IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 11, Issue 4 (May. - Jun. 2013), PP 37-43
www.iosrjournals.org

Implementation of Matching Tree Technique for Online Record Linkage

B.Rakesh[1], M.Madhavi[2] ,CH.Venkata Ratnam[3], C.Ravindra Murthy[4]
1Asst.professor, Sreevidyanikethan engg.college, Tirupathi, India
2 Asst.professor, ASRA, Hyderabad, India,
3 Asst.professor, Holy Mery institute of engg&technology, hyd,India
4 Asst.professor, Sreevidyanikethan engg.college, Tirupathi, India,

Abstract: *The task of finding records refering to the same entity across heterogeneous data sources is known as record linkage .the necessity to consolidate the information located in different data sources has been widely documented in recent years. For the purpose of completing this goal, several types of problems ,must be solved by an organization .when same real world substance is used by different identifiers in different sources entity heterogeneity problem will arise. For solving the entity heterogeneity problem statistical record linkage technique could be used. However, the use of such techniques for online record linkage could pose a tremendous communication bottleneck in a distributed environment (where entity heterogeneity problem often encountered). In order to resolve this issue, we develop a matching tree, similar to a decision tree, and use it to propose techniques that reduce the communication overhead significantly, while providing matching decisions that are guaranteed to be the same as those obtained using the conventional linkage technique. These techniques have been implemented, and experiments with real-world and synthetic databases show significant reduction in communication overhead.*

I. Introduction

The record-linkage problem identifying and linking duplicate records arises in the context of data cleansing, which is a necessary pre-step to many database applications. Databases frequently contain approximately duplicate fields and records that refer to the same real-world entity, but are not identical.

Importance of data linkage in a variety of data-analysis applications, developing effective an efficient techniques for record linkage has emerged as an important problem. It is further evidenced by the emergence of numerous organizations (e.g., Trillium, First Logic , Vality, Data Flux) that are developing specialized domain specific record-linkage and data-cleansing tools.

The data needed to support these decisions are often scattered in heterogeneous distributed databases. In such cases, it may be necessary to link records in multiple databases so that one can consolidate and use the data pertaining to the same real world entity. If the databases use the same set of design standards, this linking can easily be done using the primary key (or other common candidate keys). However, since these heterogeneous databases are usually designed and managed by different organizations (or different units within the same organization), there may be no common candidate key for linking the records. Although it may be possible to use common non key attributes (such as name, address, and date of birth) for this purpose, the result obtained using these attributes may not always be accurate. This is because non key attribute values may not match even when the records represent the same entity instance in reality.

The databases exhibiting entity heterogeneity are distributed, and it is not possible to create and maintain a central data repository or warehouse where pre computed linkage results can be stored. A centralized solution may be impractical for several reasons. First, if the databases span several organizations, the owner ship and cost allocation issues associated with the warehouse could be quite difficult to address. Second ,even if the warehouse could be developed, it would be difficult to keep it up-to-date. As updates occur at the operational databases, the linkage results would become stale if they are not updated immediately .This staleness maybe unacceptable in many situations .For instance, in a criminal investigation, one maybe interested in the profile of crimes committed in the last 24 hours within a certain radius of the crime scene. In order to keep the warehouse current, the sites must agree to transmit incremental changes to the data warehouse on a real-time basis. Even if such an agreement is reached, it would be difficult to monitor and enforce it. For example, a site would often have no incentive to report the insertion of a new record immediately. Therefore, these changes are likely to be reported to the warehouse at a later time, thereby increasing the staleness of the linkage tables and limiting their usefulness. In addition, the overall data management tasks could be prohibitively time-consuming, especially in situations where there are many databases, each with many records, undergoing real-time changes. This is because the warehouse must maintain a linkage table for each pair of sites, and must update them every time one of the associated databases changes.

The participating sites allow controlled sharing of portions of their databases using standard database queries, but they do not allow the processing of scripts, stored procedures, or other application programs from another organization .The issue here is clearly not one of current technological abilities, but that of management and control. If the management of an organization wants to open its databases to outside scripts from other organizations, there are, of course, a variety of ways to actually implement it. However, the decision to allow only a limited set of database queries (and nothing more) is not based on technological limitations; rather it is often a management decision arising out of security concerns. More investment in technology or a more sophisticated scripting technique, therefore, is not likely to change this situation. A direct consequence of this fact is that the local site cannot simply send the lone enquiry record to the remote site and ask the remote site to perform the record linkage and send the results back

1.1 OBJECTIVE

If the databases use the same set of design standards, this linking can easily be done using the primary key (or other common candidate keys). However, since these heterogeneous databases are usually designed and managed by different organizations (or different units within the same organization), there may be no common candidate key for linking the records. Although it may be possible to use common non key attributes (such as name, address, and date of birth) for this purpose, the result obtained using these attributes may not always be accurate. This is because non key attribute values may not match even when the records represent the same entity instance in reality.

II. Literature Survey:

2.1 Linked Record Health Data Systems:

The goal of record linkage is to link quickly and accurately records that correspond to the same person or entity. Whereas certain patterns of agreements and disagreements on variables are more likely among records pertaining to a single person than among records for different people, the observed patterns for pairs of records can be viewed as arising from a mixture of matches and non matches. Mixture model estimates can be used to partition record pairs into two or more groups that can be labeled as probable matches (links) and probable non matches. A method is proposed and illustrated that uses marginal information in the database to select mixture models, identifies sets of records for clerks to review based on the models and marginal information, incorporates clerically reviewed data, as they become available, into estimates of model parameters, and classifies pairs as links, non links, or in need of further clerical review. The procedure is illustrated with five datasets from the U.S. Bureau of the Census. It appears to be robust to variations in record-linkage sites. The clerical review corrects classifications of some pairs directly and leads to changes in classification of others through re estimation of mixture models

2.2 Efficient Private Record Linkage:

Record linkage is the computation of the associations among records of multiple databases. It arises in contexts like the integration of such databases, online interactions and negotiations, and many others. The autonomous entities who wish to carry out the record matching computation are often reluctant to fully share their data. In such a framework where the entities are unwilling to share data with each other, the problem of carrying out the linkage computation without full data exchange has been called private record linkage. Previous private record linkage techniques have made use of a third party. We provide efficient techniques for private record linkage that improve on previous work in that (i) they make no use of a third party; (ii) they achieve much better performance than that of previous schemes in terms of execution time and quality of output (i.e., practically without false negatives and minimal false positives). Our software implementation provides experimental validation of our approach and the above claims.

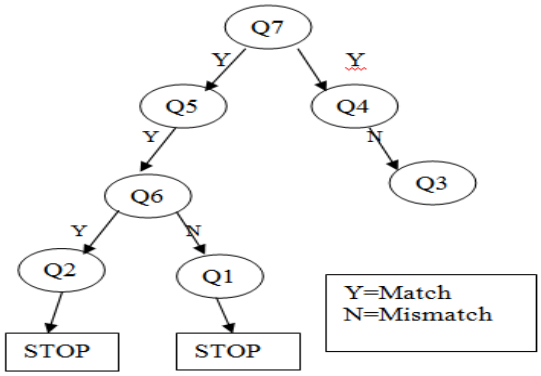


FIG 1: A sample tree showing the attribute aquistion order

The RTS technique used in that framework is scenario-slicing which is not safe. Additionally, the mechanism with which they connect services and applications together is performed manually through the creation and use of scenario-slices. Our approach, in contrast, is safe, works with Java Web services without requiring any additional information other than what is provided by Axis and the source code, and is This framework is described as rapid and adaptive end-to-end regression testing. provided programmatically.

The code-based RTS for component-based software by Harrold's team is also a significantly related work. Their approach revolves around what the authors refer to as met content. This meta-content is additional data expected to be provided by vendors in lieu of source code in such cases where that content may be restricted by patent or copyright law. Their required meta-content is threefold: edge coverage achieved by the test suite with respect to the component, component version, and a way to query the component for the edges affected by the changes between the two given versions. This code-based approach is a white-box testing using code where applicable and using meta-content where necessary. This is a safe RTS. However, it cannot be directly applied to Web services due to Web services being composable in nature. In a Web services world, an application could call a service, which in turn calls other services. In such an environment, an application can be affected not only by the changes in the services it directly calls on but also by the changes in the services that are used indirectly. Thus, just querying about changes in every called component (service) is not sufficient.

TRANSFERED	**CONCURRENT**	**SEQUENTIAL**
Identifier	Not applicable	Sequential identifier Acquistion
Attribute	Concurrent attribute acquistion	Sequential attribute Acquistion

Fig 2: Possible tree based linkage techniques

2.3 A Survey of Approaches to Automatic Schema Matching:

Schema matching is a basic problem in many database application domains, such as data integration, Ebusiness, data warehousing, and semantic query processing .In current implementations, schema matching is typically performed manually, which has significant limitations. On the other hand, previous research papers have proposed many techniques to achieve a partial automation of the match operation for specific application domains. We present a taxonomy that covers many of these existing approaches, and we describe the approaches in some detail. In particular ,we distinguish between schema-level and instance-level, element-level and structure-level, and language-based and constraint-based matchers. Based on our classification we review some previous match implementations thereby indicating which part of the solution space they cover .We intend our taxonomy and review of past work to be useful when comparing different approaches to schema matching, when developing a new match algorithm, and when implementing a schema matching component.

One empirical approach is to perform a case study on an existing program that has several versions and existing test suites. The changes to such a program would be "real". However, under this approach, the circumstances under which evolution and testing occur are not controlled. For example, the changes may have different sizes and implications, and there may be only one test suite per version. Hence, it could be difficult to draw valid inferences about causality or about the ability of results to generalize. A second empirical approach is to perform a controlled experiment in which changes are simulated. The advantage of such an experiment is that the independent variables of interest (number of changes, location of changes, and test suite) can be manipulated to determine their impact on dependent variables . This lets us apply different values to the independent variables in a controlled fashion, so that results are not likely to depend on unknown or uncontrolled factors. The primary weakness of this approach, however, is the threat to external validity posed by the change simulation. Because each approach has different advantages and disadvantages, we employed both. 3 Study 1 We present our controlled experiment first.

A Comparison of Fast Blocking Methods for Record Linkage

The task of linking databases is an important step in an increasing number of data mining projects, because linked data can contain information that is not available otherwise, or that would require time-consuming and expensive collection of specific data. The aim of linking is to match and aggregate all records that refer to the same entity. One of the major challenges when linking large databases is the efficient and accurate classification of record pairs into matches and non-matches. While traditionally classification was based on manually-set thresholds or on statistical procedures, many of the more recently developed classification methods are based on supervised learning techniques. They therefore require training data, which is often not available in real world situations or has to be prepared manually, an expensive, cumbersome and time-consuming process.

We generated 30 versions for each change level, each with a randomly defined change probability. One of our research questions concerns differences in types of code coverage information. For this experiment, we selected two types of coverage: statement coverage, which tracks the statements executed by each test, and function coverage, which tracks the functions executed by each test. In both cases, we track not frequency of execution, but just whether the component was or was not executed. To measure coverage, we used the Aristotle analysis system. Some branch modifications caused Space to crash. One option for handling this was to discard tests that caused crashes; however, this could discard valuable coverage information, causing us to underestimate the effects of changes

Moreover, in practice, such tests would not be discarded. A second option was to modify Space to capture termination signals and record coverage data before failing. We chose this second option. Similarly, we inserted traps into Space to halt execution on modifications that lead to infinite loops. (In practice, such loops would be terminated by human intervention, our traps simulate this.) 3.2 Experiment Design To put Space and its modifications through all desired experimental conditions we selected a factorial design, considering all combinations of versions, test suites, and change levels. In more formal terms, our design constitutes a completely randomized factorial design with three treatments: (T1) versions, with 30 levels (excluding the baseline), (T2) test suites, with 50 levels, and (T3) change level, with five levels. Given this design, we generated 7500 observations (excluding the 50 observations for the baseline) at the statement level and 7500 observations at the function level.3 Each observation contained the four metrics presented in Section 2.1, for each level of the three factors. 3.3 Results and Analysis First, we examine the data graphically. Box plots for each metric at the baseline and the five change levels are presented in Figure 1.4 3 The column of graphs on the left depicts results for statement coverage; the column on the right depicts results for function coverage; the four graphs in each column depict results for each of the four metrics (MD, CC, CAC, CAT), respectively.

The x-axes represents the baseline and five change levels. The y-axes represent the values measured for each metric. Each individual box plot (other than the plots for baselines) indicates the distribution of the 1500 observations (50 test suites times 30 versions) for one change level, for a given metric and coverage type. (Plots for baselines, where no change levels are involved, represent 50 observations.) As the plots indicate, MD (matrix density) decreased as change level increased. Both statement and function coverage presented the same trend. However, MD based on function coverage information exhibited a greater rate of decrease and greater deviation (as indicated by the size of the box plots). CC (component coverage) also decreased as change level increased. As expected, CC at the function level was higher at lower change levels, but at higher change levels it rapidly decreased to values similar to CC at the statement level. (Recall from Section 3.1 that Space contains some unreachable code and functions; this explains why CC is not 100% for either type of coverage, even for the baseline version.) It is interesting to note that the mean CC decreases more than 10% if even 1% of the branches are affected by a change. That decreased percentage rose to 20% when 2% of the branches were affected by a change.

The author has previously presented a novel two-step approach to automatic record pair classification. In the first step of this approach, training examples of high quality are automatically selected from the compared record pairs, and used in the second step to train a support vector machine (SVM) classifier. Initial experiments showed the feasibility of the approach, achieving results that outperformed *k*-means clustering. In this paper, two variations of this approach are presented. The first is based on a nearest neighbor classifier, while the second improves a SVM classifier by iteratively adding more examples into the training sets. Experimental results show that this two-step approach can achieve better classification results than other unsupervised approaches.

A Method for Calibrating False-Match Rates in Record Linkage

Specifying a record-linkage procedure requires both (1) a method for measuring closeness of agreement between records, typically a scalar weight, and (2) a rule for deciding when to classify records as matches or non matches based on the weights. Here we outline a general strategy for the second problem, that is, for accurately estimating false-match rates for each possible cutoff weight. The strategy uses a model where the distribution of observed weights is viewed as a mixture of weights for true matches and weights for false matches. An EM algorithm for fitting mixtures of transformed-normal distributions is used to find posterior modes; associated posterior variability is due to uncertainty about specific normalizing transformations as well as uncertainty in the parameters of the mixture model, the latter being calculated using the SEM algorithm. This mixture-model calibration method is shown to perform well in an applied setting with census data. Further, a simulation experiment reveals that, across a wide variety of settings not satisfying the model's assumptions, the procedure is slightly conservative on average in the sense of overstating false-match rates, and the one-sided confidence coverage (i.e., the proportion of times that these interval estimates cover or overstate the actual false-match rate) is very close to the nominal rate.

Applying Model Management to Classical Meta Data Problems

We wish to measure the evidence that a pair of records relates to the same, rather than different, individuals. The paper emphasizes statistical models which can be fitted to a file of record pairs known to be correctly matched, and then used to estimate likelihood ratios. A number of models are developed and applied to UK immigration statistics. The combination of likelihood ratios for possibly correlated record fields .

To determine which change levels actually differed, we performed a Bonferroni analysis (see Table 5). For each metric, the table presents the mean, and a grouping letter to represent the Bonferroni results (change levels with the same letter are not significantly different). Although the averages for all metrics seem to grow closer as change level increases, we found that only in one case (comparing CAT at the 10% and 15% levels for statement coverage) are the metrics not significantly different. Finally, we performed an analysis to compare the metrics means between types of coverage information (see Table 6). The ANOVA indicated that, when compared at each change level, statement and function coverage information generated significantly different metrics. As expected, most of the comparisons indicated that function coverage information tends to have higher values across all change levels for MD and CC. This is intuitively plausible when we consider that it is common for a change that alters statement coverage not to modify the functional control flow.

The interpretation of the change metrics was not as intuitive. For CAC, the analysis indicated that function coverage information was more susceptible to changes in the program than was statement level information. That means that although CC may have remained similar between versions, the functions that were covered were different. CAT on the other hand shows different results depending on the change level. At the 1% and 2% levels, function coverage information follows the same trend as CAC. This trend, however, is reversed at higher change levels, possibly reflecting different thresholds on maximum change.

Record Linkage Current Practice and Future Directions

Record linkage is the task quickly and accurately identifying records corresponding to the same entity from one or more data sources. Record linkage is also known as data cleaning, entity reconciliation or identification and the merge/purge problem. This paper presents the "standard" probabilistic record linkage model and the associated algorithm. Recent work in information retrieval, federated database systems and data mining have proposed alternatives to key components of the standard algorithm. The impact of these alternatives on the standard approach are assessed.

III. Results:

Fig:User login page

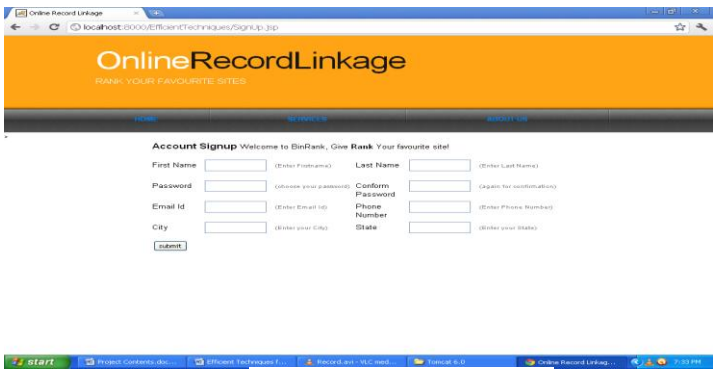
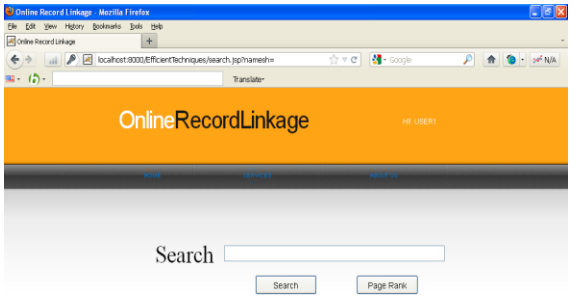
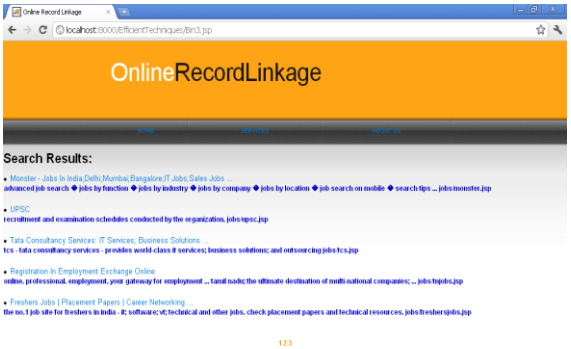
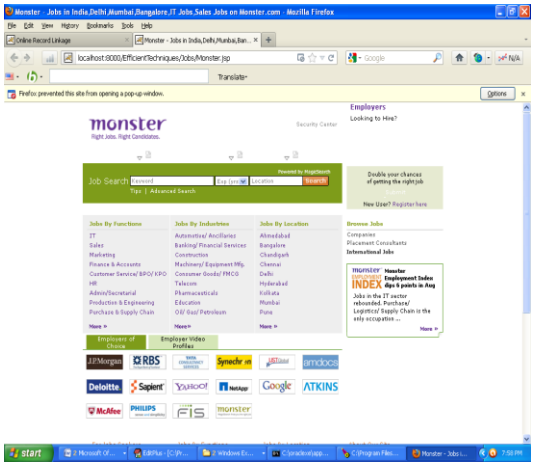
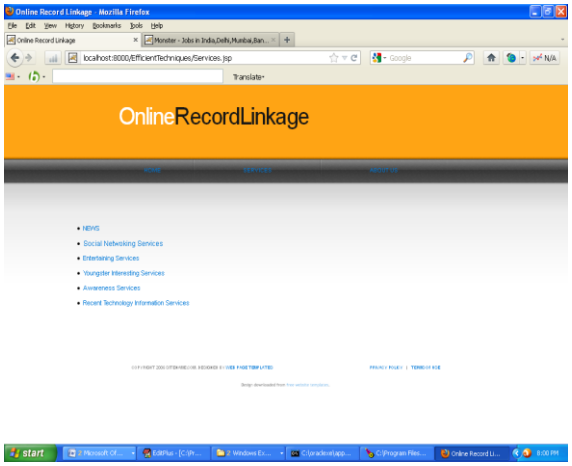


Fig: User Registration Page

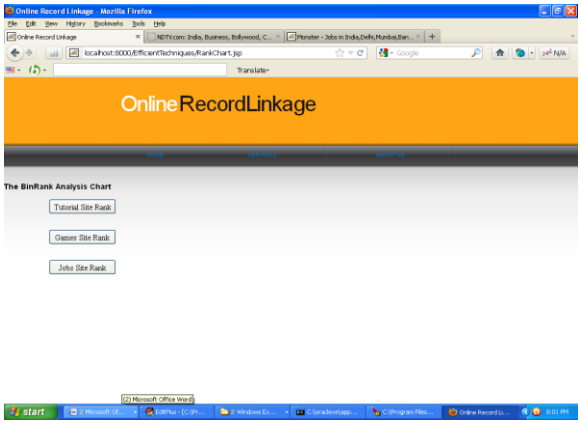

OnlineRecordLinkage
Search
Search
Page Rank

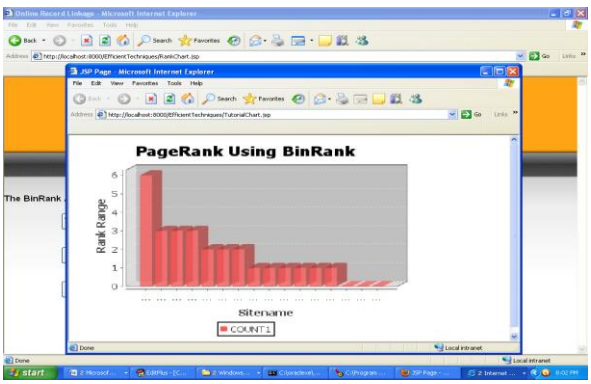

OnlineRecordLinkage
Search Results:


monster


OnlineRecordLinkage


NDTV
HTTP Status 404 -
/EfficientTechniques
/Services
/News/NDTV


OnlineRecordLinkage
The BinRank Analysis Chart
Tutorial Site Rank
Games Site Rank
John Site Rank


PageRank Using BinRank
Rank Range
Sitename
COUNT(1)

IV. Conclusion:

Efficient techniques to facilitate record linkage decisions in a distributed, online setting. Record linkage is an important issue in heterogeneous database systems where the records representing the same real-world entity type are identified using different identifiers in different databases. In the absence of a common identifier, it is often difficult to find records in a remote database that are similar to a local enquiry record. Traditional record linkage uses a probability-based model to identify the closeness between records. The matching probability is computed based on common attribute values. This, of course, requires that common attribute values of all the remote records be transferred to the local site. The communication overhead is significantly large for such an operation. Propose techniques for record linkage that draw upon previous work in sequential decision making.More specifically, we develop a matching tree for attribute acquisition and propose three different schemes of using this tree for record linkage.

Future Enhancement

Although this work presents a better approach for version differentiation to select the exact updated coverage from old version to new version by using the statistical information, we can still improve by automating without selecting the updated coverage manually by the test engineer. By this we can still reduce the testing budget and maintenance.

In regression testing many techniques are proposed that provides an efficient way of selecting regression test suite without rerunning all test cases that was developed for old version. But still IT companies are not utilizing these techniques, because these techniques are not assuring completely and test engineers are still using manual testing and automation testing for safety. I can assure that my proposed approach with enhancement is safe and precise in that it computes exactly the same information as if all test cases in the test suite were rerun.

References

[1] J.A. Baldwin, "Linked Record Health Data Systems," The Statistician, vol. 21, no. 4, pp. 325-338, 1972.
[2] C. Batini, M. Lenzerini, and S.B. Navathe, "A Comparative Analysis of Methodologies for Database Schema Integration,"ACM Computing Surveys, vol. 18, no. 4, pp. 323-364, 1986.
[3] R. Baxter, P. Christen, and T. Churches, "A Comparison of Fast Blocking Methods for Record Linkage," Proc. ACM Workshop Data Cleaning, Record Linkage and Object Consolidation, pp. 25-27, Aug.2003.
[4] T.R. Belin and D.B. Rubin, "A Method for Calibrating False-Match Rates in Record Linkage," J. Am. Statistical Assoc., vol. 90, no. 430,pp. 694-707, 1995.
[5] P. Bernstein, "Applying Model Management to Classical Meta Data Problems," Proc. Conf. Innovative Database Research (CIDR),pp. 209-220, Jan. 2003.

Books

1. Java 2 complete Reference by Herbert Schieldt
2. "Software Engineering", A Practitioner's Approach, 6th Edition, Tata McGrawHill
3. "Software Testing principles and practices", Srinivasan Desikan, Gopala swamiRamesh, Pearson edition, India.
4. A Unified Modeling Language User Guide, 2nd edition, Book by Grady Booch, James RamBaugh, IvarJacobson for UML concepts and models.

Websites

1.www.google.co.in/WirelessMeshNetwork.doc
2.www.wikiedia/Network-Security.com
3.www.ieeeexplore.ieee.org/xpl/anonymous/networks.pdf