.17 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.50.90.60.70.60.4	7.6
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.50.90.60.70.60.4	7.6
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.50.90.60.70.60.4	7.6
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.50.90.60.70.60.4	7.6
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.1
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.1
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.1
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
aa5	10001001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
as5	11101010110110000010100001110010	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2

Table A.18 Population after Crossover

Crossover Point:: 20

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.6	13.2 1 15
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.50.90.60.70.60.4	7.6	7.6 2 3
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.6	9.2 3 9
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.6	9.2 4 11
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.1	8.1 5 6
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.1	13.2 6 14
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.1	9.2 7 9
ad6	10000001100111011111101010000001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2	9.2 8 9
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2	13.2 9 15
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2	9.2 10 11
aa5	10001001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2	8.1 11 5
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.20.40.80.50.60.60.50.90.60.70.60.4	9.2	7.6 12 3
as5	11101010110110000010100001110010	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.7	9.7 13 13
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2	13.2 14 16
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2	9.2 15 9
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2	13.2 16 15
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2	8.1 17 6
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.20.40.80.50.60.60.50.90.60.70.60.4	13.2	7.6 18 3

Table A.19 Mutation Table

Bit Position	Old Number	New Number
9	0.2	0.4
28	0.9	0.6
13	0.5	0.3
29	0.6	0.3
11	0.4	0.8
16	0.7	0.2
7	0.4	0.4
5	0.4	0.1
8	0.6	0.4
18	0.7	0.2
5	0.4	0.2
31	0.6	0.1

Table A.20 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.8
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.20.40.80.50.60.60.50.60.60.70.60.4	3.4
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.20.20.40.60.30.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	6.2
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.20.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.30.70.60.4	7
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.5
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.40.60.50.70.60.20.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.4
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.3
ad6	10000001100111011111101010000001	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
add5	00111101111000000000010110111001	0.10.90.60.60.40.60.40.40.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.9
add6	01000010010010100100101111100000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.20.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.2
aa5	10001001111100011100101010111000	0.10.90.60.60.20.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.9
aa6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.20.40.80.50.60.60.50.90.60.70.10.4	8
as5	11101010110110000010100001110010	0.70.90.60.60.40.30.40.60.70.70.80.70.40.40.20.20.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	7.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	12
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.9
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.20.40.80.50.60.60.50.90.60.70.60.4	7.6

Average Fitness::8.41

==========

Best Component	Max Fitness
as6	13.2
aaa6	12
aa5	9.9

Average Fitness: 8.41

****

Generation: 7-------->>

Table A.21 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.8
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.8
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.8
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.8
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.5
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.5
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.5

ad6	10000001100111011111101010000001	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
add5	00111101111000000000010110111001	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
add6	01000010010010100100101111100000	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2
aa5	10001001111100011100101010111000	0.10.90.60.60.20.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.9
aa6	10011011100010101011010001000101	0.10.90.60.60.20.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.9
as5	11101010110110000010100001110010	0.10.90.60.60.20.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.9
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	13.2

Table A.22 Population after Crossover

Crossover Point:: 7

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.8	8.5 1 5
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.8	9.2 2 8
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.8	8.5 3 5
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.8	7.8 4 2
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.5	8.5 5 7
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	8.5	13.2 6 15
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.5	8.5 7 5
ad6	10000001100111011111101010000001	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2	9.9 8 13
add5	00111101111000000000010110111001	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.70.60.4	9.2	13.2 9 18
add6	01000010010010100100101111100000	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.2	9.9 10 13
aa5	10001001111100011100101010111000	0.10.90.60.60.20.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.9	9.2 11 10
aa6	10011011100010101011010001000101	0.10.90.60.60.20.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	9.9	7.8 12 1
as5	11101010110110000010100001110010	0.10.90.60.60.20.60.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.9	8.5 13 6
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	13.2	7.8 14 4
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	13.2	8.5 15 5
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	13.2	7.8 16 4
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	13.2	9.9 17 13
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	13.2	9.9 18 11

Table A.23 Mutation Table

Bit Position	Old Number	New Number
13	0.5	0.6
19	0.8	0.3

13	0.5	0.5
13	0.5	0.5
26	0.6	0.2
29	0.6	0.7
26	0.6	0.2
19	0.8	0.6
30	0.7	0.8
20	0.8	0.7
28	0.9	0.3
21	0.6	0.2

Table A.24 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.60.70.70.80.70.50.70.60.70.40.70.30.80.60.40.90.50.60.60.70.90.60.70.60.4	3.4
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.3
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	7.3
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.20.70.90.60.70.60.4	8.5
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.70.70.60.4	9.2
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.20.70.90.60.70.60.4	7.7
ad6	10000001100111011111101010000001	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.70.60.80.60.40.90.50.60.60.70.90.60.70.60.4	9
add5	00111101111000000000010110111001	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.90.80.40.60.40.90.50.60.60.70.90.60.80.60.4	8.8
add6	01000010010010100100101111100000	0.10.90.60.60.10.60.40.60.70.70.80.70.50.70.60.70.40.70.80.70.60.40.90.50.60.60.70.90.60.70.60.4	7.7
aa5	10001001111100011100101010111000	0.10.90.60.60.20.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.30.60.70.60.4	9.3
aa6	10011011100010101011010001000101	0.10.90.60.60.20.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.20.40.90.50.60.60.70.90.60.70.60.4	6.7
as5	11101010110110000010100001110010	0.10.90.60.60.20.60.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.6
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	6.7
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	10.2
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.7
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.60.70.70.80.70.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8

Average Fitness::8.26

==========

Best Component	Max Fitness
as6	11.1
aaa6	10.2
af5	9.7

Average Fitness: 8.26

****

Generation: 8-------->>

Table A.25 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
ad6	10000001100111011111101010000001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
add6	01000010010010100100101111100000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1

Table A.26 Population after Crossover

Crossover Point:: 18

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 1 4
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 2 2
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 3 3
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 4 11
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 5 10
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 6 8
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 7 2
ad6	10000001100111011111101010000001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.90.30.60.40.90.50.60.60.70.90.60.70.60.4	9.4	11.1 8 18
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.90.30.60.40.90.50.60.60.70.90.60.70.60.4	9.4	11.1 9 16
add6	01000010010010100100101111100000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 10 12
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 11 12
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4	9.4 12 13
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.90.30.60.40.90.50.60.60.70.90.60.70.60.4	9.4	11.1 13 15

as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1 11.1 14 16
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.1 9.4 15 2
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.1 9.4 16 5
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1 11.1 17 14
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.1 9.4 18 11

Table A.27 Mutation Table

Bit Position	Old Number	New Number
14	0.7	0.9
28	0.9	0.3
28	0.9	0.5
31	0.6	0.1
16	0.7	0.8
2	0.1	0.8
8	0.2	0.7
26	0.6	0.5
4	0.2	0.2
11	0.8	0.3
21	0.6	0.1
14	0.7	0.8

Table A.28 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.30.60.70.60.4	3.4
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.50.60.70.60.4	6.9
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.10.4	7.8
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.80.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.5
a6	00001010001111110101010101100001	0.40.80.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.4
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.70.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.7
ad6	10000001100111011111101010000001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.90.30.60.40.90.50.60.50.70.90.60.70.60.4	8.9
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.90.30.60.40.90.50.60.60.70.90.60.70.60.4	8.4
add6	01000010010010100100101111100000	0.40.10.20.20.90.90.40.20.90.20.30.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	6.5
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.10.40.90.50.60.60.70.90.60.70.60.4	9.6
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.3
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.90.30.60.40.90.50.60.60.70.90.60.70.60.4	8.6
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.80.80.60.40.90.50.60.60.70.90.60.70.60.4	6.3
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.80.80.60.40.90.50.60.60.70.90.60.70.60.4	10.6

af5 1000000111110001110010101011000 0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4 7.4
af6 1001101110001010101110100010001 01 0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.80.80.60.40.90.50.60.60.70.90.60.70.60.4 7.3
Average Fitness::8.13
==========

Best Component Max Fitness
as6 11.1
aaa6 10.6
c5 9.6
Average Fitness: 8.13
****

Generation: 9-------->>

Table A.29 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	0000101000111111010101010110000 1	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
c6	0111011000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
cc5	0000001011100100000000101111001 0	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
cc6	0001010100011000101110010001110 0	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
a5	0011100000100100110010110001110 1	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
a6	0000101000111111010101010110000 1	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
ad5	1111001001100111010010101000001 0	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
ad6	1000000110011101111110101000000 1	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
add5	0011110111100000000001011011100 1	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
add6	0100001001001010010010111110000 0	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
aa5	1000100111110001110010101011100 0	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
aa6	1001101110001010101101000100010 1	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
as5	1110101011011000001010000111001 0	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6
as6	1101000101100111111111011100111 1	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
aaa5	1001011001100001100001001001001 1	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
aaa6	0111100101010110111110010110110 0	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
af5	1000000111110001110010101011100 0	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
af6	1001101110001010101110100010001 01	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1

Table A.30 Population after Crossover

Crossover Point:: 17

Comp Name	AV	WV	Mate1	Mate2
c5	0000101000111111010101010110000 1	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6	1 11
c6	0111011000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6	2 13

cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 3 7
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 4 6
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 5 11
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 6 9
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 7 4
ad6	10000001100111011111101010000001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 8 13
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	9.6 11.1 9 17
add6	01000010010010100100101111100000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 10 6
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 11 13
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 12 10
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6 9.6 13 12
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1 11.1 14 17
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.1 9.6 15 5
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.1 9.6 16 6
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.1 9.6 17 6
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.1 9.6 18 3

Table A.31 Mutation Table

Bit Position	Old Number	New Number
31	0.6	0.2
30	0.7	0.5
3	0.2	0.9
18	0.7	0.2
17	0.4	0.5
27	0.7	0.3
18	0.7	0.4
28	0.9	0.6
28	0.9	0.4
14	0.9	0.8
27	0.7	0.8
8	0.2	0.7

Table A.32 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.50.60.4	3.4
cc5	00000010111001000000001011110010	0.40.10.90.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.5
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.20.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.8

a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.50.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.8
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.30.90.60.70.60.4	9.2
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.40.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.6
ad6	10000001100111011111101010000001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.60.60.70.60.4	9.5
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.10.90.30.60.40.90.50.60.60.70.40.60.70.60.4	7.9
add6	01000010010010100100101111100000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	6.5
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.70.60.4	10.2
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.70.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.8
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8.5
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	6.3
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.3

Average Fitness::8.29

==========

Best Component	Max Fitness
aaa6	11.2
as6	11.1
aa5	10.2

Average Fitness: 8.29

****

Generation: 10-------->>

Table A.33 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
ad6	10000001100111011111101010000001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
add6	01000010010010100100101111100000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.70.60.4	10.2
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.70.60.4	10.2
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.70.60.4	10.2
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.60.4	11.1
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2

af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2

Table A.34 Population after Crossover

Crossover Point:: 30

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6	9.6 1 7
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6	11.1 2 14
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6	10.2 3 11
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6	9.6 4 10
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6	9.6 5 2
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6	9.6 6 4
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6	10.2 7 13
ad6	10000001100111011111101010000001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6	11.2 8 18
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6	9.6 9 7
add6	01000010010010100100101111100000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.6	11.1 10 15
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.70.20.4	10.2	9.6 11 3
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.70.20.4	10.2	9.6 12 10
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.70.20.4	10.2	9.6 13 2
as6	11010001011001111111111011001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	11.1	9.6 14 1
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	11.1	9.6 15 9
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2	10.2 16 13
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	11.2	9.6 17 1
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	11.2	9.6 18 1

Table A.35 Mutation Table

Bit Position	Old Number	New Number
12	0.6	0.7
11	0.8	0.5
17	0.4	0.1
20	0.8	0.2
12	0.6	0.4
3	0.2	0.6
31	0.6	0.4
32	0.4	0.3
30	0.7	0.9
15	0.6	0.3
30	0.7	0.8
6	0.9	0.7

Table A.36 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.50.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	3.4
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.10.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	7.5
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.20.60.40.90.50.60.60.70.90.60.70.20.4	7.2
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	8.7
a6	00001010001111110101010101100001	0.40.10.60.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.6
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.40.4	7.7
ad6	10000001100111011111101010000001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.3	9.4
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.90.20.4	8.4
add6	01000010010010100100101111100000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.30.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	6.2
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	10.2
aa6	10011011100010101011010001000101	0.40.10.20.20.90.70.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.70.20.4	8.3
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.70.20.4	8.2
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	5.9
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	8
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	7.3

Average Fitness::8.2

==========

Best Component	Max Fitness
aaa6	11.2
as6	10.7
aa5	10.2

Average Fitness: 8.2

****

Generation: 11-------->>

Table A.37 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
c6	01110110000000000000010010000010	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
cc6	00010101000110001011100100011100	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7

a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
ad5	11110010011001110100101010000010	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
ad6	10000001100111011111101010000001	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
add6	01000010010010100100101111100000	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	10.2
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	10.2
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	10.2
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2

Table A.38 Population after Crossover

Crossover Point:: 4

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7	10.2 1 12
c6	01110110000000000000010010000010	0.40.10.20.20.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.7	11.2 2 18
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7	9.7 3 8
cc6	00010101000110001011100100011100	0.40.10.20.20.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	9.7	10.7 4 14
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7	9.7 5 4
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7	9.7 6 8
ad5	11110010011001110100101010000010	0.40.10.20.20.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.7	11.2 7 17
ad6	10000001100111011111101010000001	0.40.10.20.20.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.7	11.2 8 17
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7	9.7 9 9
add6	01000010010010100100101111100000	0.40.10.20.20.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.7	11.2 10 18
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	10.2	9.7 11 6
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	10.2	9.7 12 3
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	10.2	10.2 13 11
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7	10.7 14 15
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	10.7	9.7 15 6
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2	11.2 16 16
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	11.2	10.2 17 12
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	11.2	9.7 18 7

Table A.39 Mutation Table

Bit Position	Old Number	New Number
1	0.4	0.9
3	0.2	0.7
16	0.7	0.3
19	0.9	0.3
17	0.4	0.7
2	0.1	0.1
29	0.6	0.9
29	0.6	0.4
20	0.8	0.7
15	0.6	0.7
27	0.7	0.1
30	0.7	0.2

Table A.40 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
c6	01110110000000000000010010000010	0.40.10.70.20.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	3.6
cc5	00000010111001000000001011110010	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.30.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	7.1
cc6	00010101000110001011100100011100	0.40.10.20.20.40.60.40.20.40.20.40.60.50.70.60.70.40.10.30.30.60.40.90.50.60.60.70.90.60.70.20.4	6.4
a5	00111000001001001100101100011101	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.70.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9
a6	00001010001111110101010101100001	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.7
ad5	11110010011001110100101010000010	0.40.10.20.20.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.90.70.60.4	7.3
ad6	10000001100111011111101010000001	0.40.10.20.20.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.40.70.60.4	8.7
add5	00111101111000000000010110111001	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.70.60.40.90.50.60.60.70.90.60.70.20.4	8.4
add6	01000010010010100100101111100000	0.40.10.20.20.40.60.40.20.40.20.40.60.50.70.70.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	6.5
aa5	10001001111100011100101010111000	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.10.90.60.70.20.4	9.6
aa6	10011011100010101011010001000101	0.40.10.20.20.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.20.20.4	7.8
as5	11101010110110000010100001110010	0.40.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	8.2
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	6.6
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.40.20.90.20.80.70.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	8.4

Average Fitness::8.22

Best Component	Max Fitness
aaa6	11.2

as6 10.7
c5 9.7
Average Fitness: 8.22

****

Generation: 12-------->>

Table A.41 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111111010101010100001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
c6	01110110000000000000010010000010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
cc5	00000010111001000000001011110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
cc6	00010101000110001011100100011100	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
a5	00111000001001001100101100011101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
a6	00001010001111111010101010100001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
ad5	11110010011001110100101010000010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2

Table A.42 Population after Crossover

Crossover Point:: 13

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111111010101010100001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7	9.7 1 7
c6	01110110000000000000010010000010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	9.7	10.7 2 14
cc5	00000010111001000000001011110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7	9.7 3 2
cc6	00010101000110001011100100011100	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7	9.7 4 8
a5	00111000001001001100101100011101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.7	11.2 5 16
a6	00001010001111111010101010100001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7	9.7 6 4

ad5	11110010011001110100101010000010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7 9.7 7 3
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7 9.7 8 11
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7 9.7 9 11
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	9.7 11.2 10 16
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	9.7 10.7 11 14
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7 9.7 12 11
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	9.7 10.7 13 14
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7 10.7 14 14
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7 10.7 15 15
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	11.2 9.7 16 7
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	11.2 11.2 17 18
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	11.2 9.7 18 3

Table A.43 Mutation Table

Bit Position	Old Number	New Number
4	0.2	0.7
17	0.4	0.8
19	0.8	0.2
10	0.2	0.8
27	0.7	0.9
32	0.4	0.9
27	0.8	0.8
10	0.2	0.1
27	0.8	0.2
15	0.6	0.1
18	0.1	0.2
8	0.2	0.6

Table A.44 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
c6	01110110000000000000010010000010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.80.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	3
cc5	00000010111001000000001011110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.20.80.60.40.90.50.60.60.80.90.60.80.20.4	7.2
cc6	00010101000110001011100100011100	0.90.10.20.20.90.90.40.20.90.80.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	7.9
a5	00111000001001001100101100011101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.70.80.80.60.40.90.50.60.60.90.90.60.70.60.4	8.5
a6	00001010001111110101010101100001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2
ad5	11110010011001110100101010000010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	8
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.10.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	10

add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.20.90.60.80.20.4	7.9
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.10.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	6
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.20.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.1
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.60.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.3
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	8.7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	5.9
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	11.6
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.60.4	8
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	7.4

Average Fitness::8.34

==========

Best Component	Max Fitness
aaa6	11.6
as6	10.7
a6	10.2

Average Fitness: 8.34

****

Generation: 13-------->>

Table A.45 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
a6	00001010001111110101010101100001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2
ad5	11110010011001110100101010000010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.70.90.60.70.20.4	10.7
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	11.6
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	11.6
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	11.6

Table A.46 Population after Crossover

Crossover Point:: 22

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7	9.7 1 5
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7	9.7 2 4
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.7	10.2 3 11
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.7	10.2 4 8
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.7	10.2 5 12
a6	00001010001111110101010101100001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	10.2	10.7 6 14
ad5	11110010011001110100101010000010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2	10.2 7 11
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2	10.2 8 9
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2	10.2 9 13
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	10.2	9.7 10 3
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	10.2	9.7 11 2
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2	10.2 12 11
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	10.2	10.7 13 14
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	10.7	10.2 14 8
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	10.7	10.2 15 7
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6	10.2 16 12
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	11.6	9.7 17 5
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	11.6	9.7 18 4

Table A.47 Mutation Table

Bit Position	Old Number	New Number
19	0.8	0.6
27	0.8	0.6
2	0.1	0.2
12	0.6	0.2
8	0.2	0.4
13	0.6	0.3
19	0.8	0.7
24	0.5	0.5
31	0.2	0.9
12	0.6	0.2
29	0.6	0.1
19	0.8	0.2

Table A.48 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.60.90.60.80.20.4	3.5
cc5	00000010111001000000001011110010	0.90.20.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	7.2
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.90.20.80.20.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	8
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.8
a6	00001010001111110101010101100001	0.90.10.20.20.90.90.40.20.90.20.80.60.30.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	9.3
ad5	11110010011001110100101010000010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.70.80.60.40.90.50.60.60.80.90.60.80.20.9	8
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.90.9	9
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.20.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	6.6
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.10.80.20.4	10.2
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.20.80.60.40.90.50.60.60.80.90.60.80.20.9	8.8
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.70.90.60.70.20.4	8.6
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	11.3
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	6.4
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	8.1
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	7.4

Average Fitness::8.56

==========

Best Component	Max Fitness
aaa6	11.6
as6	11.3
ad6	10.5

Average Fitness: 8.56

****

Generation: 14-------->>

Table A.49 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.8
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.8

ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.8
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	11.3
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	11.3
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6

Table A.50 Population after Crossover

Crossover Point:: 25

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7	9.7 1 4
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.9	9.7	9.8 2 5
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.9	9.7	10.5 3 8
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.4	9.7	9.7 4 4
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.8	9.8 5 7
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.8	9.8 6 7
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.8	9.7 7 4
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5	10.5 8 13
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5	10.5 9 10
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5	10.5 10 12
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5	11.3 11 15
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5	9.8 12 6
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5	9.8 13 7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	11.3	10.5 14 10
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.4	11.3	9.7 15 4
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6	9.8 16 7
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6	9.8 17 5
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6	9.8 18 5

Table A.51 Mutation Table

Bit Position	Old Number	New Number
23	0.9	0.3
21	0.6	0.3
22	0.4	0.9
4	0.7	0.7
15	0.6	0.2
31	0.2	0.7
2	0.1	0.7
18	0.7	0.5
1	0.9	0.7
25	0.6	0.8
14	0.9	0.8
24	0.5	0.6

Table A.52 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	9.7
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.30.40.90.50.60.60.80.90.60.80.20.9	3.5
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.90.90.50.60.60.80.90.60.80.20.9	7.2
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.4	8.2
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.8
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.9	10.2
ad5	11110010011001110100101010000010	0.90.70.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.4	9.1
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.3
add5	00111101111000000000010110111001	0.70.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.80.60.80.90.60.80.20.9	6.8
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.7
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.60.60.60.80.90.60.80.20.9	9.4
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	8.7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	11.3
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.4	5.9
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	8.1
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	7.9

Average Fitness::8.74

Best Component	Max Fitness
aaa6	11.6

as6 11.3
aa5 10.7
Average Fitness: 8.74
****

Generation: 15-------->>

Table A.53 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111111010101010110000 1	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	9.7
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	9.7
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	9.7
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	9.7
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.8
a6	00001010001111111010101010110000 1	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.9	10.2
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.9	10.2
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.3
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.3
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.3
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.7
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.7
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	11.3
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	11.3
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6

Table A.54 Population after Crossover

Crossover Point:: 11

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111111010101010110000 1	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.7	10.3 1 10
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.7	10.3 2 8
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.7	10.7 3 12
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.9	9.7	10.2 4 6
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.8	10.3 5 9
a6	00001010001111111010101010110000 1	0.90.10.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2	10.3 6 9
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.40.90.20.80.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.2	11.6 7 17
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.3	9.8 8 5

add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.3 10.7 9 11
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.3 10.3 10 8
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.9	10.7 10.2 11 6
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	10.7 9.7 12 3
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.7 10.3 13 10
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	11.3 9.7 14 2
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.3 10.3 15 8
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	11.6 11.3 16 15
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.40.20.40.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	11.6 10.7 17 12
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	11.6 11.3 18 14

Table A.55 Mutation Table

Bit Position	Old Number	New Number
21	0.6	0.4
14	0.9	0.5
11	0.8	0.9
2	0.1	0.4
9	0.9	0.1
2	0.1	0.5
28	0.9	0.7
11	0.8	0.6
3	0.2	0.8
17	0.4	0.4
32	0.9	0.5
12	0.6	0.2

Table A.56 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.50.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	3.5
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.90.20.90.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	7.2
cc6	00010101000110001011100100011100	0.90.40.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.9	8.4
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.40.10.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.6
a6	00001010001111110101010101100001	0.90.50.20.70.90.90.40.40.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.40.90.20.80.60.50.90.60.70.40.70.80.80.60.40.90.50.60.60.80.70.60.80.20.9	8.5
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
add5	00111101111000000000010110111001	0.90.10.80.20.90.90.40.20.90.20.80.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.6
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	6.4
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.5	10.7

aa6	1001101110001010101101000100010 1	0.90.10.20.20.90.90.40.20.90.20.80.20.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	8.7
as5	1110101011011000001010000111001 0	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	8.7
as6	1101000101100111111111011100111 1	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	11.8
aaa5	1001011001100001100001001001001 1	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.60.80.20.9	6.4
aaa6	0111100101010110111110010110110 0	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	10.4
af5	1000000111110001110010101011100 0	0.10.90.60.60.40.60.40.20.40.20.40.60.60.80.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	8.1
af6	1001101110001010101101000100010 1	0.10.90.60.60.40.60.40.20.40.20.40.60.50.70.60.70.40.10.90.30.60.40.90.50.60.60.80.90.60.80.20.9	7.5

Average Fitness::8.67

==========

Best Component	Max Fitness
as6	11.8
aa5	10.7
ad6	10.5

Average Fitness: 8.67

****

Generation: 16-------->>

Table A.57 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	0000101000111111010101010110000 1	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10
c6	0111011000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10
cc5	0000001011100100000001011110010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10
cc6	0001010100011000101110010001 1100	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10
a5	0011100000100100110010110001 1101	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10
a6	0000101000111111010101010110000 1	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10
ad5	1111001001100111010010101000001 0	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10
ad6	1000000110011101111110101000000 1	0.90.10.20.20.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
add5	0011110111100000000001011011100 1	0.90.10.20.20.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
add6	0100001001001010010010111110000 0	0.90.10.20.20.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
aa5	1000100111110001110010101011100 0	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.5	10.7
aa6	1001101110001010101101000100010 1	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.5	10.7
as5	1110101011011000001010000111001 0	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.5	10.7
as6	1101000101100111111111011100111 1	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	11.8
aaa5	1001011001100001100001001001001 1	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	11.8
aaa6	0111100101010110111110010110110 0	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	11.8
af5	1000000111110001110010101011100 0	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	11.8
af6	1001101110001010101101000100010 1	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	11.8

Table A.58 Population after Crossover

Crossover Point:: 8

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	10	11.8 1 18
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10	10 2 2
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10	10 3 3
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10	10 4 2
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	10	11.8 5 18
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10	10.5 6 9
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10	10 7 2
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10.5	10 8 5
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10.5	10 9 5
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	10.5	11.8 10 17
aa5	10001001111100011100101010111000	0.90.10.20.20.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.7	10.5 11 9
aa6	10011011100010101011010001000101	0.90.10.20.20.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	10.7	11.8 12 14
as5	11101010110110000010100001110010	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	10.7	10 13 2
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	11.8	10 14 3
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	11.8	11.8 15 15
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	11.8	10 16 1
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.5	11.8	10.7 17 13
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	11.8	10 18 7

Table A.59 Mutation Table

Bit Position	Old Number	New Number
29	0.6	0.8
16	0.7	0.4
28	0.9	0.9
17	0.4	0.8
19	0.6	0.1
12	0.6	0.5
31	0.2	0.9
12	0.6	0.4
22	0.4	0.6
25	0.6	0.6
4	0.2	0.4
12	0.6	0.5

Table A.60 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	0000101000111111010101010110000 1	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
c6	0111011000000000000000010010000010	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.40.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	3.5
cc5	0000001011100100000000010111100 10	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	7.2
cc6	0001010100011000101110010001110 0	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.80.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	8.6
a5	0011100000100100110010110001110 1	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.10.80.60.40.30.50.60.60.80.90.60.80.20.4	8.3
a6	0000101000111111010101010110000 1	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.5
ad5	1111001001100111010010101000001 0	0.90.10.20.70.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.90.9	8.8
ad6	1000000110011101111110101000000 1	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	9.9
add5	0011110111100000000001011011100 1	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.60.90.50.60.60.80.90.60.80.20.9	9.2
add6	0100001001001010010010111110000 0	0.90.10.20.20.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	6
aa5	1000100111110001110010101011100 0	0.90.10.20.40.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
aa6	1001101110001010101101000100010 1	0.90.10.20.20.90.90.40.20.40.20.40.50.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	8.2
as5	1110101011011000001010000111001 0	0.90.10.20.20.90.90.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	8.5
as6	1101000101100111111111011110011 11	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	12.5
aaa5	1001011001100001100001001001001 1	0.10.90.60.60.40.60.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.4	5.9
aaa6	0111100101010110111110010110110 0	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	11.2
af5	1000000111110001110010101011100 0	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.70.5	9
af6	1001101110001010101101000100010 1	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	8.5

Average Fitness::8.59

==========

Best Component	Max Fitness
as6	12.5
aaa6	11.2
aa5	10.5

Average Fitness: 8.59

****

Generation: 17-------->>

Table A.61 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	0000101000111111010101010110000 1	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
c6	0111011000000000000000010010000010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
cc5	0000001011100100000000010111100 10	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
cc6	0001010100011000101110010001110 0	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
a5	0011100000100100110010110001110 1	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
a6	0000101000111111010101010110000 1	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.5

ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.5
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	9.9
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	9.9
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	9.9
aa5	10001001111100011100101010111000	0.90.10.20.40.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
aa6	10011011100010101011010001000101	0.90.10.20.40.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
as5	11101010110110000010100001110010	0.90.10.20.40.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	12.5
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	12.5
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	12.5
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	12.5
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	12.5

Table A.62 Population after Crossover

Crossover Point:: 22

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3	9.3 1 3
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3	9.3 2 3
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.9	9.3	10.5 3 12
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3	9.3 4 3
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.9	9.3	10.5 5 11
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.5	12.5 6 15
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.5	10.5 7 13
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	9.9	9.9 8 10
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	9.9	10.5 9 11
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	9.9	9.3 10 4
aa5	10001001111100011100101010111000	0.90.10.20.40.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5	10.5 11 11
aa6	10011011100010101011010001000101	0.90.10.20.40.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5	9.9 12 9
as5	11101010110110000010100001110010	0.90.10.20.40.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.5	10.5 13 11
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.5	9.3 14 3
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	12.5	9.5 15 7
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.5	9.3 16 3
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.5	9.3 17 5
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	12.5	10.5 18 12

Table A.63 Mutation Table

Bit Position	Old Number	New Number
17	0.4	0.4
4	0.7	0.2

29	0.6	0.6
18	0.7	0.6
14	0.9	0.5
28	0.9	0.2
18	0.7	0.2
24	0.5	0.8
24	0.5	0.9
25	0.6	0.9
5	0.9	0.8
14	0.9	0.1

Table A.64 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
c6	01110110000000000000010010000010	0.90.10.20.20.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	3
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.9	6.3
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.60.60.80.60.40.30.50.60.60.80.90.80.80.20.4	8.4
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.20.40.20.40.60.60.50.60.70.40.70.60.80.60.40.90.50.60.60.80.90.60.80.20.9	9
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.20.60.80.20.9	9.5
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.20.80.80.60.40.90.50.60.60.80.90.60.80.20.9	7.4
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.90.60.80.20.9	9.9
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.90.60.60.80.90.60.80.20.9	9.4
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.30.50.90.60.80.90.80.80.20.4	5.9
aa5	10001001111100011100101010111000	0.90.10.20.40.80.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.4
aa6	10011011100010101011010001000101	0.90.10.20.40.90.90.40.20.90.20.60.60.60.10.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	9.2
as5	11101010110110000010100001110010	0.90.10.20.40.90.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	8.7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	6.8
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	11.4
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	8.2
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.90.50.60.60.80.90.60.80.20.9	8.5

Average Fitness::8.53

==========

Best Component	Max Fitness
as6	12.2
aaa6	11.4
aa5	10.4

Average Fitness: 8.53

****

Generation: 18-------->>

Table A.65 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.20.60.80.20.9	9.5
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.20.60.80.20.9	9.5
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.90.60.80.20.9	9.9
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.90.60.80.20.9	9.9
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.90.60.80.20.9	9.9
aa5	10001001111100011100101010111000	0.90.10.20.40.80.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.4
aa6	10011011100010101011010001000101	0.90.10.20.40.80.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.4
as5	11101010110110000010100001110010	0.90.10.20.40.80.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.60.80.20.9	10.4
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.2
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.2
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.2

Table A.66 Population after Crossover

Crossover Point:: 26

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.3	9.9 1 9
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.3	9.9 2 10
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.20.60.80.20.9	9.3	9.5 3 6
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3	12.2 4 14
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	9.3	9.3 5 2
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.5	12.2 6 17
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.5	12.2 7 16
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.90.60.80.20.9	9.9	9.9 8 9
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.90.60.80.20.9	9.9	9.9 9 8
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.90.60.80.20.9	9.9	10.4 10 12
aa5	10001001111100011100101010111000	0.90.10.20.40.80.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.4	12.2 11 15
aa6	10011011100010101011010001000101	0.90.10.20.40.80.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.4	9.3 12 1

as5	11101010110110000010100001110010	0.90.10.20.40.80.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.20.60.80.20.9	10.4 9.5 13 6
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	12.2 10.4 14 13
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.2 12.2 15 17
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.20.60.80.20.9	12.2 9.5 16 6
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	12.2 9.9 17 9
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	12.2 9.3 18 3

Table A.67 Mutation Table

Bit Position	Old Number	New Number
2	0.1	0.9
27	0.8	0.5
18	0.7	0.5
31	0.2	0.6
17	0.4	0.7
23	0.9	0.9
1	0.9	0.8
9	0.9	0.3
28	0.9	0.7
10	0.2	0.1
5	0.8	0.4
15	0.2	0.6

Table A.68 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.50.90.60.80.20.9	3.5
cc5	00000010111001000000001011110010	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.50.60.80.60.40.30.50.60.60.80.20.60.80.20.9	5
cc6	00010101000110001011100100011100	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.80.80.60.4	8.4
a5	00111000001001001100101100011101	0.90.10.20.70.90.90.40.20.40.20.40.60.60.90.60.70.70.70.60.80.60.40.30.50.60.60.80.90.80.80.20.4	8.8
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9
ad5	11110010011001110100101010000010	0.80.10.20.70.90.90.40.20.90.20.60.50.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	7.8
ad6	10000001100111011111101010000001	0.90.10.20.20.90.90.40.20.30.20.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.90.60.80.20.9	9.3
add5	00111101111000000000010110111001	0.90.10.20.20.90.90.40.20.90.20.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.70.60.80.20.9	9.1
add6	01000010010010100100101111100000	0.90.10.20.20.90.90.40.20.90.10.80.40.60.90.60.70.40.50.80.80.40.40.90.80.60.60.80.90.60.80.20.9	6.4
aa5	10001001111100011100101010111000	0.90.10.20.40.40.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
aa6	10011011100010101011010001000101	0.90.10.20.40.80.90.40.20.90.20.60.60.60.90.60.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9
as5	11101010110110000010100001110010	0.90.10.20.40.80.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.20.60.80.20.9	7.9
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	12.5

aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	6.3
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.20.60.80.20.9	11.2
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	8
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.80.80.20.4	8

Average Fitness::8.34

==========

Best Component	Max Fitness
as6	12.5
aaa6	11.2
aa5	10.2

Average Fitness: 8.34

****

Generation: 19-------->>

Table A.69 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
c6	01110110000000000000010010000010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8
aa5	10001001111100011100101010111000	0.90.10.20.40.40.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
aa6	10011011100010101011010001000101	0.90.10.20.40.40.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
as5	11101010110110000010100001110010	0.90.10.20.40.40.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	12.5
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	12.5
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	12.5
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	12.5
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	12.5

Table A.70 Population after Crossover

Crossover Point:: 19

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8	9.8 1 9
c6	01110110000000000000010010000010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8	9.8 2 10
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8	9.8 3 6
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8	9.8 4 6
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8	9.8 5 10
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.40.40.30.50.60.60.80.90.60.80.20.9	9.8	12.5 6 14
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8	9.8 7 5
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.80.80.20.4	9.8	10.2 8 11
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.40.40.30.50.60.60.80.90.60.80.20.9	9.8	12.5 9 14
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	9.8	9.8 10 5
aa5	10001001111100011100101010111000	0.90.10.20.40.40.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2	10.2 11 12
aa6	10011011100010101011010001000101	0.90.10.20.40.40.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.60.40.30.50.60.60.80.90.60.80.20.9	10.2	9.8 12 3
as5	11101010110110000010100001110010	0.90.10.20.40.40.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.40.40.30.50.60.60.80.90.60.80.20.9	10.2	12.5 13 14
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.5	10.2 14 12
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.50.60.60.80.90.60.80.20.9	12.5	9.8 15 10
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.50.60.60.80.90.60.80.20.9	12.5	9.8 16 7
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.50.60.60.80.90.60.80.20.9	12.5	9.8 17 2
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	12.5	12.5 18 16

Table A.71 Mutation Table

Bit Position	Old Number	New Number
24	0.5	0.4
11	0.4	0.5
19	0.6	0.4
15	0.6	0.5
1	0.9	0.8
7	0.4	0.1
5	0.9	0.3
29	0.8	0.8
13	0.6	0.5
12	0.6	0.8
14	0.9	0.7
13	0.6	0.3

Table A.72 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
c6	01110110000000000000010010000010	0.90.90.20.70.90.90.40.20.40.20.50.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	4.3
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.40.80.60.40.30.50.60.60.80.90.60.80.20.9	5.7
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.50.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	8.2
a5	00111000001001001100101100011101	0.80.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	8.8
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.10.20.40.20.40.60.60.90.60.70.40.70.60.80.40.40.30.50.60.60.80.90.60.80.20.9	9.5
ad5	11110010011001110100101010000010	0.90.90.20.70.30.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	8.3
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.80.80.20.4	9.3
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.40.20.40.20.40.60.50.90.60.70.40.70.60.80.40.40.30.50.60.60.80.90.60.80.20.9	8.6
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.40.20.40.20.40.80.60.90.60.70.40.70.60.80.60.40.30.50.60.60.80.90.60.80.20.9	6.8
aa5	10001001111100011100101010111000	0.90.10.20.40.40.90.40.20.90.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
aa6	10011011100010101011010001000101	0.90.10.20.40.40.90.40.20.90.20.60.60.30.90.20.70.40.70.80.80.60.40.30.50.60.60.80.90.60.80.20.9	8.4
as5	11101010110110000010100001110010	0.90.10.20.40.40.90.40.20.90.20.60.60.60.90.20.70.40.70.80.80.40.40.30.50.60.60.80.90.60.80.20.9	8
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.50.60.60.80.90.60.80.20.9	6.8
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.50.60.60.80.90.60.80.20.9	11.4
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.50.60.60.80.90.60.80.20.9	8.2
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.40.40.30.50.60.60.80.90.60.80.20.9	8.5

Average Fitness::8.51

==========

Best Component	Max Fitness
as6	12.4
aaa6	11.4
aa5	10.2

Average Fitness: 8.51

****

Generation: 20-------->>

Table A.73 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
c6	01110110000000000000010010000010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7

ad5	1111001001100111010010101000010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
ad6	1000000110011101111110101000001	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
add5	0011110111100000000000101101110 01	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
add6	0100001001001010010010111110000 0	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
aa5	1000100111110001110010101011100 0	0.90.10.20.40.40.90.40.20.90.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
aa6	1001101110001010101101000100010 1	0.90.10.20.40.40.90.40.20.90.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
as5	1110101011011000001010000011100 10	0.90.10.20.40.40.90.40.20.90.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
as6	1101000101100111111111011100111 1	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
aaa5	1001011001100001100001001001001 1	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
aaa6	0111100101010110111110010110110 0	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
af5	1000000111110001110010101011100 0	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
af6	1001101110001010101101000100010 1	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4

Table A.74 Population after Crossover

Crossover Point:: 19

Comp Name	AV	WV	Mate1	Mate2
c5	0000101000111111010101010110000 1	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7	9.7 1 6
c6	0111011000000000000000010010000 010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.80.80.20.4	9.7	10.2 2 11
cc5	0000001011100100000000101111001 0	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7	9.7 3 5
cc6	0001010100011000101110010001110 0	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.80.80.20.4	9.7	12.4 4 14
a5	0011100000100100110010110001110 1	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.80.80.20.4	9.7	12.4 5 18
a6	0000101000111111010101010110000 1	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7	9.7 6 7
ad5	1111001001100111010010101000001 0	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.80.80.20.4	9.7	10.2 7 13
ad6	1000000110011101111110101000000 1	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7	9.7 8 4
add5	0011110111100000000000101101110 01	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7	9.7 9 6
add6	0100001001001010010010111110000 0	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.80.80.20.4	9.7	12.4 10 17
aa5	1000100111110001110010101011100 0	0.90.10.20.40.40.90.40.20.90.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2	10.2 11 11
aa6	1001101110001010101101000100010 1	0.90.10.20.40.40.90.40.20.90.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2	12.4 12 17
as5	1110101011011000001010000011100 10	0.90.10.20.40.40.90.40.20.90.20.60.60.60.70.20.70.40.70.80.80.60.40.30.40.60.60.80.90.60.80.20.9	10.2	9.7 13 2
as6	1101000101100111111111011100111 1	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4	10.2 14 13
aaa5	1001011001100001100001001001001 1	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.40.60.60.80.90.60.80.20.9	12.4	9.7 15 5
aaa6	0111100101010110111110010110110 0	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.40.60.60.80.90.60.80.20.9	12.4	9.7 16 5
af5	1000000111110001110010101011100 0	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4	12.4 17 14
af6	1001101110001010101101000100010 1	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.40.60.60.80.90.60.80.20.9	12.4	9.7 18 2

Table A.75 Mutation Table

Bit Position	Old Number	New Number
7	0.4	0.6
21	0.6	0.9
31	0.2	0.2
4	0.7	0.2
28	0.9	0.1
4	0.7	0.5
8	0.2	0.5
9	0.4	0.9
1	0.9	0.1
20	0.8	0.3
10	0.2	0.9
6	0.9	0.2

Table A.76 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
c6	01110110000000000000010010000010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.90.40.90.50.60.60.80.90.80.80.20.4	4.3
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	5.7
cc6	00010101000110001011100100011100	0.90.90.20.20.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.80.80.20.4	7.9
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.10.80.80.20.4	8.3
a6	00001010001111110101010101100001	0.90.90.20.50.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.40.50.40.20.40.60.60.90.60.70.40.70.60.80.60.40.90.50.60.60.80.90.80.80.20.4	8.9
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.40.20.90.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.7
add5	00111101111000000000010110111001	0.10.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	8.5
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.40.20.40.20.40.60.60.90.60.70.40.70.60.30.60.40.90.50.60.60.80.90.80.80.20.4	7.4
aa5	10001001111100011100101010111000	0.90.10.20.40.40.90.40.20.90.90.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.9
aa6	10011011100010101011010001000101	0.90.10.20.40.40.20.40.20.90.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	8.2
as5	11101010110110000010100001110010	0.90.10.20.40.40.90.40.20.90.20.60.60.60.70.20.70.40.70.80.80.60.40.30.40.60.60.80.90.60.80.20.9	8.2
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.40.60.60.80.90.60.80.20.9	6.8
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.40.60.60.80.90.60.80.20.9	11.3
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.30.40.60.60.80.90.60.80.20.9	8.5

Average Fitness::8.64

Best Component	Max Fitness
as6	12.4

aaa6 11.3
aa5 10.9
Average Fitness: 8.64
****

Generation: 21-------->>

Table A.77 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111111010101010100001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
c6	01110110000000000000010010000010	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
a6	00001010001111111010101010100001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9
aa5	10001001111100011100101010111000	0.90.10.20.40.40.90.40.20.90.90.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.9
aa6	10011011100010101011010001000101	0.90.10.20.40.40.90.40.20.90.90.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.9
as5	11101010110110000010100001110010	0.90.10.20.40.40.90.40.20.90.90.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.9
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4

Table A.78 Population after Crossover

Crossover Point:: 10

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111111010101010100001	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.9	12.4 1 14
c6	01110110000000000000010010000010	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.9	12.4 2 16
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.9	12.4 3 17
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9	9.9 4 9
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.40.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.9	10.9 5 11
a6	00001010001111111010101010100001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9	9.9 6 6
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.40.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.9	10.9 7 13

ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.40.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.9 10.9 8 12
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9 9.9 9 6
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	9.9 9.9 10 2
aa5	10001001111100011100101010111000	0.90.10.20.40.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.9 12.4 11 17
aa6	10011011100010101011010001000101	0.90.10.20.40.40.90.40.20.90.90.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	10.9 9.9 12 6
as5	11101010110110000010100001110010	0.90.10.20.40.40.90.40.20.90.90.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	10.9 9.9 13 7
as6	11010001011001111111110111001111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4 12.4 14 15
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4 12.4 15 15
aaa6	01111001010101101111100101101100	0.10.90.60.60.40.60.40.20.90.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	12.4 9.9 16 10
af5	10000001111100011100101010111000	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4 12.4 17 18
af6	10011011100010101011010001000101	0.10.90.60.60.40.60.40.20.90.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	12.4 9.9 18 6

Table A.79 Mutation Table

Bit Position	Old Number	New Number
22	0.4	0.2
3	0.2	0.6
12	0.6	0.9
11	0.4	0.8
4	0.7	0.7
27	0.8	0.6
21	0.6	0.1
12	0.6	0.9
10	0.2	0.4
21	0.6	0.5
21	0.6	0.7
5	0.4	0.1

Table A.80 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
c6	01110110000000000000010010000010	0.90.90.60.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	4.9
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.60.20.40.20.80.90.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	6.9
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	8.1
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.40.20.60.60.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.1
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	9.7
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.40.20.60.60.60.70.20.70.40.70.80.80.10.40.90.50.60.60.80.90.80.80.20.4	8.2
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.40.20.60.90.60.70.20.70.40.70.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.6
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.40.40.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	8.7

add6	0100001001001010010010111111100000	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.50.40.30.40.60.60.80.90.60.80.20.9	6.8
aa5	1000100111110001110010101011100 0	0.90.10.20.40.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	11
aa6	1001101110001010101101000100 0101	0.90.10.20.40.10.90.40.20.90.90.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	8.6
as5	1110101011011000001010000 1110010	0.90.10.20.40.40.90.40.20.90.90.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	8.7
as6	1101000101100111111111011100 1111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
aaa5	1001011001100001100001001001 0011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	6.3
aaa6	0111100101010110111110010110 1100	0.10.90.60.60.40.60.40.20.90.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	11.3
af5	1000000111110001110010101011100 0	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9
af6	1001101110001010101101000100 0101	0.10.90.60.60.40.60.40.20.90.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.80.90.60.80.20.9	8.3

Average Fitness::8.73

==========

Best Component	Max Fitness
as6	12.4
aaa6	11.3
aa5	11

Average Fitness: 8.73

****

Generation: 22-------->>

Table A.81 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	0000101000111111010101010110 0001	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
c6	0111011000000000000001001000 0010	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
cc5	0000001011100100000000101111 0010	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
cc6	0001010100011000101110010001 1100	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
a5	0011100000100100110010110000 1101	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
a6	0000101000111111010101010110 0001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	9.7
ad5	1111001001100111010010101000 0010	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	9.7
ad6	1000000110011101111110101000 0001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	9.7
add5	0011110111100000000001011011 1001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	9.7
add6	0100001001001010010010111111 00000	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	9.7
aa5	1000100111110001110010101011 1000	0.90.10.20.40.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	11
aa6	1001101110001010101101000100 0101	0.90.10.20.40.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	11
as5	1110101011011000001010000 1110010	0.90.10.20.40.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	11
as6	1101000101100111111111011100 1111	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
aaa5	1001011001100001100001001001 0011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
aaa6	0111100101010110111110010110 1100	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
af5	1000000111110001110010101011100 0	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4
af6	1001101110001010101101000100 0101	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4

Table A.82 Population after Crossover

Crossover Point:: 4

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9.5	11 1 12
c6	01110110000000000000010010000010	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	9.5	9.7 2 6
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	9.5 3 2
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	9.5 4 5
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	9.5 5 4
a6	00001010001111110101010101100001	0.90.90.20.70.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.7	12.4 6 16
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.7	9.5 7 4
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.7	9.5 8 3
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	9.7	9.7 9 8
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.7	9.5 10 5
aa5	10001001111100011100101010111000	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	11	12.4 11 18
aa6	10011011100010101011010001000101	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	11	12.4 12 15
as5	11101010110110000010100001110010	0.90.10.20.40.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	11	11 13 12
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.4	9.5 14 3
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	12.4	12.4 15 16
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	12.4	9.7 16 10
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	12.4	9.7 17 8
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.4	9.5 18 2

Table A.83 Mutation Table

Bit Position	Old Number	New Number
17	0.4	0.2
24	0.4	0.3
11	0.8	0.4
10	0.2	0.2
9	0.4	0.2
26	0.6	0.8
30	0.8	0.4
24	0.5	0.2
27	0.6	0.3
11	0.8	0.8
5	0.4	0.4
24	0.5	0.5

Table A.84 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9
c6	01110110000000000000010010000010	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.30.60.60.60.90.60.80.20.9	4.5
cc5	00000010111001000000001011110010	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	6.5
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.6
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
a6	00001010001111110101010101100001	0.90.90.20.70.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.80.80.90.80.80.20.4	9.2
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.40.20.4	9.3
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.20.60.60.80.90.80.80.20.4	9.3
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.30.90.60.80.20.9	8
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	7.4
aa5	10001001111100011100101010111000	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
aa6	10011011100010101011010001000101	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	8.6
as5	11101010110110000010100001110010	0.90.10.20.40.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	6.3
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	11.6
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.40.60.60.90.60.70.40.70.60.80.60.40.30.40.60.60.60.90.60.80.20.9	7.3
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8

Average Fitness::8.57

==========

Best Component	Max Fitness
as6	12.2
aaa6	11.6
aa5	10.2

Average Fitness: 8.57

****

Generation: 23-------->>

Table A.85 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3

ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
aa5	10001001111100011100101010111000	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
aa6	10011011100010101011010001000101	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
as5	11101010110110000010100001110010	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	10.2
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2

Table A.86 Population after Crossover

Crossover Point:: 11

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9	12.2 1 14
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9	12.2 2 15
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9 9	3 3
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9 9	4 1
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.40.90.50.60.60.80.90.80.80.20.4	9.3	10.2 5 13
a6	00001010001111111010101010100001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3	9.3 6 5
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3	12.2 7 14
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9.3	9 8 3
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3	12.2 9 16
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3	12.2 10 14
aa5	10001001111100011100101010111000	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.2	9.3 11 9
aa6	10011011100010101011010001000101	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.2	9.3 12 6
as5	11101010110110000010100001110010	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.2	9.3 13 5
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	12.2 14 17
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	12.2 15 16
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	12.2	9 16 2
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	9.3 17 6
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	12.2	9 18 4

Table A.87 Mutation Table

Bit Position	Old Number	New Number
14	0.9	0.1
20	0.8	0.2
9	0.9	0.7
1	0.9	0.8
22	0.4	0.5
29	0.8	0.8
5	0.9	0.9
17	0.2	0.7
6	0.9	0.5
31	0.2	0.9
5	0.4	0.3
4	0.4	0.4

Table A.88 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.20.60.20.90.50.60.60.80.90.80.80.20.4	4.1
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.70.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	7.7
cc6	00010101000110001011100100011100	0.80.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	8.5
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.50.90.50.60.60.80.90.80.80.20.4	9.3
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.70.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	9.5
add5	00111101111000000000010110111001	0.90.90.20.70.90.50.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	7.9
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.90.4	7.4
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1
aa6	10011011100010101011010001000101	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.4
as5	11101010110110000010100001110010	0.90.10.20.40.40.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.2
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	6.6
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	12
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.5
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	8

Average Fitness::8.62

Best Component Max Fitness

as6 12.2
aaa6 12
aa5 10.1
Average Fitness: 8.62
****

Generation: 24-------->>

Table A.89 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8
cc6	00010101000110001011100100011100	0.80.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.80.70.40.90.50.60.60.80.90.80.80.20.4	8.5
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.50.90.50.60.60.80.90.80.80.20.4	9.3
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa6	01111001010101101111100101011100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2

Table A.90 Population after Crossover

Crossover Point:: 12

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8 10.1	1 12
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8 12.2	2 14
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8 10.1	3 12
cc6	00010101000110001011100100011100	0.80.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.5 12.2	4 18
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3 9.5	5 6
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5 12.2	6 16

ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5 9.5 7 8
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5 12.2 8 17
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5 12.2 9 17
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5 10.1 10 12
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1 8 11 3
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1 8 12 1
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1 9.5 13 8
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2 9.5 14 8
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2 9.5 15 8
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2 10.1 16 12
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2 9.5 17 7
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2 8 18 3

Table A.91 Mutation Table

Bit Position	Old Number	New Number
32	0.4	0.8
32	0.4	0.8
12	0.6	0.6
10	0.9	0.4
10	0.2	0.9
9	0.2	0.7
25	0.6	0.1
16	0.7	0.8
14	0.9	0.6
1	0.9	0.4
22	0.2	0.3
13	0.6	0.6

Table A.92 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	9.2
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	4.1
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	7.9
cc6	00010101000110001011100100011100	0.80.90.20.70.40.90.40.20.90.40.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.6
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.10.60.80.90.80.80.20.4	8.8
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.90.60.80.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.2

add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.20.20.80.60.60.60.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.3
add6	01000010010010100100101111100000	0.40.90.20.70.90.90.60.20.20.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	7.4
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.3
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.1
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	6.6
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.1
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.5
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8

Average Fitness::8.68

==========

Best Component	Max Fitness
as6	12.2
aaa6	12.1
aa5	10.1

Average Fitness: 8.68

****

Generation: 25-------->>

Table A.93 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111111010101010110000 1	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	9.2
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	9.2
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	9.2
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	9.2
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
a6	00001010001111111010101010110000 1	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2

Table A.94 Population after Crossover

Crossover Point:: 23

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.2	9.5 1 9
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.2	12.2 2 17
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	9.2	9.2 3 4
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.2	10.1 4 11
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3	10.1 5 12
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	9.5 6 9
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	9.5 7 9
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	12.2 8 14
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	9.5	9.2 9 1
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	9.5 10 6
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1	10.1 11 11
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1	12.2 12 14
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.8	10.1	9.2 13 4
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	10.1 14 11
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	12.2 15 16
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	12.2	9.2 16 3
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	12.2 17 16
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	12.2	9.2 18 2

Table A.95 Mutation Table

Bit Position	Old Number	New Number
20	0.8	0.3
6	0.9	0.8
18	0.5	0.9
22	0.2	0.1
5	0.9	0.9
11	0.8	0.9
14	0.9	0.8
9	0.7	0.1
3	0.2	0.4
29	0.8	0.5
19	0.8	0.4
12	0.6	0.6

Table A.96 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	0000101000111111010101010110000 1	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
c6	0111011000000000000000010010000010	0.90.90.20.70.40.80.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	4
cc5	0000001011100100000000010111100 10	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.90.80.80.60.20.90.50.60.60.80.90.80.80.20.8	7.9
cc6	0001010100011000101110010001110 0	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.10.90.50.60.60.80.90.80.80.20.4	8.6
a5	0011100000100100110010110001110 1	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
a6	0000101000111111010101010110000 1	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
ad5	1111001001100111010010101000001 0	0.90.90.20.70.90.90.60.20.70.20.80.60.60.80.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.2
ad6	1000000110011101111110101000000 1	0.90.90.20.70.90.90.60.20.10.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9
add5	0011110111100000000001011011100 1	0.90.90.40.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	9.4
add6	0100001001001010010010111110000 0	0.90.90.20.70.90.90.60.20.70.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.50.80.20.4	7.4
aa5	1000100111110001110010101011100 0	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1
aa6	1001101110001010101101000100010 1	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.4	8.4
as5	1110101011011000001010000111001 0	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.8	8.1
as6	1101000101100111111111011110011 11	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	1001011001100001100001001001001 1	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	6.6
aaa6	0111100101010110111110010110110 0	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	12.1
af5	1000000111110001110010101011100 0	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.5
af6	1001101110001010101101000100010 1	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.8	8.4

Average Fitness::8.73

==========

Best Component	Max Fitness
as6	12.2
aaa6	12.1
aa5	10.1

Average Fitness: 8.73

****

Generation: 26-------->>

Table A.97 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	0000101000111111010101010110000 1	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
c6	0111011000000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
cc5	0000001011100100000000010111100 10	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
cc6	0001010100011000101110010001110 0	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.10.90.50.60.60.80.90.80.80.20.4	8.6
a5	0011100000100100110010110001110 1	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
a6	0000101000111111010101010110000 1	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6

ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2

Table A.98 Population after Crossover

Crossover Point:: 27

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3	9.3 1 5
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3	10.1 2 11
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3	12.2 3 16
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.10.90.50.60.60.80.90.80.80.20.4	8.6	9.6 4 8
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3	9.3 5 5
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6	9.3 6 5
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6	9.3 7 5
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6	9.6 8 10
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6	9.6 9 6
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6	8.6 10 4
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1	9.3 11 5
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1	9.6 12 8
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.4	10.1	9.6 13 9
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	9.6 14 10
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	9.6 15 10
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	12.2 16 14
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	8.3 17 2
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	12.2 18 15

Table A.99 Mutation Table

Bit Position	Old Number	New Number
17	0.4	0.2
8	0.2	0.9
25	0.6	0.7
30	0.8	0.2
16	0.7	0.5
8	0.2	0.6
14	0.9	0.4
24	0.5	0.8
28	0.9	0.3
8	0.2	0.7
32	0.4	0.2
21	0.6	0.9

Table A.100 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.90.90.90.80.60.60.90.60.70.40.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	4.1
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.30.60.20.90.50.70.60.80.90.80.80.20.4	8
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.40.50.80.80.60.10.90.50.60.60.80.90.80.20.20.4	8
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.70.20.90.60.60.40.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.9
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.80.60.60.80.90.80.80.20.4	9.6
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.30.80.80.20.4	8.3
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.70.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	7.4
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.2	10.1
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.90.30.90.50.60.60.80.90.80.80.20.4	8
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.4	7.7
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	6.6
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.1
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.5
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8

Average Fitness::8.59

Best Component	Max Fitness
as6	12.2

aaa6 12.1
aa5 10.1
Average Fitness: 8.59
****

Generation: 27-------->>

Table A.101 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111111010101010100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3
a6	00001010001111111010101010100001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.2	10.1
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.2	10.1
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.2	10.1
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2

Table A.102 Population after Crossover

Crossover Point:: 16

Comp Name	AV	WV	Mate1 Mate2
c5	00001010001111111010101010100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3 8.3 1 3
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3 8.3 2 4
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3 8.3 3 2
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3 8.3 4 4
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.3 12.2 5 14
a6	00001010001111111010101010100001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	9.6 8.3 6 2

ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6 12.2 7 17
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6 12.2 8 16
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6 12.2 9 14
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6 12.2 10 17
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1 12.2 11 17
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1 9.6 12 10
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.1 9.6 13 8
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2 9.6 14 9
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	12.2 8.3 15 4
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.2	12.2 10.1 16 11
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	12.2 8.3 17 3
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	12.2 8.3 18 2

Table A.103 Mutation Table

Bit Position	Old Number	New Number
28	0.9	0.7
27	0.8	0.2
23	0.9	0.6
1	0.9	0.3
32	0.4	0.6
15	0.6	0.4
18	0.5	0.3
20	0.8	0.7
10	0.2	0.9
5	0.9	0.8
26	0.6	0.9
23	0.9	0.5

Table A.104 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	8.3
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.20.90.80.80.20.4	4.1
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.60.50.60.60.80.90.80.80.20.4	7.6
cc6	00010101000110001011100100011100	0.30.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	7.9
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.6	9.5
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.40.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.9
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.30.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.2
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	9.9

add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10
add6	01000010010010100100101111100000	0.90.90.20.70.80.90.60.60.70.20.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	7.4
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.90.80.90.80.80.20.4	10.1
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.50.50.60.60.80.90.80.80.20.4	8.3
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.1
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	6.4
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.40.80.60.30.90.50.60.60.80.90.80.80.20.2	11.7
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	8.3
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.90.80.80.20.4	7.3

Average Fitness::8.62

==========

Best Component	Max Fitness
as6	12.2
aaa6	11.7
aa5	10.1

Average Fitness: 8.62

****

Generation: 28-------->>

Table A.105 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	8.3
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	8.3
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	8.3
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	8.3
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.6	9.5
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.6	9.5
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.6	9.5
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.60.70.20.90.60.60.90.60.70.40.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	9.9
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10
aa5	10001001111100011100101010111000	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.90.80.90.80.80.20.4	10.1
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.90.80.90.80.80.20.4	10.1
as5	11101010110110000010100001110010	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.90.80.90.80.80.20.4	10.1
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2

Table A.106 Population after Crossover

Crossover Point:: 5

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.3	12.2 1 14
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.6	8.3	9.5 2 7
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.3	12.2 3 16
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	8.3	8.3 4 2
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	10 5 9
a6	00001010001111110101010101100001	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	12.2 6 15
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	12.2 7 14
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.9	12.2 8 17
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10	12.2 9 16
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10	12.2 10 14
aa5	10001001111100011100101010111000	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.40.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	10.1	9.9 11 8
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.90.80.90.80.80.20.4	10.1	10.1 12 13
as5	11101010110110000010100001110010	0.90.10.20.40.30.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.6	10.1	9.5 13 7
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.2	8.3 14 2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	12.2 15 15
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.2	8.3 16 1
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.6	12.2	9.5 17 6
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.2	8.3 18 3

Table A.107 Mutation Table

Bit Position	Old Number	New Number
10	0.2	0.1
31	0.2	0.1
27	0.8	0.3
3	0.2	0.5
14	0.9	0.9
2	0.9	0.1
29	0.8	0.9
9	0.4	0.3
25	0.6	0.8
22	0.2	0.3
17	0.4	0.8
19	0.8	0.8

Table A.108 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.10.6	4.2
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.30.90.80.80.20.4	6.4
cc6	00010101000110001011100100011100	0.90.90.50.70.40.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	7.7
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.4
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
ad5	11110010011001110100101010000010	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.90.80.20.4	9.3
ad6	10000001100111011111101010000001	0.90.90.20.70.90.90.60.20.30.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.2
add5	00111101111000000000010110111001	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.80.60.80.90.80.80.20.4	8.7
add6	01000010010010100100101111100000	0.90.90.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.30.90.50.60.60.80.90.80.80.20.4	7.4
aa5	10001001111100011100101010111000	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	10.8
aa6	10011011100010101011010001000101	0.90.10.20.40.30.60.40.20.90.20.80.60.60.10.60.70.40.50.80.80.60.20.90.50.60.90.80.90.80.80.20.4	8.6
as5	11101010110110000010100001110010	0.90.10.20.40.30.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.6	8.3
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	6.6
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.1
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.60.20.20.90.80.60.60.90.60.50.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.6	8.8
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	7.6

Average Fitness::8.66

==========

Best Component	Max Fitness
as6	12.2
aaa6	12.1
aa5	10.8

Average Fitness: 8.66

****

Generation: 29-------->>

Table A.109 Population after Reproduction

Comp Name	AV	WV	Fitness Value
c5	00001010001111110101010101100001	0.90.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9
c6	01110110000000000000010010000010	0.90.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9
cc5	00000010111001000000001011110010	0.90.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9
cc6	00010101000110001011100100011100	0.90.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.4
a6	00001010001111110101010101100001	0.90.10.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
ad5	11110010011001110100101010000010	0.90.10.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
ad6	10000001100111011111101010000001	0.90.10.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
add5	00111101111000000000010110111001	0.90.10.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5
add6	01000010010010100100101111100000	0.90.10.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5

aa5	10001001111100011100101010111000	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	10.8
aa6	10011011100010101011010001000101	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	10.8
as5	11101010110110000010100001110010	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	10.8
as6	11010001011001111111110111001111	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.2
aaa6	01111001010101101111100101101100	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.2
af5	10000001111100011100101010111000	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.2
af6	10011011100010101011010001000101	0.10.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	12.2

Table A.110 Population after Crossover

Crossover Point:: 1

Comp Name	AV	WV	Mate1	Mate2
c5	00001010001111110101010101100001	0.90.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	9	12.2 1 17
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9	9.5 2 8
cc5	00000010111001000000001011110010	0.90.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	9	12.2 3 14
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.60.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9	9.4 4 5
a5	00111000001001001100101100011101	0.90.90.20.70.90.90.60.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.4	9.4 5 5
a6	00001010001111110101010101100001	0.90.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.5	9 6 2
ad5	11110010011001110100101010000010	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	9.5	10.8 7 13
ad6	10000001100111011111101010000001	0.90.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	9.5	12.2 8 17
add5	00111101111000000000010110111001	0.90.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	9.5	12.2 9 16
add6	01000010010010100100101111100000	0.90.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	9.5	12.2 10 18
aa5	10001001111100011100101010111000	0.90.10.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.8	9.5 11 8
aa6	10011011100010101011010001000101	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	10.8	10.8 12 11
as5	11101010110110000010100001110010	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	10.8	10.8 13 12
as6	11010001011001111111110111001111	0.10.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	9 14 4
aaa5	10010110011000011000010010010011	0.10.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	12.2	10.8 15 11
aaa6	01111001010101101111100101101100	0.10.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	9 16 1
af5	10000001111100011100101010111000	0.10.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	9 17 2
af6	10011011100010101011010001000101	0.10.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2	9 18 3

Table A.111 Mutation Table

Bit Position	Old Number	New Number
27	0.8	0.6
15	0.6	0.6
6	0.9	0.9
7	0.6	0.9
3	0.2	0.4

30	0.8	0.1
31	0.2	0.1
8	0.2	0.3
1	0.9	0.9
21	0.6	0.2
8	0.2	0.6
20	0.7	0.4

Table A.112 Population after Mutation

Comp Name	AV	WV	New Fitness
c5	00001010001111110101010101100001	0.90.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.60.70.80.80.20.4	8.6
c6	01110110000000000000010010000010	0.90.10.20.70.90.90.60.20.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	3.5
cc5	00000010111001000000001011110010	0.90.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	7.7
cc6	00010101000110001011100100011100	0.90.90.20.70.90.90.90.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9
a5	00111000001001001100101100011101	0.90.90.40.70.90.90.60.60.70.90.90.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	9.6
a6	00001010001111110101010101100001	0.90.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.10.20.4	9
ad5	11110010011001110100101010000010	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.10.4	8.2
ad6	10000001100111011111101010000001	0.90.90.60.60.90.90.40.30.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	9.2
add5	00111101111000000000010110111001	0.90.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.60.20.90.50.60.60.80.70.80.80.20.4	9.8
add6	01000010010010100100101111100000	0.90.90.60.60.90.90.40.20.90.90.80.60.60.90.60.70.20.50.80.30.20.20.90.50.60.60.80.70.80.80.20.4	7.5
aa5	10001001111100011100101010111000	0.90.10.20.70.90.90.60.60.40.20.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	10.6
aa6	10011011100010101011010001000101	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.40.60.20.90.50.60.60.80.90.80.80.20.4	8.7
as5	11101010110110000010100001110010	0.90.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	8.1
as6	11010001011001111111110111001111	0.10.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	12.2
aaa5	10010110011000011000010010010011	0.10.10.20.40.30.90.60.60.70.20.90.60.60.90.60.70.80.50.80.70.60.20.90.50.60.60.80.90.80.80.20.4	6.9
aaa6	01111001010101101111100101101100	0.10.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	11.2
af5	10000001111100011100101010111000	0.10.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	8.4
af6	10011011100010101011010001000101	0.10.90.20.70.40.90.60.20.40.10.80.60.60.90.60.70.40.50.80.80.60.20.90.50.60.60.80.90.80.80.20.4	7.6

Average Fitness::8.66

==========

Best Component	Max Fitness
as6	12.2
aaa6	11.2
aa5	10.6

Average Fitness: 8.66

****