ISSN:2320-0790

An International Journal of Advanced Computer Technology

Cost-Effective Resource Allocation of Overlay Routing Relay Nodes

B.Sandhya Rani[1], K.Pradeep Reddy[2], M.Parimala[3], Prof.Dr.G.Manoj Someswar[4]

[1]M.Tech., Tirumala Engineering College, Affiliated to JNTUH, Hyderabad, Telangana, India
[2]M.Tech.(CSE), Associate Professor, Tirumala Engineering College, Affiliated to JNTUH, Hyderabad, Telangana, India
[3]M.Tech.(CSE), Associate Professor, Tirumala Engineering College, Affiliated to JNTUH, Hyderabad, Telangana, India
[4]B.Tech., M.S.(USA), M.C.A., Ph.D., Director General & Scientist 'G', Global Research Academy, Hyderabad, Telangana, India

Abstract: Overlay routing is a very attractive scheme that allows improving certain properties of the routing (such as delay or TCP throughput) without the need to change the standards of the current underlying routing. However, deploying overlay routing requires the placement and maintenance of overlay infrastructure. This gives rise to the following optimization problem: Find a minimal set of overlay nodes such that the required routing properties are satisfied. In this research paper, we rigorously study this optimization problem. We show that it is NP-hard and derive a nontrivial approximation algorithm for it, where the approximation ratio depends on specific properties of the problem at hand. We examine the practical aspects of the scheme by evaluating the gain one can get over several real scenarios. The first one is BGP routing, and we show, using up-to-date data reflecting the current BGP routing policy in the Internet, that a relative small number of less than 100 relay servers is sufficient to enable routing over shortest paths from a single source to all autonomous systems (ASs), reducing the average path length of inflated paths by 40%. We also demonstrate that the scheme is very useful for TCP performance improvement (results in an almost optimal placement of overlay nodes) and for Voice-over-IP (VoIP) applications where a small number of overlay nodes can significantly reduce the maximal peer-to-peer delay.

Keywords: Autonomous Systems, Voice-over-IP, Round-Trip Time (RTT), Overlay Routing Resource Allocation (ORRA), Voice-over-IP(VoIP), Overlay Router, BGP Router, Virtual Private Network (VPN)

I. INTRODUCTION

OVERLAY routing has been proposed in recent years as an effective way to achieve certain routing properties, without going into the long and tedious process of standardization and global deployment of a new routing protocol. For example, in [1], overlay routing was used to improve TCP performance over the Internet, where the main idea is to break the end-to-end feedback loop into smaller loops. This requires that nodes capable of performing TCP Piping would be present along the route at relatively small distances. Other examples for the use of overlay routing are projects like RON [2] and Detour [3], where overlay routing is used to improve reliability. Yet another example is the concept of the "Global-ISP" paradigm introduced in [4], where an overlay node is used to reduce latency in BGP routing.

In order to deploy overlay routing over the actual physical infrastructure, one needs to deploy and manage overlay nodes that will have the new extra functionality. This comes with a non negligible cost both in terms of capital and operating costs. Thus, it is important to study the benefit one gets from improving the routing metric against this cost.

In this research paper, we concentrate on this point and study the minimum number of infrastructure nodes that need to be added in order to maintain a specific property in the overlay routing. In the shortest-path routing over the Internet BGP-based routing example, this question is mapped to: What is the minimum number of relay nodes that are needed in order to make the routing between a group of autonomous systems (ASs) use the underlying shortest path between them? In the TCP performance example, this may translate to: What is the minimal number of relay nodes needed in order to make sure that for each TCP connection, there is a path between the connection endpoints for which every predefined round-trip time (RTT), there is an overlay node capable of TCP Piping?

Regardless of the specific implication in mind, we define a general optimization problem called the Overlay Routing Resource Allocation (ORRA) problem and study its complexity. It turns out that the problem is NP-hard, and we present a nontrivial approximation algorithm for it.

Note that if we are only interested in improving routing properties between a single source node and a single destination, then the problem is not complicated, and finding the optimal number of nodes becomes trivial since the potential candidate for overlay placement is small, and in general any assignment would be good. However, when we consider one-to-many or many-to-many scenarios, then a single overlay node may affect the path property of many paths, and thus choosing the best locations becomes much less trivial.

We test our general algorithm in three specific such cases, where we have a large set of source–destination pairs, and the goal is to find a minimal set of locations, such that using overlay nodes in these locations allows to create routes (routes are either underlay routes or routes that use these new relay nodes) such that a certain routing property is satisfied.[5]

The first scenario we consider is AS-level BGP routing, where the goal is to find a minimal number of relay node locations that can allow shortest-path routing between the source–destination pairs. Recall that routing in BGP is policy-based and depends on the business relationship between peering ASs, and as a result, a considerable fraction of the paths in the Internet do not go along a shortest path (see [5]). This phenomenon, called path inflation, is the motivation for this scenario. We consider a one-to-many setting where we want to improve routing between a single source and many destinations. This is the case where the algorithm power is most significant since, in the many-to-many setting, there is very little overlap between shortest paths, and thus not much improvement can be made over a basic greedy approach.

We demonstrate, using real up-to-date Internet data, that the algorithm can suggest a relatively small set of relay nodes that can significantly reduce latency in current BGP routing.

The second scenario we consider is the TPC improvement example discussed above. In this case, we test the algorithm on a synthetic random graph, and we show that the general framework can be applied also to this case, resulting in very close-to-optimal results.

The third scenario addresses overlay Voice-over-IP (VoIP) applications such as Skype (http://www.skype.com), Google Talk (http://www.google.com/talk/), and others. Such applications are Becoming more and more popular offering IP telephone services for free, but they need abounded end-to-end delay (or latency)between any pair of users to maintain a reasonable service quality. We show that our scheme can be very useful also in this case, allowing applications to choose a smaller number of hubs, yet improving performance for many users.[6]

Note that the algorithmic model we use assumes a full knowledge of the underlying topology, the desired routing scheme, and the locations of the required endpoints. In general, the algorithm is used by the entity that needs the routing improvement and carries the cost of establishing and maintaining overlay nodes, using the best available topology information. For example, in the VoIP case, the VoIP application is establishing the overlay nodes, and thus the application can gain by using our approach.

The main contributions of this research paper are as follows.

- We develop a general algorithmic framework that can be used in order to deal with efficient resource allocation in overlay routing.
- We develop a nontrivial approximation algorithm and prove its properties. We demonstrate the actual benefit one can gain from using our scheme in three practical scenarios, namely BPG routing, TCP improvement, and VoIP applications.

EXISTING SYSTEM

In the existing system, the system is using overlay routing to improve network performance is motivated by many works that studied the inefficiency of varieties of networking architectures and applications. Analyzing a large set of data, Savage *et al.* explore the question: How "good" is Internet routing from a user's perspective considering round-trip time, packet loss rate, and bandwidth? They showed that in 30%–80% of the cases, there is an alternate routing path with better quality compared to the default routing path. In the current system and later in existing system, the authors show that TCP performance is strictly affected by the RTT. Thus, breaking a TCP connection into low-latency sub connections improves the overall connection performance. In the existing system, the authors show that in many cases, routing paths in the Internet are inflated, and the actual length (in hops) of routing paths between clients is longer than the minimum hop distance between them. Using overlay routing to improve routing and network performance has been studied before in several works.[7]

In the existing system, the authors studied the routing inefficiency in the Internet and used an overlay routing in order to evaluate and study experimental techniques improving the network over the real environment. While the concept of using overlay routing to improve routing scheme was presented in this work, it did not deal with the deployment aspects and the optimization aspect of such infrastructure. A resilient overlay network (RON), which is architecture for application-layer overlay routing to be used on top of the existing Internet routing infrastructure, has been presented in the current system. Similar to our work, the main goal of this architecture is to replace the existing routing scheme, if necessary, using the overlay infrastructure. This work mainly focuses on the overlay infrastructure (monitoring and detecting routing problems, and maintaining the overlay system), and it does not consider the cost associated with the deployment of such system.

PROPOSED SYSTEM

In the proposed system, The system concentrates on this point and study the minimum number of infrastructure nodes that need to be added in order to maintain a specific property in the overlay routing. In the shortest-path routing over the Internet BGP-based routing example, this question is mapped to: What is the minimum number of relay nodes that are needed in order to make the routing between a groups of autonomous systems (ASs) use the underlying shortest path between them? In the TCP performance example, this may translate to: What is the minimal number of relay nodes needed in order to make sure that for each TCP connection, there is a path between the connection endpoints for which every predefined round-trip time (RTT), there is an overlay node capable of TCP Piping?

Regardless of the specific implication in mind, we define a general optimization problem called the Overlay Routing Resource Allocation (ORRA) problem and study its complexity. It turns out that the problem is NP-hard, and we present a nontrivial approximation algorithm for it.

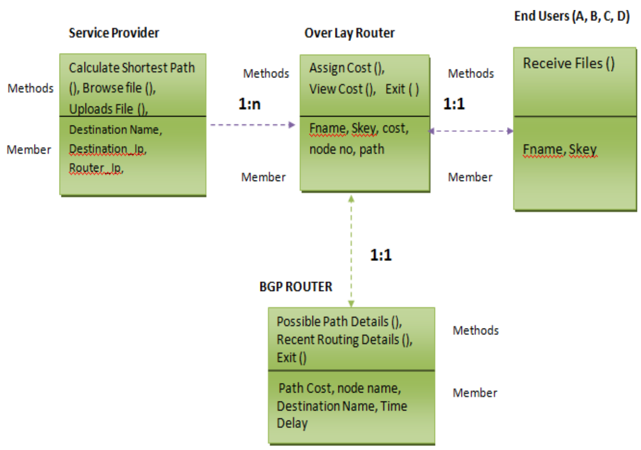
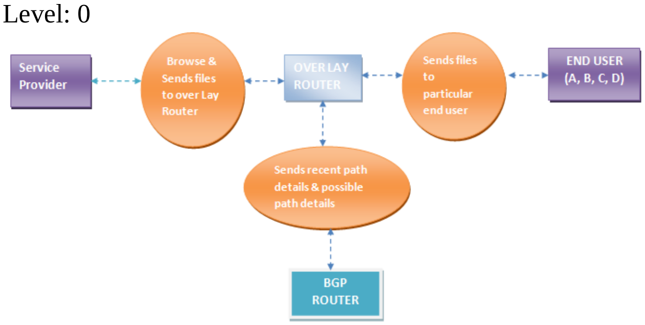


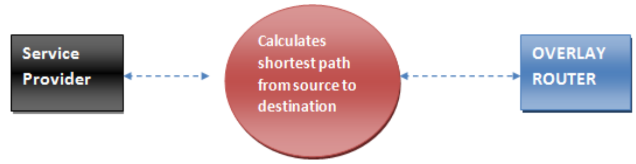
Figure 2: Data Flow Diagram

System Design

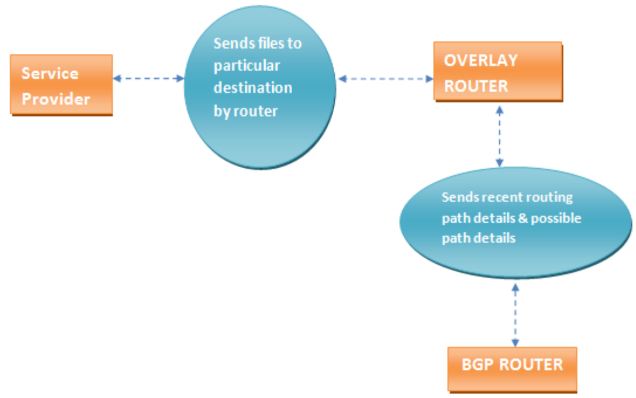
Level: 0



Level: 1



Level: 2



Level: 3

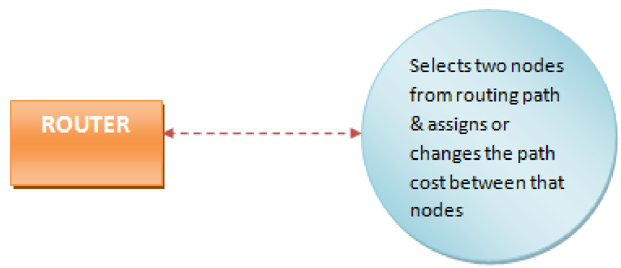


Figure 3: Data Flow Diagram

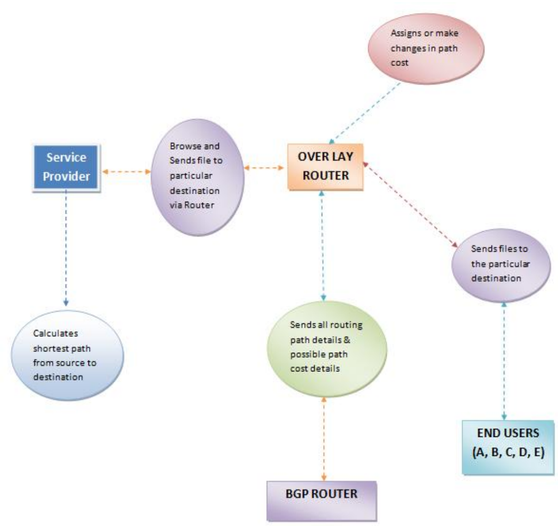


Figure 4: Flow Chart

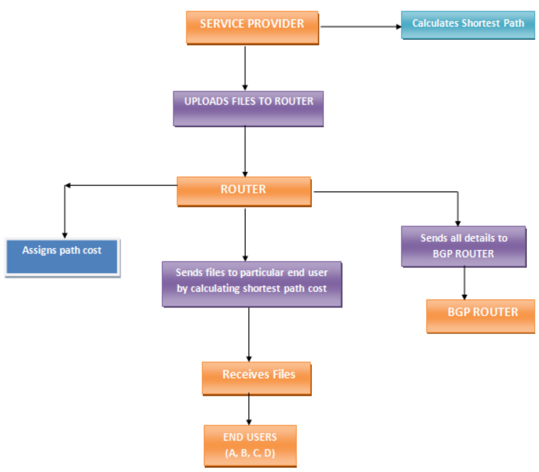


Figure 5: Sequence Diagram

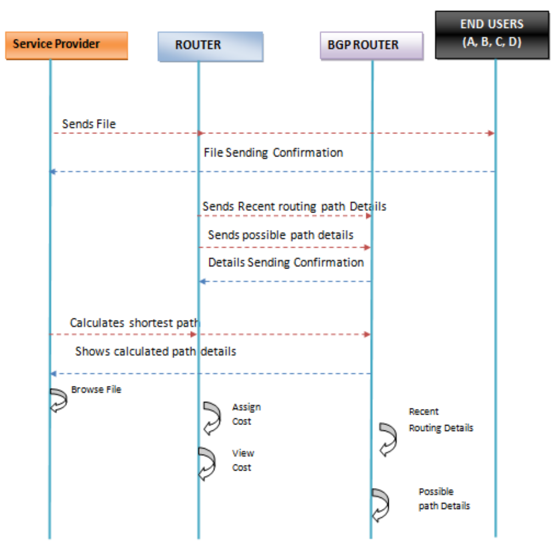
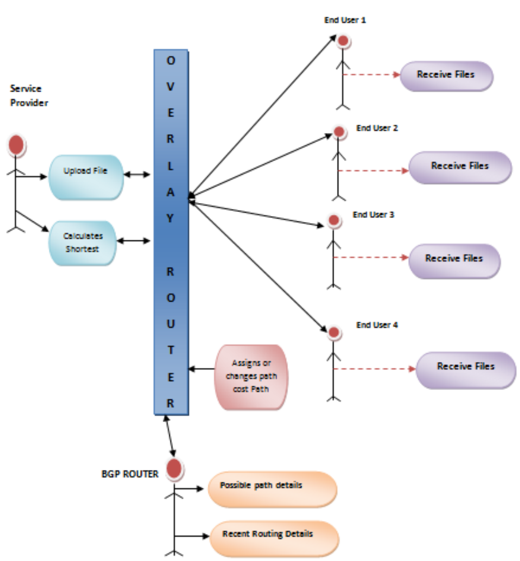


Figure 6: Use Case Diagram



INPUT DESIGN

The input design is the link between the information system and the user. It comprises the developing specification and procedures for data preparation and those steps are necessary to put transaction data in to a usable form for processing can be achieved by inspecting the computer to read data from a written or printed document or it can occur by having people keying the data directly into the system. The design of input focuses on controlling the amount of input required, controlling the errors, avoiding delay, avoiding extra steps and keeping the process simple. The input is designed in such a way so that it provides security and ease of use with retaining the privacy. Input Design considered the following things:

What data should be given as input?

How the data should be arranged or coded?

The dialog to guide the operating personnel in providing input.

Methods for preparing input validations and steps to follow when error occur.

What kind of output will be generating by the system?

1. The encrypted file from the service provider
2. The encrypted file from router after load balancing to the destination
3. The encrypted file and decrypting the file in Destination, storing into destination

OBJECTIVES

1. Input Design is the process of converting a user-oriented description of the input into a computer-based system. This design is important to avoid errors in the data input process and show the correct direction to the management for getting correct information from the computerized system.

2. It is achieved by creating user-friendly screens for the data entry to handle large volume of data. The goal of designing input is to make data entry easier and to be free from errors. The data entry screen is designed in such a way that all the data manipulates can be performed. It also provides record viewing facilities.

3. When the data is entered it will check for its validity. Data can be entered with the help of screens. Appropriate messages are provided as when needed so that the user will not be in maize of instant. Thus the objective of input design is to create an input layout that is easy to follow.

OUTPUT DESIGN

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any system results of processing are communicated to the users and to other system through outputs. In output design it is determined how the information is to be displaced for immediate need and also the hard copy output. It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship to help user decision-making.

1. Designing computer output should proceed in an organized, well thought out manner; the right output must be developed while ensuring that each output element is designed so that people will find the system can use easily and effectively. When analysis design computer output,

they should Identify the specific output that is needed to meet the requirements.
2. Select methods for presenting information.
3. Create document, report, or other formats that contain information produced by the system.
The output form of an information system should accomplish one or more of the following objectives.
Convey information about past activities, current status or projections of the Future.
Signal important events, opportunities, problems, or warnings.
Trigger an action.
Confirm an action.

SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the
Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. [8] This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.[9]

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:
Valid Input: identified classes of valid input must be accepted.
Invalid Input: identified classes of invalid input must be rejected.
Functions: identified functions must be exercised.
Output
Systems/Procedures: interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.[10]

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot "see" into it. The test provides inputs and responds to outputs without considering how the software works.[11]

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

All field entries must work properly.
Pages must be activated from the identified link.
The entry screen, messages and responses must not be delayed.
Features to be tested
Verify that the entries are of the correct format
No duplicate entries should be allowed
All links should take the user to the correct page.[12]

IMPLEMENTATION

- **Service Provider**
  In this module, the Service Provider calculates the shortest path to Destination, The shortest-path routing over the Internet BGP-based router. The Service provider browses the required file and uploads their data files to the Specified End User (A, B, C, D) and with their DIP (Destination IP) of End User.
- **Overlay Router**
  The Overlay Router is responsible to route the file to the specified destination, the overlay routing scheme is the set of the shortest physical paths simplifies the execution of this system, and finding a minimal path to the destination using overlay routing, one can perform routing via shortest paths, the router is also responsible for Assigning the cost and also can view the cost of nodes with their tags From the node (from), To the node (to) and the cost.
- **BGP Router**
  The BGP Router is responsible to route the nodes using BGP routing, where the goal is to find a minimal number of relay node locations that can allow shortest-path routing between the source–destination pairs, BGP Router consider a one-to-many destination where we want to improve routing between a single source and many destinations. BGP routing table contains valid paths from its source to the entire set of nodes. BGP is also responsible for storing the possible path to destination, can view the recent routing path to destination with their tags Filename, Recent Path, Destination, DIP, Delay and date and time.[13]
- **End User(Destination)**
  In this module, the End user (Node A, Node B, Node C, Node D) is responsible to receive the file from the Service Provider In the shortest-path routing between the source–destination nodes, the system consists of a one-to-many relationship. Where end User receives file from a single source to destination (Node A, Node B, Node C, Node D).[14]

PRELIMINARY INVESTIGATION

The first and foremost strategy for development of a project starts from the thought of designing a mail enabled platform for a small firm in which it is easy and convenient of sending and receiving messages, there is a search engine ,address book and also including some entertaining games. When it is approved by the organization and our project guide the first activity, ie. Preliminary investigation begins. The activity has three parts:

Request Clarification
Feasibility Study
Request Approval

REQUEST CLARIFICATION

After the approval of the request to the organization and project guide, with an investigation being considered, the project request must be examined to determine precisely what the system requires.

Here our project is basically meant for users within the company whose systems can be interconnected by the Local Area Network(LAN). In today's busy schedule man need everything should be provided in a readymade manner. So taking into consideration of the vastly use of the net in day to day life, the corresponding development of the portal came into existence.

FEASIBILITY ANALYSIS

An important outcome of preliminary investigation is the determination that the system request is feasible. This is possible only if it is feasible within limited resource and time. The different feasibilities that have to be analyzed are

Operational Feasibility
Economic Feasibility
Technical Feasibility

Operational Feasibility

Operational Feasibility deals with the study of prospects of the system to be developed. This system operationally eliminates all the tensions of the Admin and helps him in effectively tracking the project progress. This kind of automation will surely reduce the time and energy, which previously consumed in manual work. Based on the study, the system is proved to be operationally feasible.[15]

Economic Feasibility

Economic Feasibility or Cost-benefit is an assessment of the economic justification for a computer based project. As hardware was installed from the beginning & for lots of purposes thus the cost on project of hardware is low. Since the system is a network based, any number of employees connected to the LAN within that organization can use this tool from at anytime. The Virtual Private Network is to be developed using the existing resources of the organization. So the project is economically feasible.[16]

Technical Feasibility

According to Roger S. Pressman, Technical Feasibility is the assessment of the technical resources of the organization. The organization needs IBM compatible machines with a graphical web browser connected to the Internet and Intranet.[17] The system is developed for platform Independent environment. Java Server Pages, JavaScript, HTML, SQL server and WebLogic Server are used to develop the system. The technical feasibility has been carried out. The system is technically feasible for development and can be developed with the existing facility.

REQUEST APPROVAL

Not all request projects are desirable or feasible. Some organization receives so many project requests from client users that only few of them are pursued. However, those projects that are both feasible and desirable should be put into schedule. After a project request is approved, it cost, priority, completion time and personnel requirement is estimated and used to determine where to add it to any project list. Truly speaking, the approval of those above factors, development works can be launched.

SYSTEM DESIGN AND DEVELOPMENT

INPUT DESIGN

Input Design plays a vital role in the life cycle of software development, it requires very careful attention of developers. The input design is to feed data to the application as accurate as possible. So inputs are supposed to be designed effectively so that the errors occurring while feeding are minimized. According to Software Engineering Concepts, the input forms or screens are designed to provide to have a validation control over the input limit, range and other related validations.

This system has input screens in almost all the modules. Error messages are developed to alert the user whenever he commits some mistakes and guides him in the right way so that invalid entries are not made. Let us see deeply about this under module design.

Input design is the process of converting the user created input into a computer-based format. The goal of the input design is to make the data entry logical and free from errors. The error is in the input are controlled by the input design. The application has been developed in user-friendly manner. The forms have been designed in such a way during the processing the cursor is placed in the position where must be entered. The user is also provided with in an option to select an appropriate input from various alternatives related to the field in certain cases.

Validations are required for each data entered. Whenever a user enters an erroneous data, error message is displayed and the user can move on to the subsequent pages after completing all the entries in the current page.

OUTPUT DESIGN

The Output from the computer is required to mainly create an efficient method of communication within the company primarily among the project leader and his team members, in other words, the administrator and the clients. The output of VPN is the system which allows the project leader to manage his clients in terms of creating new clients and assigning new projects to them, maintaining a record of the project validity and providing folder level access to each client on the user side depending on the projects allotted to him. After completion of a project, a new project may be assigned to the client. User authentication procedures are maintained at the initial stages itself. A new user may be created by the administrator himself or a user can himself register as a new user but the task of assigning projects and validating a new user rests with the administrator only.

The application starts running when it is executed for the first time. The server has to be started and then the internet explorer in used as the browser. The project will run on the local area network so the server machine will serve as the administrator while the other connected systems can act as the clients. The developed system is highly user friendly and can be easily understood by anyone using it even for the first time.

SYSTEM TESTING

TESTING METHODOLOGIES

The following are the Testing Methodologies:

Unit Testing.
Integration Testing.
User Acceptance Testing.
Output Testing.
Validation Testing.

Unit Testing

Unit testing focuses verification effort on the smallest unit of Software design that is the module. Unit testing exercises specific paths in a module's control structure to ensure complete coverage and maximum error detection. This test focuses on each module individually, ensuring that it functions properly as a unit. Hence, the naming is Unit Testing.[18]

During this testing, each module is tested individually and the module interfaces are verified for the consistency with design specification. All important processing path are tested for the expected results. All error handling paths are also tested.

Integration Testing

Integration testing addresses the issues associated with the dual problems of verification and program construction. After the software has been integrated a set of high order tests are conducted. The main objective in this testing process is to take unit tested modules and builds a program structure that has been dictated by design.[19]

The following are the types of Integration Testing:

Top Down Integration

This method is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main program module. The module subordinates to the main program module are incorporated into the structure in either a depth first or breadth first manner.[20]

In this method, the software is tested from main module and individual stubs are replaced when the test proceeds downwards.

Bottom-up Integration

This method begins the construction and testing with the modules at the lowest level in the program structure. Since the modules are integrated from the bottom up, processing required for modules subordinate to a given level is always available and the need for stubs is eliminated. The bottom up integration strategy may be implemented with the following steps:

The low-level modules are combined into clusters into clusters that perform a specific Software sub-function.
A driver (i.e.) the control program for testing is written to coordinate test case input and output.
The cluster is tested.
Drivers are removed and clusters are combined moving upward in the program structure.

The bottom up approaches tests each module individually and then each module is module is integrated with a main module and tested for functionality.

User Acceptance Testing

User Acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system users at the time of developing and making changes wherever required. The system developed provides a friendly user interface that can easily be understood even by a person who is new to the system. [21]

Output Testing

After performing the validation testing, the next step is output testing of the proposed system, since no system could be useful if it does not produce the required output in the specified format. Asking the users about the format required by them tests the outputs generated or displayed by the system under consideration. Hence the output format is considered in 2 ways – one is on screen and another in printed format.

Validation Checking

Validation checks are performed on the following fields.

Text Field:

The text field can contain only the number of characters lesser than or equal to its size. The text fields are alphanumeric in some tables and alphabetic in other tables. Incorrect entry always flashes and error message.

Numeric Field

The numeric field can contain only numbers from 0 to 9. An entry of any character flashes an error messages. The individual modules are checked for accuracy and what it has to perform. Each module is subjected to test run along with sample data. The individually tested modules are integrated into a single system. Testing involves executing the real data information is used in the program the existence of any program defect is inferred from the output. The testing should be planned so that all the requirements are individually tested.

A successful test is one that gives out the defects for the inappropriate data and produces and output revealing the errors in the system.

Preparation of Test Data

Taking various kinds of test data does the above testing. Preparation of test data plays a vital role in the system testing. After preparing the test data the system under study is tested using that test data. While testing the system by using test data errors are again uncovered and corrected by using above testing steps and corrections are also noted for future use.

Using Live Test Data

Live test data are those that are actually extracted from organization files. After a system is partially constructed, programmers or analysts often ask users to key in a set of data from their normal activities. Then, the systems person uses this data as a way to partially test the system. In other instances, programmers or analysts extract a set of live data from the files and have them entered themselves.

It is difficult to obtain live data in sufficient amounts to conduct extensive testing. And, although it is realistic data that will show how the system will perform for the typical processing requirement, assuming that the live data entered are in fact typical, such data generally will not test all combinations or formats that can enter the system. This bias toward typical values then does not provide a true systems test and in fact ignores the cases most likely to cause system failure.

Using Artificial Test Data

Artificial test data are created solely for test purposes, since they can be generated to test all combinations of formats and values. In other words, the artificial data, which can quickly be prepared by a data generating utility program in the information systems department, make possible the testing of all login and control paths through the program.

The most effective test programs use artificial test data generated by persons other than those who wrote the programs. Often, an independent team of testers formulates a testing plan, using the systems specifications.

The package "Virtual Private Network" has satisfied all the requirements specified as per software requirement specification and was accepted.

USER TRAINING

Whenever a new system is developed, user training is required to educate them about the working of the system so that it can be put to efficient use by those for whom the system has been primarily designed. For this purpose the normal working of the project was demonstrated to the prospective users. Its working is easily understandable and since the expected users are people who have good knowledge of computers, the use of this system is very easy.

MAINTAINENCE

This covers a wide range of activities including correcting code and design errors. To reduce the need for maintenance in the long run, we have more accurately defined the user's requirements during the process of system development. Depending on the requirements, this system has been developed to satisfy the needs to the largest possible extent. With development in technology, it may be possible to add many more features based on the requirements in future. The coding and designing is simple and easy to understand which will make maintenance easier.

TESTING STRATEGY

A strategy for system testing integrates system test cases and design techniques into a well planned series of steps that results in the successful construction of software. The testing strategy must co-operate test planning, test case design, test execution, and the resultant data collection and evaluation .A strategy for software testing must accommodate low-level tests that are necessary to verify that a small source code segment has been correctly implemented as well as high level tests that validate major system functions against user requirements.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification design and coding. Testing represents an interesting anomaly for the software. Thus, a series of testing are performed for the proposed system before the system is ready for user acceptance testing.

SYSTEM TESTING

Software once validated must be combined with other system elements (e.g. Hardware, people, database). System testing verifies that all the elements are proper and that overall system function performance is achieved. It also tests to find discrepancies between the system and its original objective, current specifications and system documentation.

UNIT TESTING

In unit testing different are modules are tested against the specifications produced during the design for the modules. Unit testing is essential for verification of the code produced during the coding phase, and hence the goals to test the internal logic of the modules. Using the detailed design description as a guide, important Conrail paths are tested to uncover errors within the boundary of the modules. This testing is carried out during the programming stage itself. In this type of testing step, each module was found to be working satisfactorily as regards to the expected output from the module.

In due course, latest technology advancements will be taken into consideration. As part of technical build-up many components of the networking system will be generic in nature so that future projects can either use or interact with this. The future holds a lot to offer to the development and refinement of this project.

RESULTS & CONCLUSION

While using overlay routing to improve network performance was studied in the past by many works both practical and theoretical, very few of them consider the cost associated with the deployment of overlay infrastructure. In this paper, we addressed this fundamental problem developing an approximation algorithm to the problem. Rather than considering a customized Algorithm for a specific application or scenario, we suggested a general framework that fits a large set of overlay applications. Considering three different practical scenarios, we evaluated the Performance of the algorithm, showing that in practice the algorithm provides close-to-optimal results.

Many issues are left for further research. One interesting direction is an analytical study of the vertex cut used in the algorithm. It would be interesting to find properties of the underlay and overlay routing that assure a bound on the size of the cut. It would be also interesting to study the performance of our framework for other routing scenarios and to study issues related to actual implementation of the scheme.

REFERENCES

[1] H. Pucha and Y. C. Hu, "Overlay TCP: Multi-hop overlay transport for high throughput transfers in the Internet," Purdue University, West Lafayette, IN, USA, Tech. Rep., 2005.

[2] D. Andersen, H. Balakrishnan, F. Kaashoek, and R. Morris, "Resilient overlay networks," in *Proc. 18th ACM SOSP*, 2001, pp. 131–145.

[3] S. Savage, T. A. A. Aggarawl, T. Anderson, A. Aggarwal, D. Becker, N. Cardwell, A. Collins, E. Hoffman, J. Snell, A. Vahdat, G. Voelker, and J. Zahorjan, "Detour: A case for informed internet routing and transport," *IEEE Micro*, vol. 19, no. 1, pp. 50–59, Jan.–Feb. 1999.

[4] R. Cohen and A. Shochot, "The "global-ISP" paradigm," Comput. *Netw.*, vol. 51, no. 8, pp. 1908–1921, 2007.

[5] L. Gao and F. Wang, "The extent of as path inflation by routing policies," in *Proc. IEEE GLOBECOM*, 2002, vol. 3, pp. 2180–2184.

[6] S. Savage, A. Collins, E. Hoffman, J. Snell, and T. Anderson, "The end-to-end effects of Internet path selection," in *Proc. ACM SIGCOMM*, 1999, pp. 289–299.

[7] R. Cohen and S. Ramanathan, "Using proxies to enhance TCP performance over hybrid fiber coaxial networks," *Comput. Commun*, vol. 20, no. 16, pp. 1502–1518, Jan. 1998.

[8] N. Spring, R. Mahajan, and T. Anderson, "The causes of path inflation," in *Proc. ACM SIGCOMM*, 2003, pp. 113–124.

[9] H. Tangmunarunkit, R. Govindan, S. Shenker, and D. Estrin, "The impact of routing policy on Internet paths," in *Proc. IEEE INFOCOM*, 2001, pp. 736–742.

[10] M. Cha, S. Moon, C.-D. Park, and A. Shaikh, "Placing relay nodes for intra-domain path diversity," in *Proc. IEEE INFOCOM*, Apr. 2006, pp. 1–12.

[11] M. Kodialam, T. V. Lakshman, and S. Sengupta,"Efficient and robust routing of highly variable traffic," in *Proc. HotNets III*, 2004.

[12] S. Roy, H. Pucha, Z. Zhang, Y. C. Hu, and L. Qiu, "On the placement of infrastructure overlay nodes," *IEEE/ACM Trans. Netw.*, vol. 17, no. 4, pp. 1298–1311, Aug. 2009.

[13] L. Qiu, V. N. Padmanabhan, and G. M. Voelker "On the placement of Web server replicas," in *Proc. IEEE INFOCOM*, 2001, vol. 3, pp. 1587–1596.

[14] E. Cronin, S. Jamin, C. Jin, A. R. Kurc, D. Raz, and Y. Shavitt, "Constrained mirror placement on the Internet," in *Proc. IEEE INFOCOM*, 2001, vol. 1, pp. 31–40.

[15] M. R. Garey and D. S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness*. San Francisco, CA, USA: Freeman, 1979.

[16] U. Feige, "A threshold of ln n for approximating set cover," *J. ACM*, vol. 45, no. 4, pp. 634–652, 1998.

[17] R. Bar-Yehuda and S. Even, "A local-ratio theorem for approximating the weighted vertex cover problem," *Ann. Discrete Math.*, vol. 25, pp. 27–46, 1985.

[18] Y. Rekhter, T. Watson, and T. Li, "Border gateway protocol 4," RFC 1771, 1995.

[19] C. Alaettinoglu, "Scalable router configuration for the Internet," in *Proc. IEEE IC3N*, Oct. 1996.

[20] G. Huston, "Interconnection, peering and settlement," Internet *Protocol J.*, vol. 2, no. 1, Mar. 1999.

[21] L. Gao and J. Rexford, "Stable Internet routing without global coordination," *IEEE/ACM Trans. Netw.*, vol. 9, no. 6, pp. 681–692, Dec. 2001.